A non-linear simulator written in C for orbital spacecraft rendezvous applications.

Paulo Ricardo Arantes Gilz

A non-linear simulator written in C for orbital spacecraft rendezvous applications

Paulo R. Arantes Gilz
LAAS-CNRS
7, av. du Colonel Roche,
31077 Toulouse, France,
prarante@laas.fr

Contents

1 Brief introduction

This document details the models and equations (sec. 2) used in the core of a non-linear simulator written in C for orbital spacecraft rendezvous applications. It additionally explains the relation between the C functions and the mathematical models (sec. 4) and presents a practical example of how to simulate a given rendezvous scenario (sec. 3).

The simulator here discussed is a modified version of the Matlab®/Simulink® simulator developed by Mounir Kara-Zaitri during his PhD thesis at the LAAS-CNRS. The implemented modifications were performed in order to obtain a better-suited tool for simulating and developing control algorithms for the orbital spacecraft rendezvous without using the Matlab® environment.

For a given orbital rendezvous scenario, the output of the simulator is the evolution of the relative position and velocity between the two spacecrafts, obtained by simulating the evolution of their trajectories via the integration of the Gauss equations for the orbital motion under the disturbances provoked by the Earth's flatness (the J_2-effect) and the atmospheric drag.

2 Models and equations

Hereafter we introduce some variables, state representations and coordinate system changes that will be further employed in the equations modelling the evolution of the spacecrafts trajectories and the disturbances provoked by the atmospheric drag and the Earth's flatness.

Remark: the following references are strongly recommended to help the reader to understand the nomenclature and notations used in the sequel:

- Mounir's PhD thesis: `http://thesesups.ups-tlse.fr/1026/1/Kara%2DZaitri_Mounir.pdf`

- Gerogia's PhD thesis: `http://thesesups.ups-tlse.fr/2105/1/2013TOU30170.pdf`

2.1 State representations

- The classical orbital elements state:

$$X_{OE_c} = [a,\ e,\ i,\ \Omega,\ \omega,\ \nu]^t \tag{2.1}$$

  where:

  - a is the semi-major axis of the orbit;
  - e is the eccentricity the orbit;
  - i is the inclination;
  - Ω is the longitude of ascending node;
  - ω is the argument of perigee;

– ν is the true anomaly;

- The equinoctial orbital elements state:

$$X_{OE_{eq}} = [p,\ f,\ g,\ h,\ k,\ L]^t \tag{2.2}$$

- The inertial state:

$$X_{\mathrm{I}} = [x_{\mathrm{I}},\ y_{\mathrm{I}},\ z_{\mathrm{I}},\ \dot{x}_{\mathrm{I}},\ \dot{y}_{\mathrm{I}},\ \dot{z}_{\mathrm{I}}]^t \tag{2.3}$$

representing the position and the velocity in the Earth's inertial frame.

- The LVLH state:

$$X_{\mathrm{LVLH}} = [x_{\mathrm{LVLH}},\ y_{\mathrm{LVLH}},\ z_{\mathrm{LVLH}},\ \dot{x}_{\mathrm{LVLH}},\ \dot{y}_{\mathrm{LVLH}},\ \dot{z}_{\mathrm{LVLH}}]^t \tag{2.4}$$

representing the position and the velocity in the LVLH frame.

- The RSW state:

$$X_{\mathrm{RSW}} = [x_{\mathrm{RSW}},\ y_{\mathrm{RSW}},\ z_{\mathrm{RSW}},\ \dot{x}_{\mathrm{RSW}},\ \dot{y}_{\mathrm{RSW}},\ \dot{z}_{\mathrm{RSW}}]^t \tag{2.5}$$

representing the position and the velocity in the RSW frame.

2.2 Coordinate system changes

2.2.1 Classical orbital elements state to equinoctial orbital elements state

$$\begin{array}{rcl} p & = & a(1-e^2) \\ f & = & e\cos(\Omega+\omega) \\ g & = & e\sin(\Omega+\omega) \\ h & = & \tan(i/2)\cos\Omega \\ k & = & \tan(i/2)\sin\Omega \\ L & = & \Omega+\omega+\nu \end{array} \tag{2.6}$$

2.2.2 Equinoctial orbital elements state to inertial state

$$\begin{array}{rcl} x_I & = & \frac{r}{s^2}(\cos L+\alpha^2\cos L+2hk\sin L) \\ y_I & = & \frac{r}{s^2}(\sin L-\alpha^2\sin L+2hk\cos L) \\ z_I & = & \frac{2r}{s^2}(h\sin L-k\cos L) \\ \dot{x}_I & = & -\frac{1}{s^2}\sqrt{\frac{\mu}{p}}(\sin L+\alpha^2\sin L-2hk\cos L+g-2fhk+\alpha^2 g) \\ \dot{y}_I & = & -\frac{1}{s^2}\sqrt{\frac{\mu}{p}}(-\cos L+\alpha^2\cos L-2hk\sin L-f+2ghk+\alpha^2 f) \\ \dot{z}_I & = & \frac{2}{s^2}\sqrt{\frac{\mu}{p}}(h\cos L+k\sin L+fh+gk) \end{array} \tag{2.7}$$

where

$$\begin{aligned} \mu &: \text{Earth's gravitational constant} \\ s^2 &= 1 + h^2 + k^2 \\ w &= 1 + f\cos L + g\sin L \\ r &= p/w \\ \alpha^2 &= h^2 - k^2 \end{aligned} \tag{2.8}$$

2.2.3 RSW state to LVLH state / LVLH state to RSW state

$$\begin{aligned} x_{\text{LVLH}} &= y_{\text{RSW}} \\ y_{\text{LVLH}} &= z_{\text{RSW}} \\ z_{\text{LVLH}} &= -x_{\text{RSW}} \end{aligned} \quad \text{and} \quad \begin{aligned} \dot{x}_{\text{LVLH}} &= \dot{y}_{\text{RSW}} \\ \dot{y}_{\text{LVLH}} &= \dot{z}_{\text{RSW}} \\ \dot{z}_{\text{LVLH}} &= -\dot{x}_{\text{RSW}} \end{aligned} \tag{2.9}$$

2.2.4 Computing the RSW basis in function of the inertial state

The following computations must be performed in order to express the RSW orthonormal basis $(\vec{R}, \vec{S}, \vec{W})$ in the Earth's inertial frame in function of the inertial state $X_{\text{I}} = [x_{\text{I}},\ y_{\text{I}},\ z_{\text{I}},\ \dot{x}_{\text{I}},\ \dot{y}_{\text{I}},\ \dot{z}_{\text{I}}]^t$:

$$\begin{aligned} \vec{R} &= \frac{1}{\sqrt{x_{\text{I}}^2+y_{\text{I}}^2+z_{\text{I}}^2}}(x_{\text{I}}, y_{\text{I}}, z_{\text{I}}) \\ \vec{W} &= \frac{1}{\|\vec{U}\|}\vec{U} \\ \vec{S} &= \frac{1}{\|\vec{V}\|}\vec{V} \end{aligned} \quad \text{where} \quad \begin{aligned} \vec{T} &= \frac{1}{\sqrt{\dot{x}_{\text{I}}^2+\dot{y}_{\text{I}}^2+\dot{z}_{\text{I}}^2}}(\dot{x}_{\text{I}}, \dot{y}_{\text{I}}, \dot{z}_{\text{I}}) \\ \vec{U} &= -\vec{T} \times \vec{R} \\ \vec{V} &= \vec{U} \times \vec{R} \end{aligned} \tag{2.10}$$

2.2.5 Computing the relative LVLH state

Hereafter we show how to compute the relative state between two spacecrafts in the LVLH frame of the leader spacecraft ($\bar{X}^l_{\text{LVLH}}$) in function of the inertial states of each spacecraft (X^f_{I} and X^l_{I}) and the equinoctial orbital elements of the leader spacecraft ($X^l_{OE_{eq}}$):

First we define the inertial relative state $\bar{X}_{\text{I}} = [\bar{x}_{\text{I}},\ \bar{y}_{\text{I}},\ \bar{z}_{\text{I}},\ \dot{\bar{x}}_{\text{I}},\ \dot{\bar{y}}_{\text{I}},\ \dot{\bar{z}}_{\text{I}}]^t$ as:

$$\bar{X}_{\text{I}} = X^f_{\text{I}} - X^l_{\text{I}} \tag{2.11}$$

and the inertial relative position and velocity vectors $\vec{p}_{\text{I}}$ and $\vec{v}_{\text{I}}$:

$$\vec{p}_{\text{I}} = (\bar{x}_{\text{I}},\ \bar{y}_{\text{I}},\ \bar{z}_{\text{I}}) \quad \text{and} \quad \vec{v}_{\text{I}} = (\dot{\bar{x}}_{\text{I}},\ \dot{\bar{y}}_{\text{I}},\ \dot{\bar{z}}_{\text{I}}) \tag{2.12}$$

After that, the RSW orthonormal base of the leader spacecraft $(\vec{R}^l, \vec{S}^l, \vec{W}^l)$ is computed in function of the inertial state of the leader satellite (X^l_{I}) via (2.10). The relative state in the

LVLH frame of the leader spacecraft is then computed as follows:

$$\bar{X}^l_{\text{LVLH}} = \begin{bmatrix} \bar{x}^l_{\text{LVLH}} \\ \bar{y}^l_{\text{LVLH}} \\ \bar{z}^l_{\text{LVLH}} \\ \dot{\bar{x}}^l_{\text{LVLH}} \\ \dot{\bar{y}}^l_{\text{LVLH}} \\ \dot{\bar{z}}^l_{\text{LVLH}} \end{bmatrix} = \begin{bmatrix} \langle \vec{S}^l, \vec{p}_{\text{I}} \rangle \\ \langle \vec{W}^l, \vec{p}_{\text{I}} \rangle \\ -\langle \vec{R}^l, \vec{p}_{\text{I}} \rangle \\ \langle \vec{S}^l, \vec{v}_{\text{I}} \rangle - \dot{\nu}^l \langle \vec{R}^l, \vec{p}_{\text{I}} \rangle \\ \langle \vec{W}^l, \vec{v}_{\text{I}} \rangle + \\ -\langle \vec{R}^l, \vec{v}_{\text{I}} \rangle - \dot{\nu}^l \langle \vec{S}^l, \vec{p}_{\text{I}} \rangle \end{bmatrix} \tag{2.13}$$

where $\langle \cdot, \cdot \rangle$ is the dot product in $\mathbb{R}^3$ and

$$\dot{\nu}^l = \sqrt{\mu p^l} \left(\frac{1 + f^l \cos L^l + g^l \sin L^l}{p^l} \right)^2 \tag{2.14}$$

Remark: the terms $-\dot{\nu}^l \langle \vec{R}^l, \vec{p}_{\text{I}} \rangle$ and $-\dot{\nu}^l \langle \vec{S}^l, \vec{p}_{\text{I}} \rangle$ are due to the relative velocity between frames.

2.3 Dynamical equations

The dynamics of the spacecrafts movement is modelled by a non-classical form of the Gauss planetary equations. We choose this different representation because the classical one presents singularities for low eccentricity (e) and inclination (i) values. However, by rewriting these equations in function of the equinoctial orbital elements, these singularities are eliminated.

Given that, the dynamics of the equinoctial orbital elements is given by:

$$\frac{dX_{OE_c}}{dt} = A\vec{u} + B \tag{2.15}$$

where $\vec{u} = \vec{u}_c + \vec{u}_{J_2} + \vec{u}_d$ is the combined acceleration induced by a possible control law and the disturbances (J_2-effect and atmospheric drag) expressed in the RSW coordinate system and:

$$A = \sqrt{\frac{p}{\mu}} \begin{bmatrix} 0 & \frac{2p}{w} & 0 \\ \sin L & \frac{(w+1)\cos L + f}{w} & -\frac{g(h \sin L - k \cos L)}{w} \\ -\cos L & \frac{(w+1)\sin L + g}{w} & \frac{f(h \sin L - k \cos L)}{w} \\ 0 & 0 & \frac{s^2 \cos L}{2w} \\ 0 & 0 & \frac{s^2 \sin L}{2w} \\ 0 & 0 & \frac{h \sin L - k \cos L}{w} \end{bmatrix} \tag{2.16}$$

$$B = \left[0,\ 0,\ 0,\ 0,\ 0,\ \frac{w^2}{p^2}\sqrt{\mu p}\ \right]^t \tag{2.17}$$

2.4 J_2 disturbance

The effect of the Earth's flatness is modelled as a external disturbance that provokes an acceleration on the satellite. This acceleration is given in the RSW coordinates system by:

$$\vec{u}_{J_2} = -\frac{3\mu J_2 R_e^2}{2r^4}\begin{bmatrix} 1 - \dfrac{12(h\sin L - k\cos L)^2}{(1+h^2+k^2)^2} \\ \dfrac{8(h\sin L - k\cos L)(h\cos L + k\sin L)}{(1+h^2+k^2)^2} \\ \dfrac{4(h\sin L - k\cos L)(1-h^2-k^2)}{(1+h^2+k^2)^2} \end{bmatrix} \tag{2.18}$$

where J_2 is the second degree term in Earth's gravity potential and R_e is the Earth's radius.

2.5 Atmospheric drag disturbance

The effect of the atmospheric drag is also modelled as a external disturbance that provokes an acceleration on the satellite. This acceleration is given in the RSW coordinates system by:

$$\vec{u}_d = -\frac{\rho S C_d}{2m}\frac{\mu}{p}\sqrt{1 + 2(g\sin L + f\cos L) + f^2 + g^2}\begin{bmatrix} f\sin L - g\cos L \\ 1 + f\cos L + g\sin L \\ 0 \end{bmatrix} \tag{2.19}$$

where ρ is the atmospheric density and m, S and C_d are respectively the mass, the cross sectional area and the drag coefficient of the spacecraft. In this equation, ρ is not a constant, and the model used for it is discussed in the next subsection.

2.6 Atmospheric density model

The atmospheric density model initially used by Mounir in the first version of the simulator was given in function of the distance between the satellite and the center of the Earth by the following equation:

$$\rho(x_{\text{I}}, y_{\text{I}}, z_{\text{I}}) = \rho_{LH}\ \exp\left(\frac{R_e + 400000 - \sqrt{x_{\text{I}}^2 + y_{\text{I}}^2 + z_{\text{I}}^2})}{46830}\right) \tag{2.20}$$

where ρ_{LH} is a constant that depends of the solar activity (2.2644×10^{-12} for low and 3.5475×10^{-11} for high solar activity). The author himself admitted in his work that this model was an

approximation and should be modified in order to obtain a better verisimilitude.

The main problem noticed by using this model during simulations is that as the distance between the satellite and the Earth decreases, ρ grows exponentially and the intensity of the drag disturbance increase too much.

In order to avoid this behaviour, the new simulator uses the following equation instead:

$$\rho(x_{\text{I}}, y_{\text{I}}, z_{\text{I}}) = \min \left\{ \rho_{LH} \, \exp \left(\frac{R_e + 400000 - \sqrt{x_{\text{I}}^2 + y_{\text{I}}^2 + z_{\text{I}}^2)}}{46830} \right), \; \rho_{LH} \right\} \tag{2.21}$$

2.7 Simulation parameters

The integration of the differential equation (2.15) is performed by a simple first-order Euler scheme with fixed-step. In this context, the user only need to inform the initial and final times and the integration step for the execution of the following loop:

$t^* = t_0$

While $(t^* < t_f)$ do

$$X_{OE_{eq}}(t^* + \text{step}) = X_{OE_{eq}}(t^*) + (\text{step}) \left. \frac{dX_{OE_{eq}}}{dt} \right|_{t^*}$$

$t^* = t^* + \text{step}$

Loop

3 How to use it?

The file `Simulator.zip` contains five files that must be extracted to a same directory:

- `Makefile`, the file containing the script for the compilation routines;
- `main.c`, the main C file;
- `library.c`, the C file containing the functions;
- `inputs`, the file where the user inputs the rendezvous scenario parameters;
- `plot_trajectory.m`, an auxiliary function to plot the obtained results in Matlab®.

In order to run the simulator, the user has to compile the C code to obtain the executable, define the scenario parameters and run the executable.

3.1 Compilation

To compile the C code, the user has to open a terminal, go to the directory where the files where extracted and run the `make` command:

```
$make
gcc -Wunused-macros -Wall -O2 main.c -o rdvz_simulator -lm
$ll
total 60
-rwxr--r-- 1 prarante mac   246 août  16 17:30 inputs*
-rwxr--r-- 1 prarante mac 14655 août  16 17:28 library.h*
-rwxr--r-- 1 prarante mac   991 août   7 16:23 main.c*
-rwxr--r-- 1 prarante mac   611 août   8 02:29 Makefile*
-rwxr--r-- 1 prarante mac   229 août  16 17:30 plot_trajectory.m*
-rwxr-xr-x 1 prarante mac 26992 août  17 17:12 rdvz_simulator*
```

The executable file `rdvz_simulator` will be created.

3.2 Defining the rendezvous scenario

The rendezvous scenario must be declared by the user in the file `inputs`. This file should contains one parameter of the simulation in each line, in the following order:

Parameter	**Description**
Initial simulation time	-
Final simulation time	-
Integration step	-
Output frequency	Defines the frequency for writing to the output file. If the value is set to ten, for example, one out of ten points will be outputted. The initial and final points are always outputted.
Earth's gravitational constant	-
Earth's radius	-
J_2 effect constant	-
Mass of leader spacecraft	-
Mass of follower spacecraft	-
Drag surface of leader spacecraft	-
Drag surface of follower spacecraft	-
Drag coefficient of the leader spacecraft	-
Drag coefficient of the follower spacecraft	-
Semi-major axis of leader orbit	-
Eccentricity of leader orbit	-
Inclination of leader orbit	-
Longitude of ascending node of leader orbit	-
Argument of perigee of leader orbit	-
True anomaly of leader orbit	-
Semi-major axis of follower orbit	-
Eccentricity of follower orbit	-
Inclination of follower orbit	-
Longitude of ascending node of follower orbit	-
Argument of perigee of follower orbit	-
True anomaly of follower orbit	-
J_2 effect	1 - yes, 0 - no
Atmospheric drag effect	1 - yes, 0 - no
Solar activity	1 - hi, 0 - low

In order to illustrate what the `inputs` should look like, we give hereafter an example:

```
0
5000
1
10
3.986004418e14
6378136.55
1.0826268361960958e-3
462949
20000
1703
50
3
2.274
7011e3
0.4
0
0
0
0
7.011000000699955e+06
4.000000000514252e-01
2.377143906190743e-06
4.712393734818199e+00
1.570796326794897e+00
6.283185307179586e+00
1
1
1
```

3.3 Running the simulation

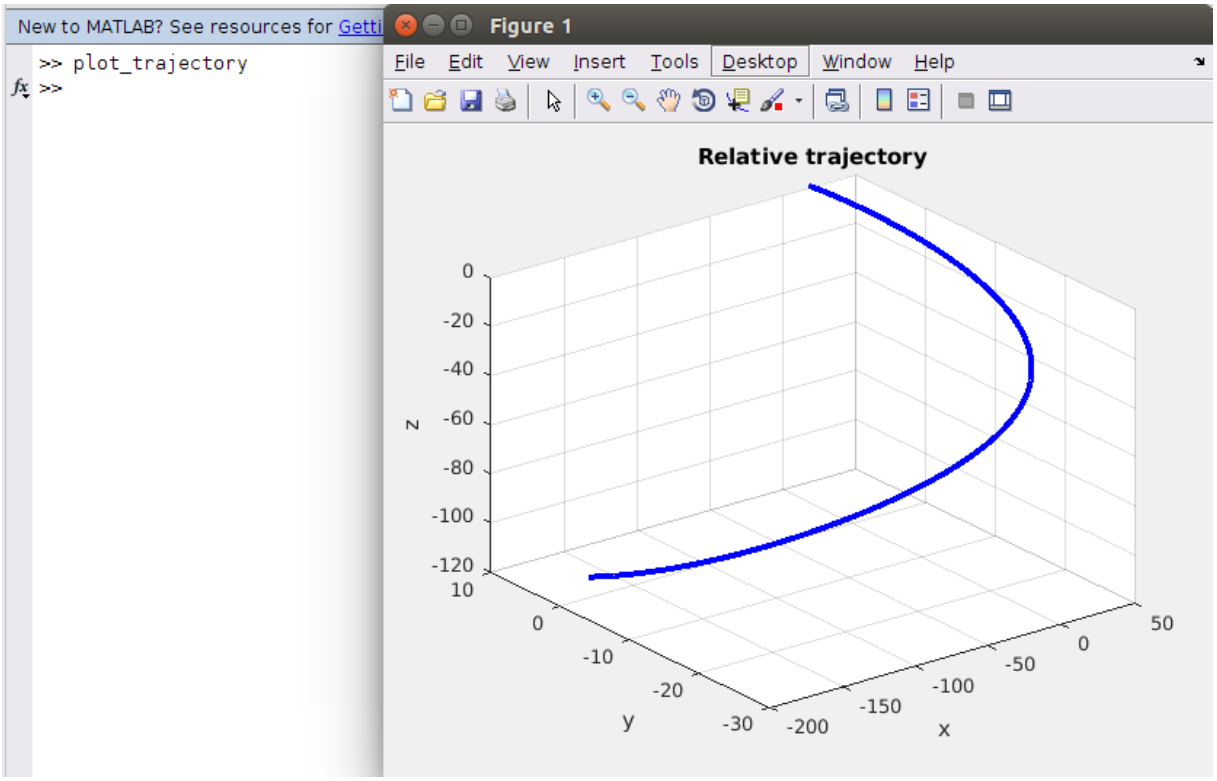
To run the simulation, the user must run the command `rdvz_simulator`:

```
$./rdvz_simulator
Elapsed time: 19306 microseconds
$ll
total 232
-rwxr--r-- 1 prarante mac    246 août  16 17:30 inputs*
-rwxr--r-- 1 prarante mac  14661 août  17 17:25 library.h*
-rwxr--r-- 1 prarante mac    991 août   7 16:23 main.c*
-rwxr--r-- 1 prarante mac    611 août   8 02:29 Makefile*
-rw-r--r-- 1 prarante mac 172383 août  17 17:54 outputs
-rwxr--r-- 1 prarante mac    229 août  16 17:30 plot_trajectory.m*
-rwxr-xr-x 1 prarante mac  26992 août  17 17:12 rdvz_simulator*
```

The time the simulation took to be performed will be displayed and the file `outputs` will be generated.

3.4 Plotting

The file `outputs` contains the relative positions and velocities between spacecrafts in the leader's LVLH frame. In order to plot the obtained relative trajectory, the user must run the Matlab® script `plot_trajectory.m`:



4 Understanding the role of each function

The file `library.c` contains all the C functions used to perform the simulation. The role of each function is described hereafter:

- tic: used to measure the elapsed time, stocks the initial time;
- toc: used to measure the elapsed time, computes the difference between the current time and a stocked initial time and displays the difference in milliseconds;
- read_inputs: read the parameters in the `inputs` file and assign it to global variables;
- eq2in: converts the equinoctial orbital elements to inertial state (equation (2.7));
- c2eq: converts the classical orbital elements to equinoctial orbital elements (equation (2.6));
- accJ2: computes the J_2 acceleration in the RSW frame (equation (2.18));
- accAD: computes the atmospheric drag acceleration in the RSW frame (equation (2.19));
- rho: computes the value of the air density (equation (2.21))
- deq: computes the derivatives of the equinoctial orbital elements (equation (2.15))
- lvlh_rel: computes the relative state between spacecrafts in the leader's LVLH frame (equation (2.2.5));
- integrate: performs the loop described in section 2.7