Java Flight Simulator

A Six-Degree-of-Freedom Flight Simulator Developed in Java

Christopher Ali

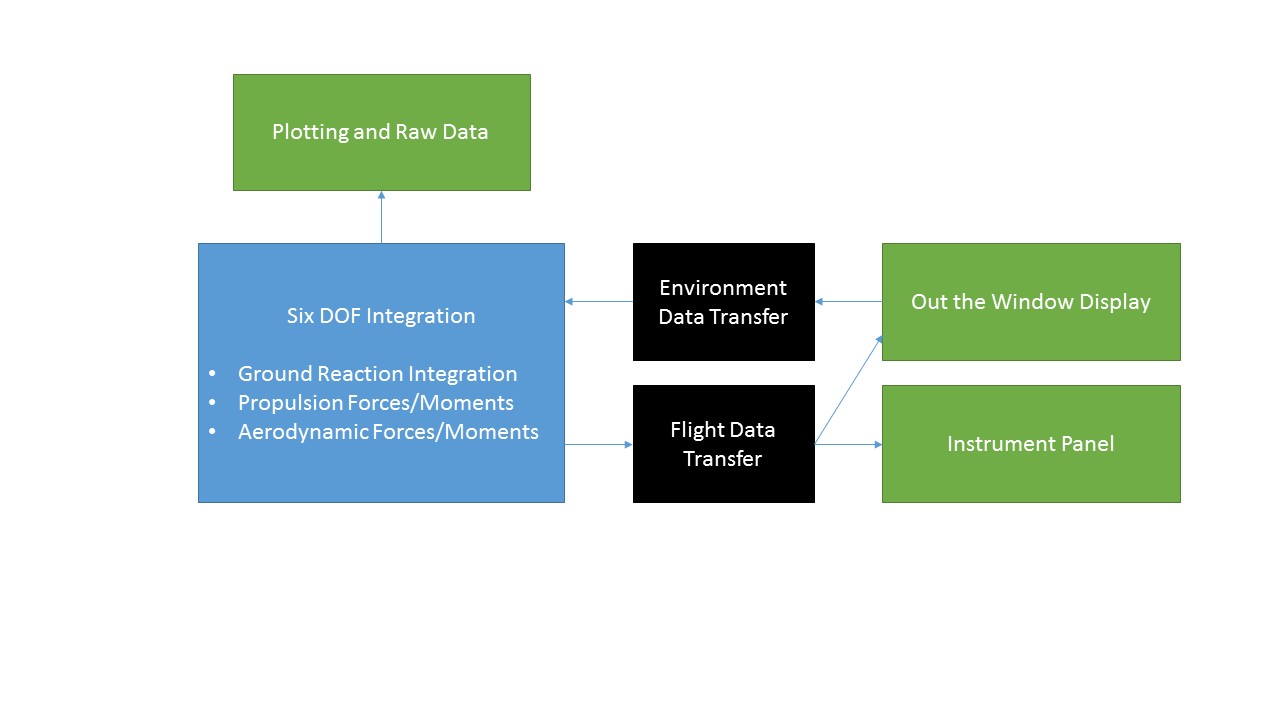
# Introduction

A personal project of mine to apply my knowledge and background in Flight Dynamics and Flight Simulation to build my Java language skills, Java Flight Simulator is meant to fuse two different worlds of flight simulation: one is the engineering flight simulator, which flight dynamics and aerodynamic engineers use to analyze stability, control and response of an aircraft; another is the real-time flight simulator, which test pilots use to assess flying qualities and performance.

In Java Flight Simulator, the user can easily select between analysis or flight, and then be presented with flight data visually via plots, or in tabulated form that can be exported as a CSV file.

The properties of aircraft in Java Flight Simulator can easily be changed by editing configuration files that correspond to aerodynamics, mass properties and propulsion to create a custom aircraft of one’s own imagination. As development of Java Flight Simulator progresses, custom aircraft development will become more accessible with the integration of USAF Digital DATCOM (<http://www.pdas.com/datcom.html>), a program capable of calculating stability and control derivatives for an input of lifting surface geometry and flight conditions.

# Simulation Overview

The below figure shows an overview of the framework of the simulation and how data is transferred. 

Green represents GUIs or display windows, black represents data transfer “intermediaries” that convert data to fit the destination as needed. Blue represents the core simulation, where the numerical integration to drive the simulation takes place.

Inside the Six DOF (degree of freedom) integration block, a loop runs the Apache Commons first degree Runge-Kutta numerical integrator (<http://commons.apache.org/proper/commons-math/>) to simultaneously integrate 12 6DOF state equations. After each step, the results are used to calculate forces and moments that are factored back into the 6DOF equations.

All variables are then organized into a Map data structure, which is accessed by the Flight Data Transfer block to send flight data to the out the window instrument panel displays. A List of these Maps is also used to plot the simulation results. Likewise, the Environment Data block gathers data from the out the window display to send back to the simulation.

# Simulation Limitations

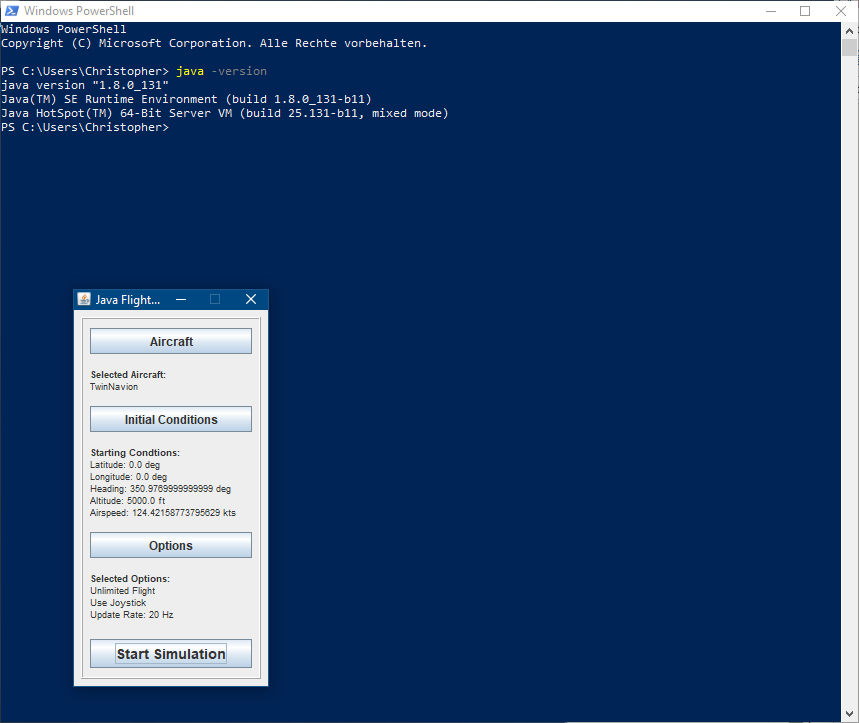
* Flight is restricted to subsonic flight, as Mach effects on aerodynamics are not simulated
* The out the window display only displays a fixed, flat area in daytime with moderate cloud cover
* The simulation is meant for operations in flight, as while a simple ground reaction model is included, it does not provide the realism for takeoff, landing or ground handling. As the development of the simulation progresses, the ground reaction model will be improved upon.
* No fuel burn model is implemented yet, so the aircraft maintains a constant mass.
* Only simple fixed-pitch propeller engines are supported as engine types for aircraft
* Joystick axes, buttons and keyboard inputs are currently fixed

# Requirements

## Java

Java Flight Simulator was developed entirely in Java, so users must have at least Java 8 SE installed on their computer. Many computers have Java already installed as part of other software or browser requirements. To verify that your computer has Java installed:

1. Open the Command Prompt or terminal if using Linux or OSX
2. Type *java -version* and press ENTER
3. *java version “1.8.x.x”*should be displayed in the window



If some similar output is not shown, or an error is returned, Java can be downloaded from the following website:

<https://java.com/en/download/>

## Additional System Requirements

* Processor: Dual Core (2.0+ GHz) or better
* RAM: 4GB or higher
* Hard Drive Space: 200 MB free
* Graphics: At least 256 MB VRAM
* Display Resolution: 1440 x 900 pixels or more
* Input Devices: Mouse and Keyboard

## Controllers

Although at least a keyboard and mouse are required for controlling the simulation and navigating the menus, a joystick or flight controls such as a yoke, rudder and throttle quadrant are strongly recommended to improve realism and increase controllability.

Flight controls in Java Flight Simulator are optimized for use with the CH Flight Sim Yoke, Pro Pedals and Throttle Quadrant (<http://www.chproducts.com/>), although any joystick or controller setup should provide similar functionality.

# Installation

Java Flight Simulator is installed by extracting the ZIP archive to a folder of choice. The application can run anywhere on the computer as long as the folder structure within the Java Flight Simulator folder is unchanged.

## Folder Structure

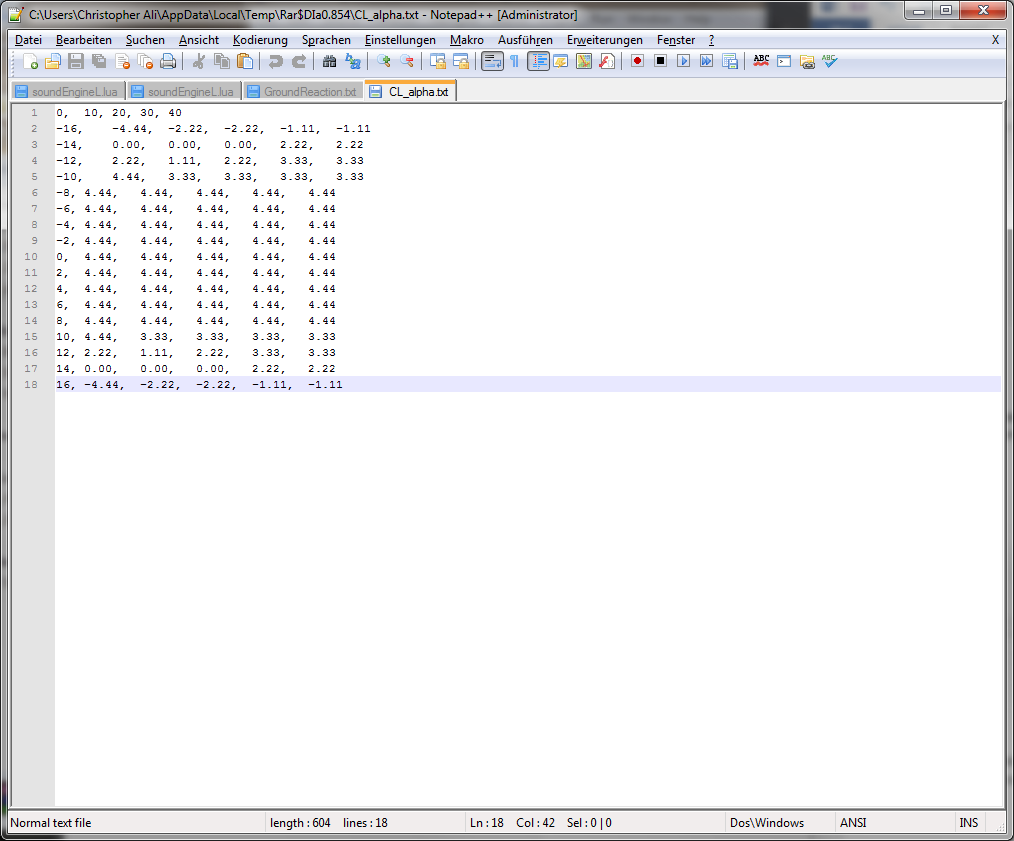
The root directory of Java Flight Simulator should contain the following:

* **./Aircraft/** Contains all aircraft for the simulation to use, each in their own folder. For more information about the contents of each aircraft folder, see
* *.***/logs/** Contains generated log files that output program states and errors during run time, useful for debugging
* *.***/natives/** Contains compiled libraries required for running Windows, Linux and OSX
* **./SimConfig/** Contains all configuration files for the simulation.
  + *AnalysisControls.json*
    - When running in Analysis Mode, the deflections (singlet, doublet, etc) listed here will automatically be input at the appropriate time
  + *ControlsConfiguration.json*
    - Contains all keyboard and joystick configurations (key, hat switch, button, axis, sensitivity, etc)
  + *PlotConfiguration.json*
    - Configures the kinds of plots (windows, subplots, axes, data sets, titles etc) displayed when the Generate Plots Option is selected
  + *SimulationConfiguration.json* 
    - Core settings for the simulation, such as using a joystick, displaying raw data results, running in Analysis Mode
    - Volume settings for various types of sounds (engine, systems, etc)
    - Display settings for the out-the-window display, such as resolution and anti-aliasing
    - Initial heading, airspeed, position and altitude values for the simulation
    - Initial flight control deflections for the simulation Settings for the numerical integration, which drives the simulation
* **./Resources/** Contains models, textures and audio for the out-the-window display for the simulation
  + **Audio/** WAV Audio files for the simulation (engine, systems, environment, etc)
  + **Entities/** OBJ Modelsand PNG texture files for model entities (trees etc)
  + **Fonts/** FNTFont files for text displayed on screen
  + **Gauges/** PNG texture files that make up the gauges on the instrument panel
  + **Particles/** PNG texture files used for particle effects (clouds, smoke etc)
  + **Terrain/** PNG texture files and blend maps used to texture the terrain
  + **Water/** PNG texture for dynamic water
* *Documentation.docx*
* *RunJavaFlightSimulator.bat (.sh)* Main batch/shell script that is used to start Java Flight Simulator
* *javaflightsimulator.jar* Compiled JAR file with all Java libraries and code needed to run Java Flight Simulator
* *license.txt* GPL V3 License

## Custom Aircraft

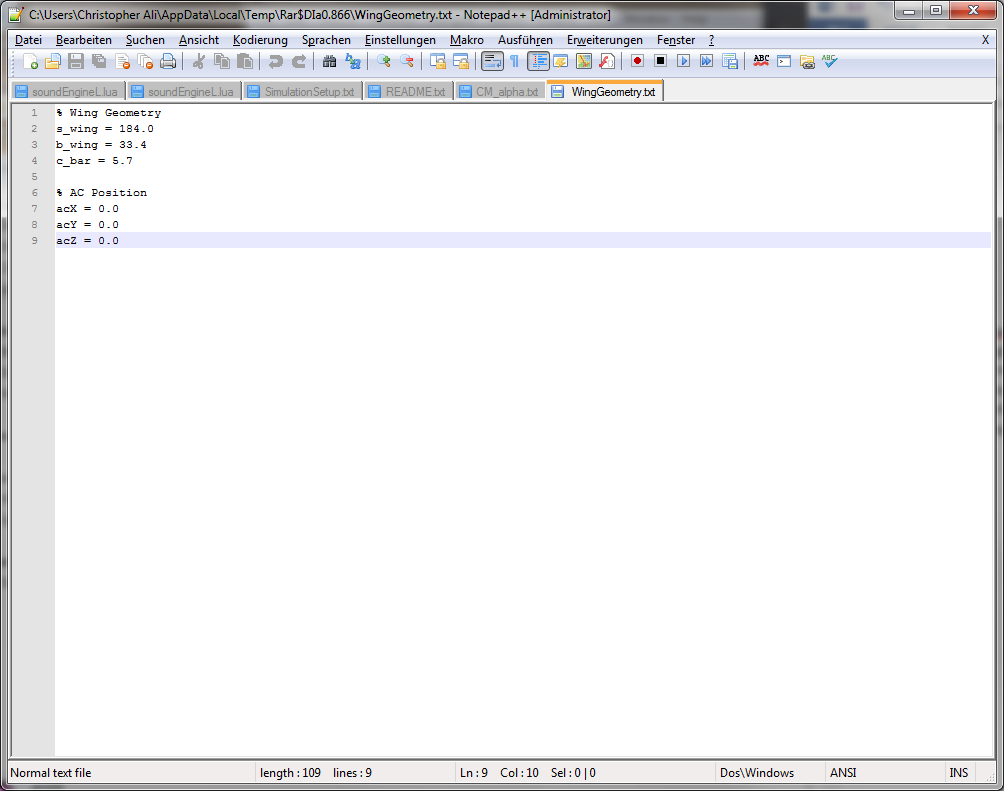
Inside of each aircraft folder in **./Aircraft/** is the following folder structure:

* **./LookupTables/** Several stability derivatives are nonlinear with respect to variables such as angle of attack and flap deflection. These derivatives are best defined with a lookup table, an array of values, where a value of this array is interpolated using breakpoints. An example of a lookup table for this application:

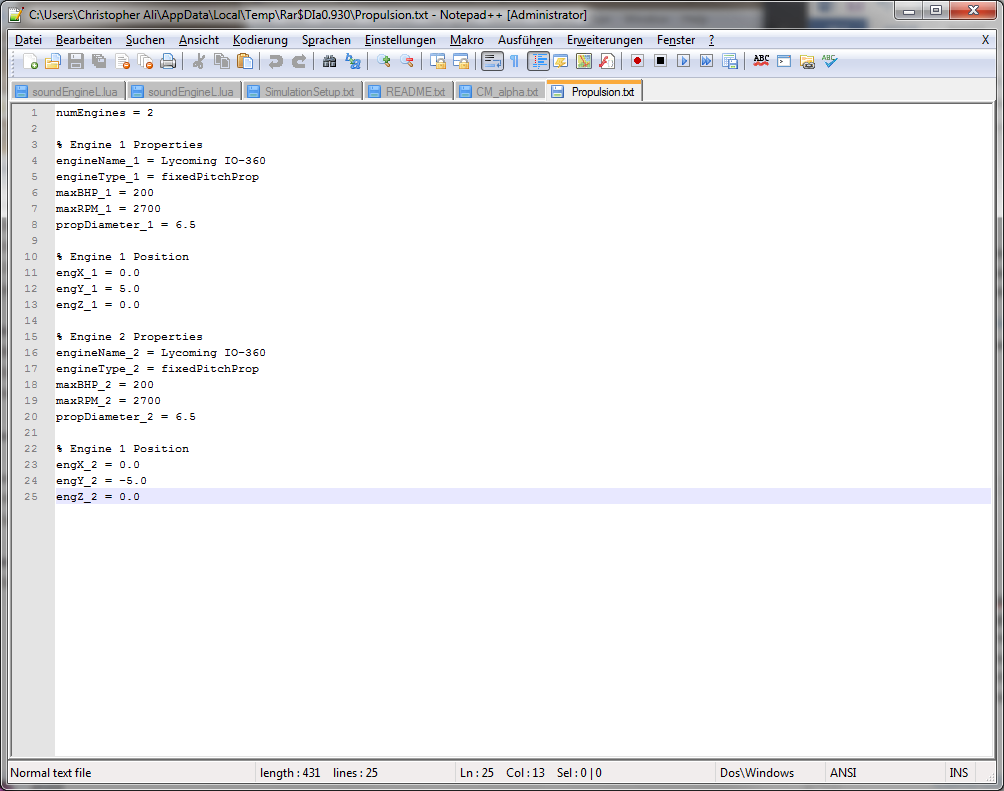


The break points here are the first row (flaps [deg]) and first column (angle of attack [deg]) of the array. The other values are potential values of CL\_alpha for a given angle of attack and flap deflection. All values in the lookup table are comma-tab separated.

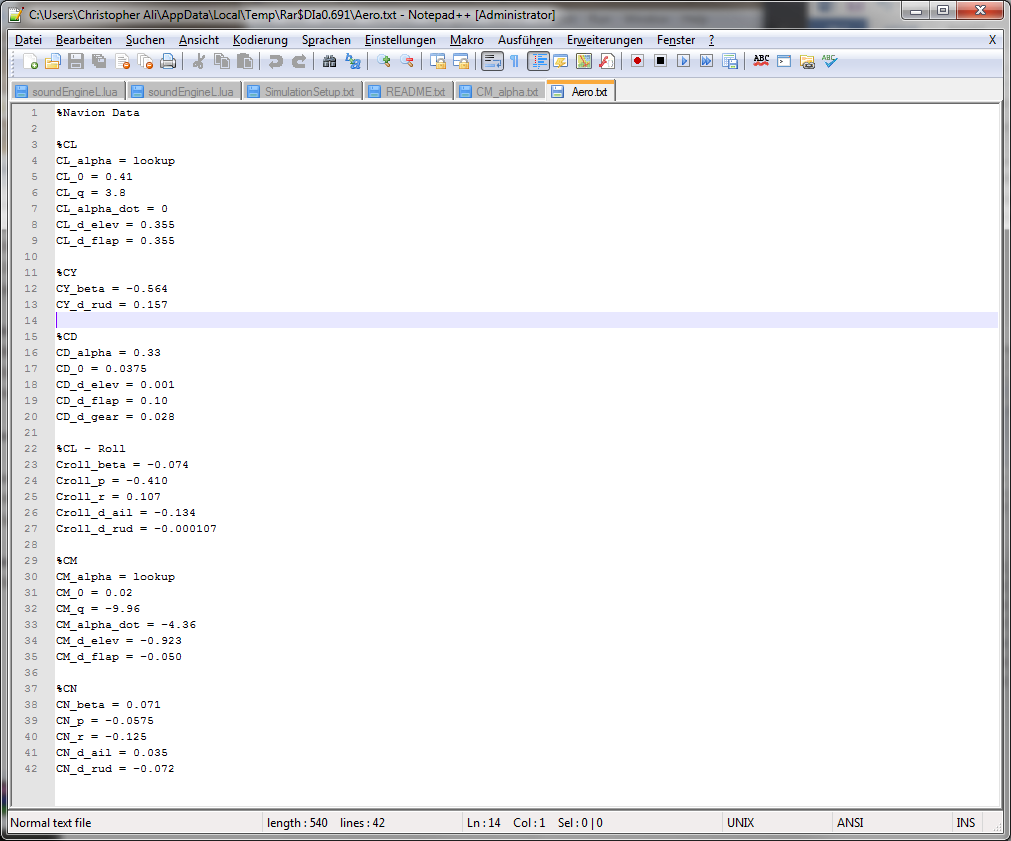
* *WingGeometry.json* Defines the aircraft’s wing span [ft], mean aerodynamic chord [ft] wing surface area [ft2] and aerodynamic center position relative to the center of gravity



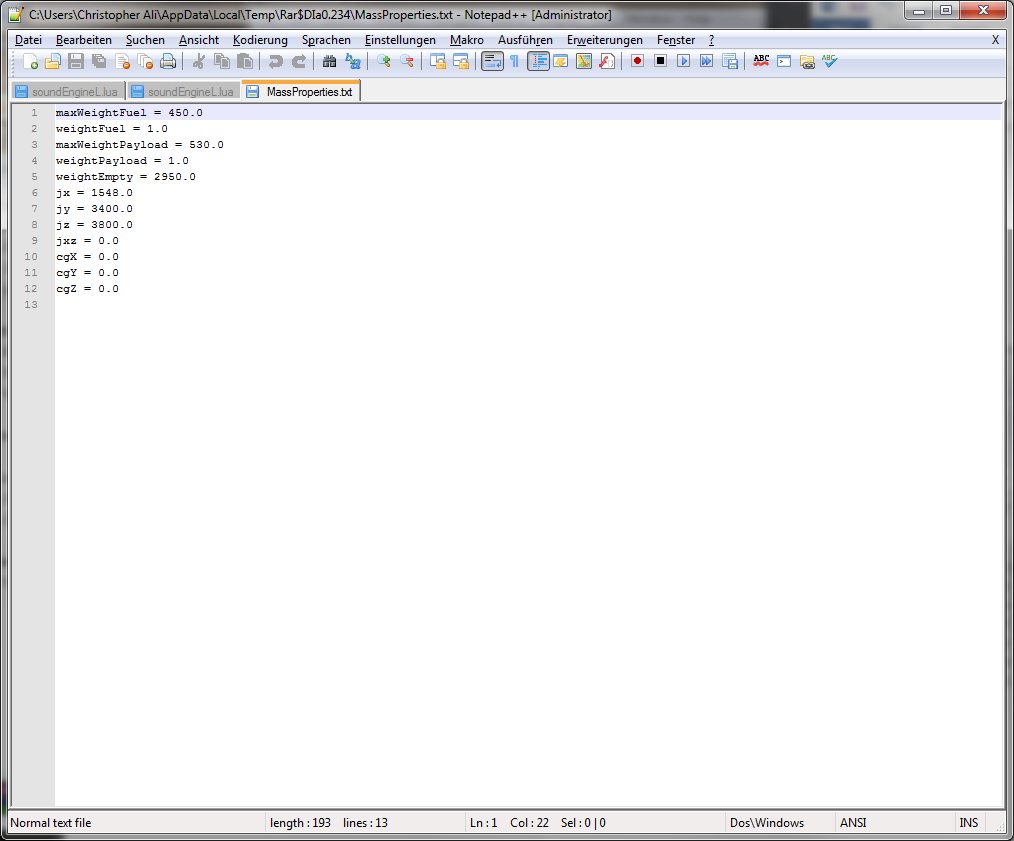
* *Propulsion.json* Defines the properties of each engine on the aircraft:



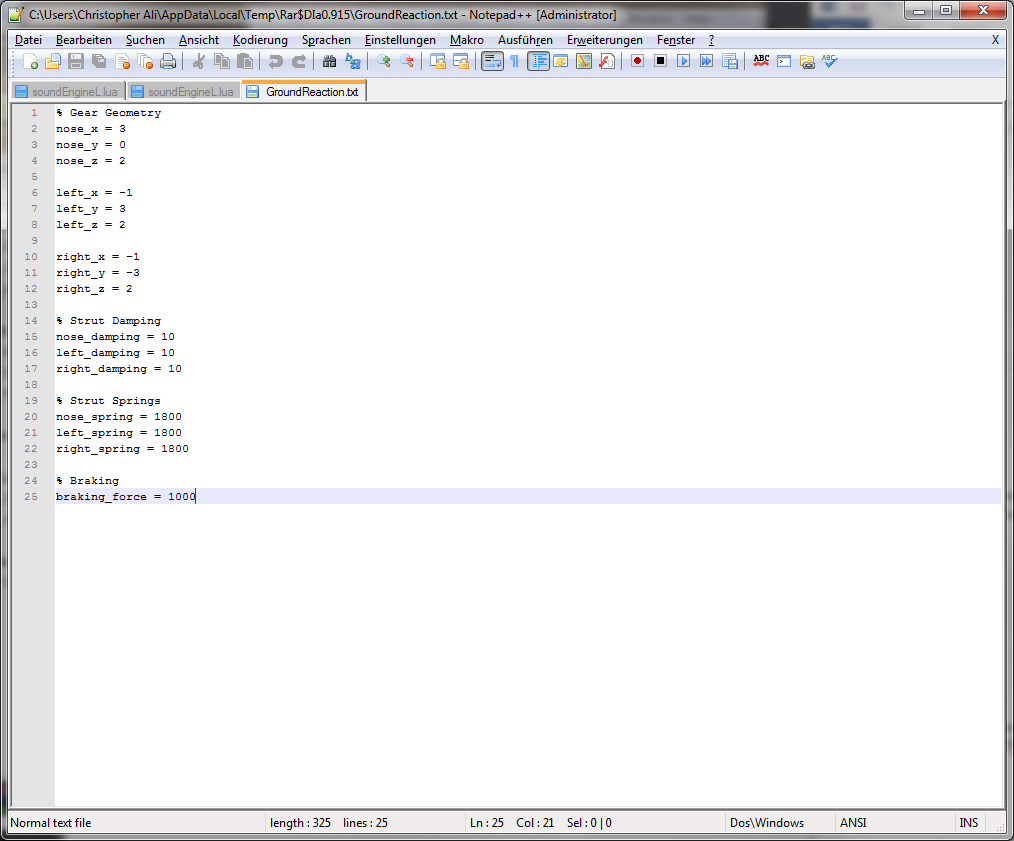
* *Aero.json* Contains stability derivative values for the aircraft [1/rad]. Any derivative with a value of *lookup* must have a lookup table text file of the same name in **./LookupTables/**



* *MassProperties.json* Contains weight [lbf], center of gravity [ft] and inertia [slug/ft3] values for the aircraft



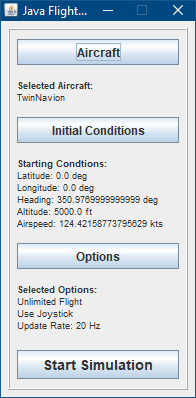
* *GroundReaction.json* Contains landing gear position relative to center of gravity [ft], spring [lbf/ft], damping [lbf/ft/sec] and braking data



* *Description.txt* Short description of the aircraft, displayed in the Aircraft menu when selected
* *PreviewPicture.jpg* Picture of the aircraft displayed in the Aircraft menu when selected. The size of the image must be roughly 430 x 230 pixels

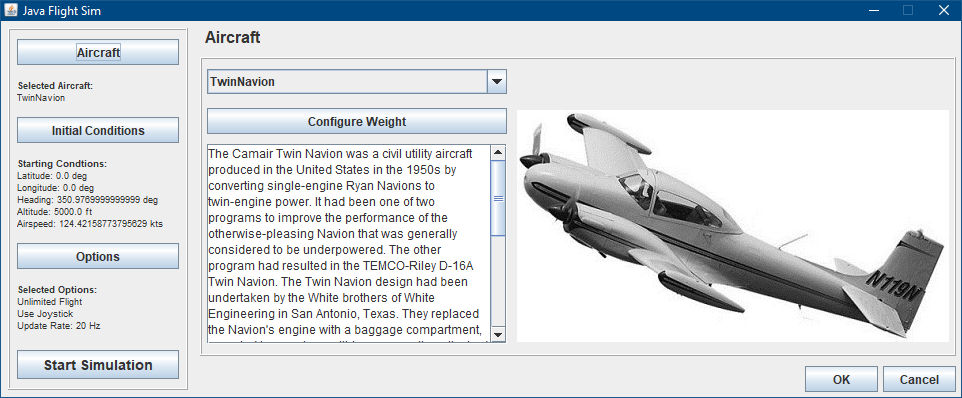
# Menus

Upon starting Java Flight Simulator, the following main menu will appear:

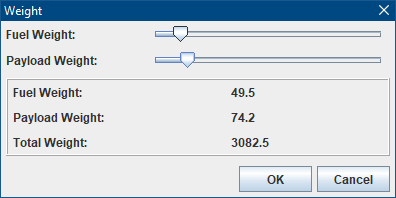


The main menu of Java Flight Simulator contains buttons to select an aircraft, initial conditions, configure simulation options and start the simulation.

## Aircraft

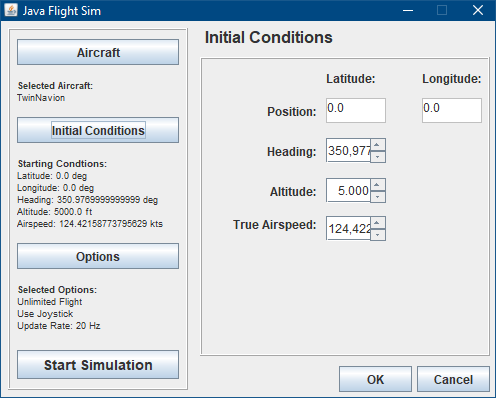


Select an aircraft from the available aircraft the dropdown window. The weight of fuel and payload can be configured by selecting **Configure Weight**:



The fuel weight and payload weights are adjusted by moving the sliders, which then recalculates the total weight for the aircraft.

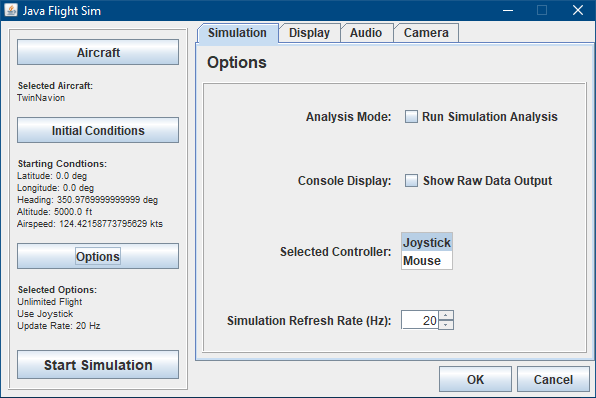
## Initial Conditions



Initial values for the aircraft’s position [deg], true airspeed [kts], altitude [ft] and heading [deg] are specified here.

## Options

### Simulation



Configures core simulation settings:

*Analysis Mode* specifies whether to run a flight dynamics analysis for the selected aircraft, or fly the aircraft in a real time pilot-in-the-loop simulation.

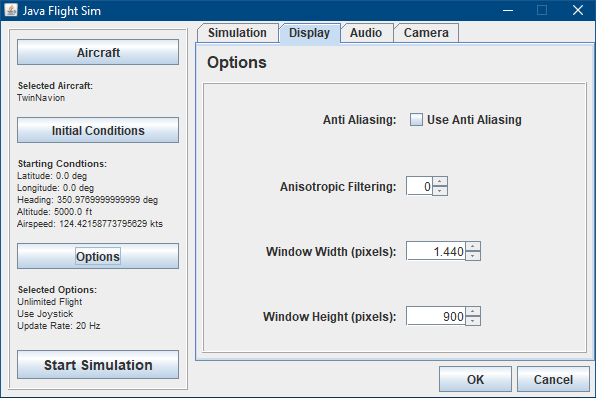
*Console Display* opens a window showing a table of raw simulation output data at each simulation step.

*Selected Controller* specifies which type of controller to use with the real time simulation. If analysis mode is selected, this window will be unavailable.

*Instrument Panel* selects whether an instrument panel should be displayed when not running in analysis mode.

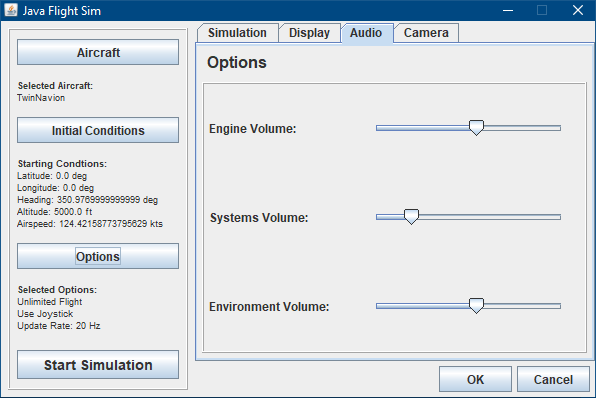
*Simulation Refresh Rate (Hz)* controls how many times per second the simulation recalculates. The larger the value, the smoother the simulation will run, at the cost of processing power and memory.

### Display



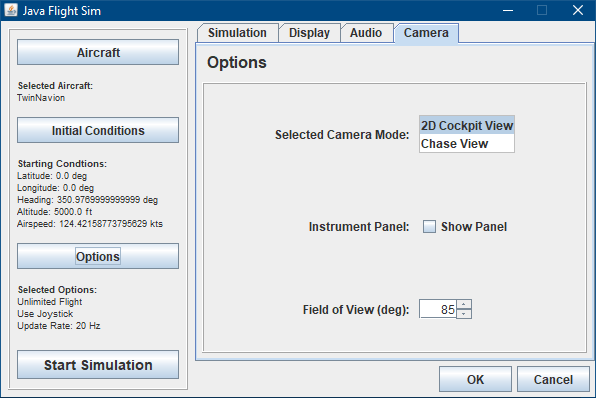
Controls the window size, field of view (FOV) in degrees, and anti-aliasing (2x) settings for the out the window display

### Audio



Sets the volume of engine, systems (flaps, gear, stall horn etc) and environment (wind) sounds

### Camera



Sets the

# Controls and Hotkeys

When running as a real-time simulation, keyboard hotkeys are always available regardless of the controller selected. All mappings mentioned below are all configurable in *ControlsConfiguration.json* located in .**/SimConfig/ .** The default setup is:

* L – Opens the plotting window, where the last 100 seconds of flight data is available to view
* P – Pauses the simulation when the cockpit window is selected
* R – Resets the simulation back to initial conditions when the simulation is paused
* Q – Exits the simulation and returns to the main menu
* Up/Down Arrow – Down/Up elevator
* Left/Right Arrow – Left/Right aileron
* Page Up/Down – Increase/Decrease throttle
* F6/F7 – Increase Decrease flaps

If *Mouse* is the selected controller in *Options*, the following controls are used by default:

* Mouse Y Axis – Elevator
* Mouse X Axis – Aileron
* Mouse Wheel Up/Down – Increase/Decrease throttle

If *Joystick* is the selected controller in *Options*, the following default controls are used, although they can be changed in *ControlsConfiguration.json located in .***/SimConfig/**. For all button commands, please verify the button numbers with your joystick, as different joysticks can have different button configurations:

* Y Axis – Elevator
* X Axis – Aileron
* Rotation – Rudder
* Slider – Throttle
* POV Hat Up/Down – Down/Up elevator trim
* Button 0 – Brakes
* Button 4 – Gear down
* Button 5 – Gear up
* Button 6 – Decrease flaps
* Button 7 – Increase flaps

The following axes and buttons are used from the rudder, yoke and throttle quadrant if CH Products are used. The yoke, rudder and brake axes are assigned as expected for typical flight simulation controls and therefore omitted from the image:



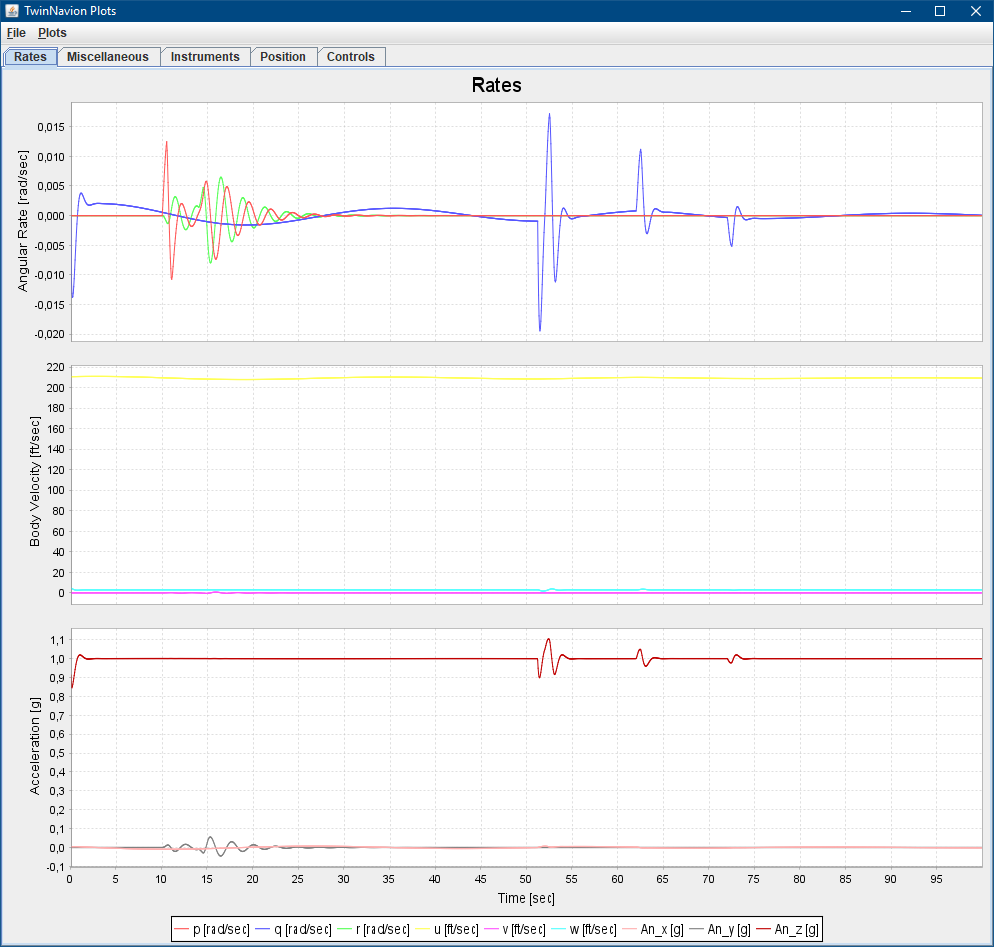
# Simulation Modes

## Analysis Mode

Runs a flight dynamics analysis of the selected aircraft, where a series of aileron, rudder and elevator control doublets are generated to incite dynamic longitudinal and lateral modes of the aircraft. All human input controls are disabled in this mode, and no out-the-window display is shown. If *Console Display* is selected from *Options*, the raw data console will open as well.

Java Flight Simulator uses JFreeChart to plot flight data. It is capable of zooming, printing, saving and customizing each plot displayed. For more information on JFreeChart’s capabilities, please visit <http://www.jfree.org/jfreechart/> . The tabs, plots and subplots can be customized by modifying *PlotConfiguration.json* in .**/SimConfig/** . The following figures show the default views available in the plotting window, and the data that they display:

### Rates



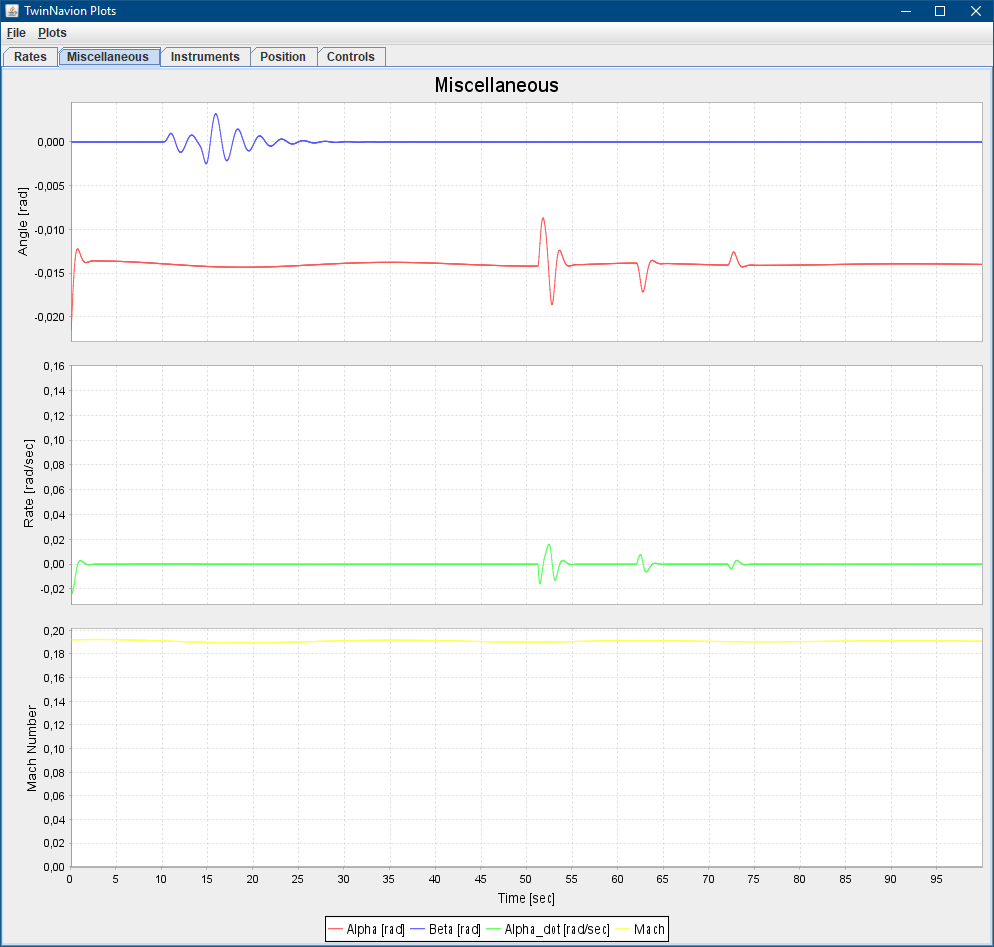
From top to bottom:

* Angular velocities *p* (roll rate [rad/sec]), *q* (pitch rate [rad/sec]) and *r* (yaw rate [rad/sec])
* Linear velocities *u* (forward velocity [ft/sec]), *v* (side velocity [ft/sec]) and *w* (vertical velocity [ft/sec])
* Linear accelerations in the X, Y and Z direction [g]

From the Plots dropdown menu:

* *Refresh* (Ctrl-R) refreshes all plots in the window, updating them with the most recent data from the simulation
* *Clear Plots* (Ctrl-E) clears all plot data. This is useful for preparing readying the plots for a new maneuver. Refreshing with *Refresh* will then redraw all plots with new data since the deletion

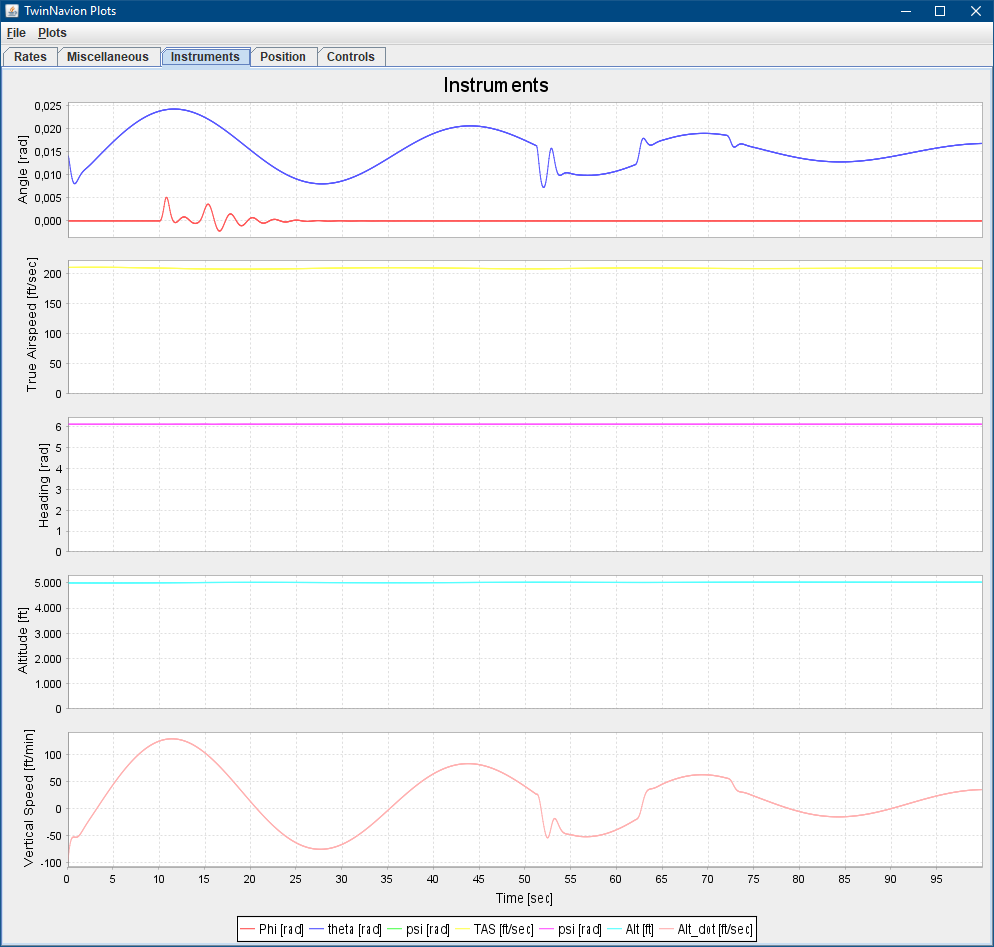
### Miscellaneous



From top to bottom:

* Angle of attack (alpha) [rad] and angle of sideslip (beta) [rad]
* Change of angle of attack with time (alpha dot) [rad/sec]
* Mach number

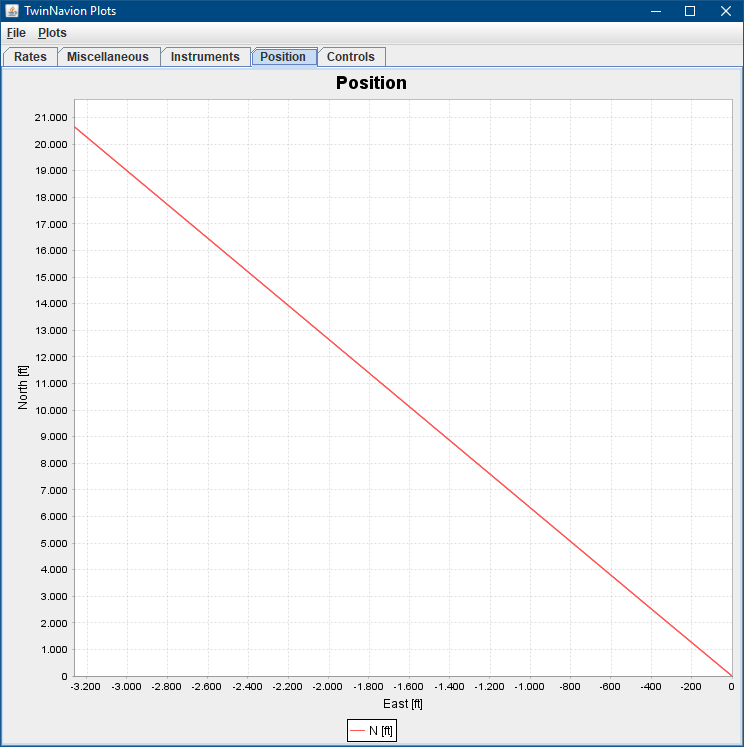
### Instruments



From top to bottom:

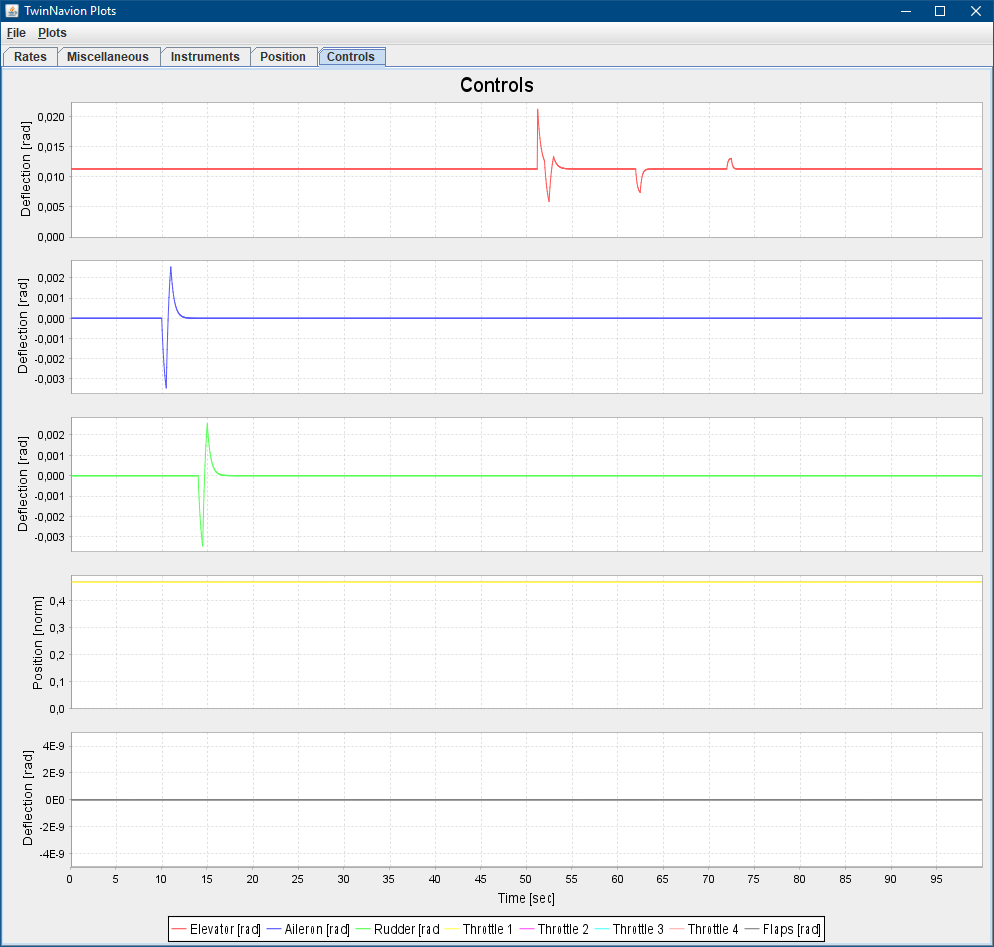
* Roll (phi [rad]) and pitch (theta[rad])
* True airspeed [ft/sec]
* Heading (psi) [rad]
* Altitude [ft]
* Vertical speed (alt dot [ft/min])

### Position



Aircraft’s north and east position [ft]

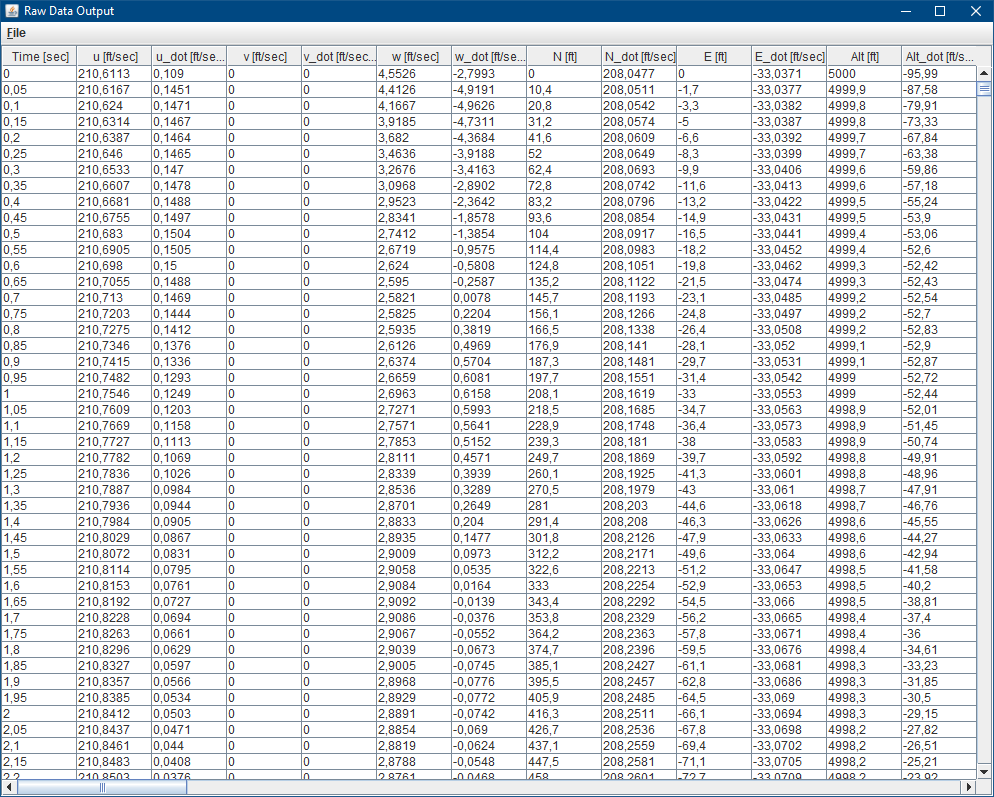
### Controls



From top to bottom:

* Elevator position[rad]
* Aileron position [rad]
* Rudder position [rad]
* Throttle position [norm]
* Flap position [rad]

### Raw Data Output



When *Console Display* is selected in *Options* the above window opens, showing a tabulated view of raw data from the simulation for each step of the simulation. Columns can be shifted as desired, and copied to the clipboard using Ctrl+C. The entire data set can be exported as a CSV file by selecting Export as CSV from the File dropdown menu.

## Real-time (Normal) Mode

Starts a real-time, pilot-in-the-loop simulation of the selected aircraft. Opens an out-the-window display. If *Instrument Panel* is selected from *Options*, an instrument panel is also rendered, containing a simple six-pack of instruments, a multi-engine tachometer and indicators of landing gear and flap position. If not, a text display of relevant flight information (airspeed, altitude, heading, position and G-force) will render instead. If *Console Display* is selected from *Options*, the raw data console will open as well.





# References and Resources

* *Small Unmanned Aircraft: Theory and Practice by Beard, R.W. and McLain, T.W.*
* *Principles of Flight Simulation, Allerton David*
* *Instrument panel gauges were created using Gerrit Grunwald's SteelSeries Swing port:* [*https://github.com/HanSolo/SteelSeries-Swing*](https://github.com/HanSolo/SteelSeries-Swing)
* *jinput is used for joystick, mouse and keyboard support:* [*https://github.com/jinput/jinput*](https://github.com/jinput/jinput)
* *LWJGL2 (2.9.3) (Lightweight Java Game Library) is used to create the out the window display using OpenGL and audio engine in OpenAL:* [*https://www.lwjgl.org/*](https://www.lwjgl.org/)
* *I do not own any rights to textures, models and sounds used in Java Flight Simulator and relinquish all rights to those resources*