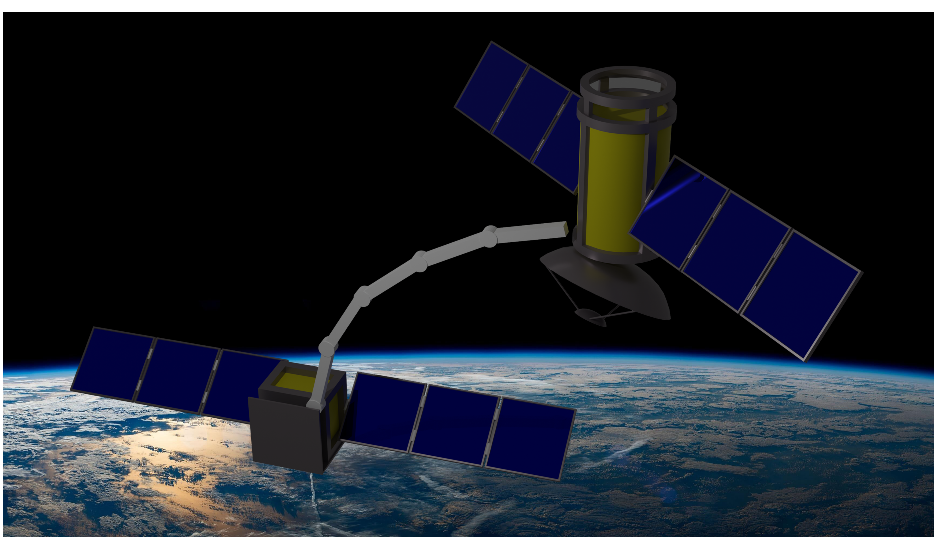
Final Project Report

Scazzi Antonio, Vergari Pierluigi
Villa Andrea, Vetrugno Fernando



Orbital Robotics and Distributed Space Systems (Prof. Marcello Romano)
Politecnico di Torino

Contents

1 The System

Although modern satellites should be completely independent, both from an energy supply and a de-orbit standpoint, many of them incur system failures or get damaged. The system here presented, through a 5 degrees of freedom mechanical manipulator, serves multiple purposes, among which repairing, refueling, and de-orbiting other satellites around the Earth.

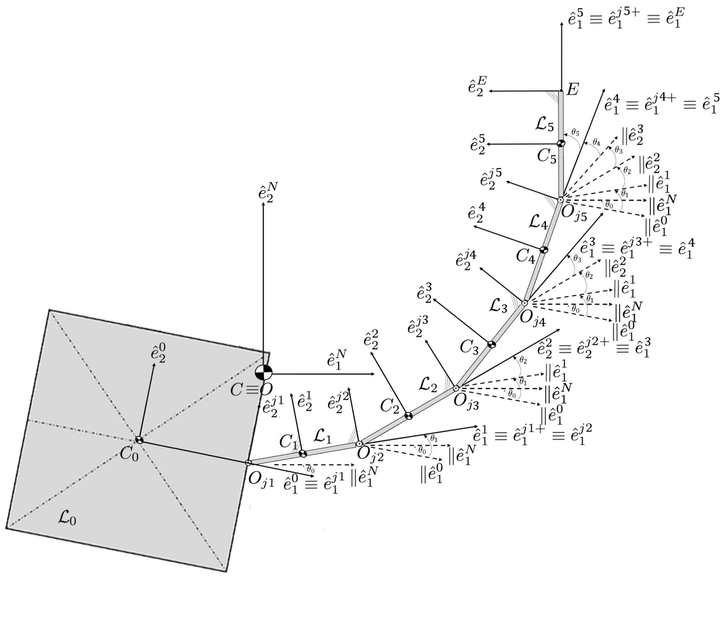


Figure 1: Scheme of the robotic manipulator and the links' reference frames.

Any of these missions requires the deputy to be put in orbit around Earth. To do it, the deputy has been designed to fit inside the payload volume of a Falcon 9 rocket alongside many other satellites [3]. This is required to lower the cost of the launch. Once in space, every type of mission requires the deputy to get near the chief, by being put in the same orbit and by closing the distance as much as possible through a rendezvous maneuver. Then, the deputy interacts with the target through its manipulator.
As shown in Figure 1, the system is designed with 6 links and 5 revolute joints. This allows the deputy to reach the space behind the base link without using its thrusters, thus sparing energy. Furthermore, many links and joints ensure a wide workspace, allowing the deputy to work from a longer distance.

For example, a de-orbit mission would be possible through berthing: the manipulator acts as a mechanical constraint between the chief and deputy; then, the deputy is directed into the atmosphere through a built-in thrust system, taking the chief with it [2].

2 Plan and Software

A de-orbit mission, which requires both rendezvous and docking, will be taken into consideration for the sizing of the free-flyer.

Focusing on the initial part of the mission, it's necessary to compute the orbits of the deputy with reference to the chief for different initial conditions and to calculate the most optimal total velocity to give to the spacecraft to execute a two-impulse maneuver. After that, the software analyzes the movement of the SC's manipulator's links to berth the target and start the de-orbit part of the mission.

Base dimensions	0.8 x 0.8 x 0.8 m^3
Link dimensions	0.1 x 0.1 x 0.4 m
Base mass	50 kg
Manipulator mass	10 (x5) kg
Total mass	100 kg

Preliminary design parameters.

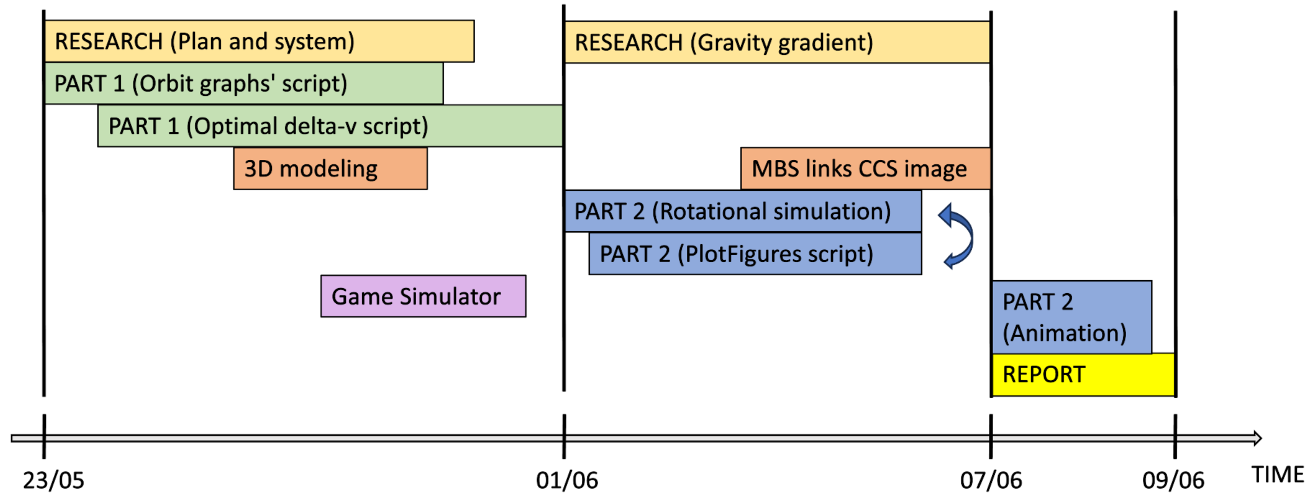


Figure 2: Project plan.

As showed in Figure 2, we have set three milestones:

1. The completion of the analysis of the translational motion (Pierluigi), along with document research (Andrea, Fernando) to develop a plausible and effective system and the development of the game-like simulation (Pierluigi, Andrea, Antonio);
2. The completion of the analytical part of the rotational-internal motion (Pierluigi, Andrea), along with research (everyone) to implement the effects of gravity on the movement of the links;
3. The completion of the animated part (Andrea, Pierluigi) of the rotational-internal motion, data-collecting, and the writing of the report (Fernando).

2.1 Analysis of the translational motion

The software's analysis begins by defining the relative position and velocity vectors between the deputy and the chief spacecraft at a specific point in time. Using this information, it employs a numerical integration method to propagate the spacecraft's motion forward in time. This process simulates the interaction between the spacecraft and the environment, accounting for gravitational forces exerted by the Earth.

The `'HCW_IC_MakeFigs'` script works by inputting the simulation time and the initial conditions (position and velocity of deputy w.r.t. chief). Then, it can print the orbits of the deputy in the Hill system by considering variations in positions or velocities from the initial values. The integration is done in the `'printFig'` script using ode45, since it looks to be more computationally efficient rather than the use of the exact solution with the function expm(). For example:

```
T = input('Enter the simulation time in seconds T = ');
x0 = [-500 -100 -50 -10 -5 5 10 50 100 500];
ICs = zeros(6,1);
printFig(ICs,x0,1,T)
```

Different graphs can be obtained for different starting imposed variations.

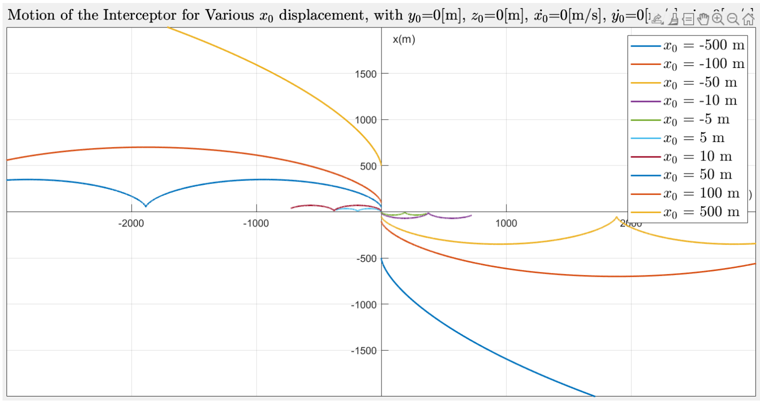


Figure 3: Orbits for different x_0 starting positions.

In Figure 3, as shown in the title, the initial conditions are null, which means that the deputy and the chief start from the same position with the same velocity. The parameter that varies is x_0, and the figure represents different orbits for different x_0, represented in the legend. Doing this considering variations of all the initial quantities (x_0, y_0, z_0 and their respective velocities), enables the program to print accurate orbits of the motion of the deputy spacecraft and to predict the response of the motion after every kind of perturbation of the initial conditions.

2.1.1 Minimum total ΔV for the two-impulse maneuver

This part of the software oversees the approaching phase of the deputy to the chief. In order to optimize the ΔV needed for the rendezvous (which aims at the minimum fuel consumption), the software computes the minimum value of the objective function $J = \|\Delta \underline{v_1}\| + \|\Delta \underline{v_2}\|$, which represents the total ΔV required for the two-impulse maneuver. The function `'ObjFun'` needs as input the vector $\underline{x_0}$, which contains the initial position and velocity of the deputy w.r.t. the chief.

Within the `'ObjFun'` function, the software computes $\underline{\Delta v_1}$ and $\underline{\Delta v_2}$, which are respectively the velocity required for the deputy to reach the same position as the chief ($x = y = z = 0$) and the velocity needed to eliminate the remaining relative velocity. To ensure a reasonable duration for the maneuver, the software searches for the minimum value of J satisfying the constraint $t \leq \frac{\tau}{2}$, where $\tau = 2\pi\sqrt{\frac{r_c^3}{\mu}}$ and r_c is the distance between the chief and Earth.

2.2 Analysis of the rotational-internal motion

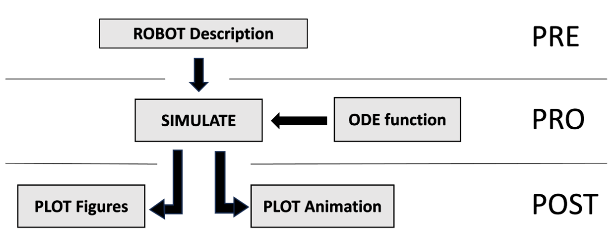


Figure 4: Block scheme of the development plan.

For the study of the rotation of the base link and the manipulator links, the software simulates, with a recursive analysis, the motion of all the links and computes fundamental quantities which are then utilized to evaluate the behavior of the deputy. in Figure 4 the three phases that constitute the analysis are shown: collecting geometry data in the pre-processing phase, computing the most important quantities in the processing phase, and producing two different representations of the results in the post-processing phase.

2.2.1 Pre-processing

The pre-processing part is conducted by the `'FFP6L5R_Robot_Description'` script. It collects all the information regarding the deputy using the Denavit and Hartenberg convention, including masses, sizes, inertial data, and every joint parameter. Then it encloses all the data, using a `'DH_Serial2robot'` function, that transforms the DH parameters into the `'robot'` model structure.

2.2.2 Processing

The main processing procedure is conducted by the `'SCRIPT1_FFP6L5R_Simulate'` script. It receives as input the configuration of the multi-body system: the base link orientation and position, the linear and angular velocities of the base link, and the speed of all the joints. Moreover, the user can set external and internal non-control wrenches, control actions (forces and torques acting on the links), and propagation parameters.

```
finaltime=tau/4;
propagation_timespan = 0:1:finaltime;
```

For our study, we have conducted the analysis for a quarter of an orbit ($\frac{\tau}{4}$). The numerical integration has been done with the solver `'ode113'`, which is more performing when dealing with nonstiff differential equations [4]. In the current analysis the gravitational effects have been taken into account, by using the following formulas [1] implemented in the ODE function:

$$\underline{f}_i = \cancel{-\frac{\mu m_i}{R_c^2}\hat{h}_1} - \frac{\mu m_i}{R_c^3}\underline{c}_i + 3\frac{\mu m_i}{R_c^3}\hat{h_1}\hat{h_1}\cdot\underline{c}_i \qquad\qquad \underline{g}^i_{\mathrm{Ci}} = \frac{3\mu}{R_c^3}\hat{h}_1 \times \underline{\underline{I}}^i_{\mathrm{Ci}}\cdot\hat{h}_1$$

Since we conducted our studies in a reference frame coincident with the center of mass of the system, and so with a non-rotating inertial reference frame orbiting at a constant tangential velocity given by the assumption of a circular orbit, the first term of the links forces has been neglected because it cancels out with the centrifugal force acting on the center of mass of each link. Moreover, in the aforementioned reference frame the $\hat{h_1}$ versor is rotating, and a time-varying direction cosine matrix $\mathbf{C}_{NH}$ has been used. In addition, in order to maintain the base parallel to the nadir direction, we have imposed an initial angular rate n relative to the CoM of the base, and the associated linear velocity of the base CoM w.r.t. the inertial reference frame.

2.2.3 Post-processing

In the post-processing phase, two different results could be analyzed:

- `'SCRIPT2_FFP6L5R_PlotFigures'` plots the figures representing fundamental quantities of the motion;
- `'SCRIPT3_FFP6L5R_Animation'` runs the pseudo-real-time animation of the robot orbiting with the gravitational effects acting on it on the left side and its position in the orbit around the Earth on the right side.

3 Conclusions

3.1 System's translational motion (Base + Manipulator)

Firstly, we validated the proper operation of the game-like simulator[1] by setting equivalent initial conditions in both the `'HCW_IC_MakeFigs'` script and the simulator. We compared the trajectory plotted by

[1]Game simulator is a function that allows the user to control in real-time the relative position of the deputy w.r.t. the chief.

the script (graphs in Section 2.1) with the evolution in time of the relative position of the deputy w.r.t. the chief generated by the simulator without interacting with the commands on the keyboard.

Secondly, to verify the correct implementation of the optimization problem, we computed the $\underline{\Delta v_1}$ needed to reach the target given a set of initial conditions through the `'Optimal_t'` script (as in Section 2.1.1). We added this velocity vector to the previous initial conditions as input for the game-like simulator, and we verified that the deputy reaches the chief exactly after an elapsed time equal to the optimal time given by the `'Optimal_t'` function. This test was also conducted without giving commands through the keyboard.

3.2 Optimal ΔV (Rendezvous maneuver)

After many attempts, the following observations have been developed:

1. Launching the analysis at $t = 0$ s requires the inversion of a singular matrix. Therefore, we start from $t = 0.1$ s. By doing this, we also rule out the case where the deputy and chief positions are superimposed, which entails an output $\underline{\Delta v_2}$ opposite to the relative initial velocity at $t = 0$;
2. Using equal and opposite x_0 or y_0 values, with other null initial conditions, the optimal final time is the same, and $\underline{\Delta v}$ are equal and opposite. For the x_0 case, the optimal final time is within $(0, \frac{\tau}{2})$, while for the y_0 case it's equal to $t_{max} = \frac{\tau}{2}$;
3. For each y_0, $\Delta v_{1x} = \Delta v_{2x}$ and $\Delta v_{1y} = -\Delta v_{2y}$;
4. Different patterns can be observed with mixed initial conditions, which have not been considered in this analysis due to time and space constraints.

3.3 System's rotational motion (Base + Manipulator)

With the imposed initial displacement of the first joint ($\frac{\pi}{4}$ w.r.t. the local vertical), we have observed that the linear momentum is preserved (Figure 6) because $\sum_{i=0}^{5} \underline{f}_i = \underline{0}$ given the chosen reference frame (as explained in Section 2.2.2). Differently, the moment of momentum w.r.t. the center of mass is not preserved (Figure 7), because in this case the resultant moment w.r.t. the center of mass is not null, since the gravity gradient is at work on the links. Looking at Figure 5, as we expected, the center of mass is fixed in the origin of the inertial reference frame.

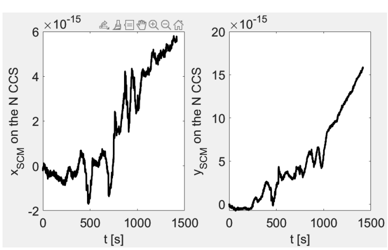


Figure 5: Position of the CoM of the system.

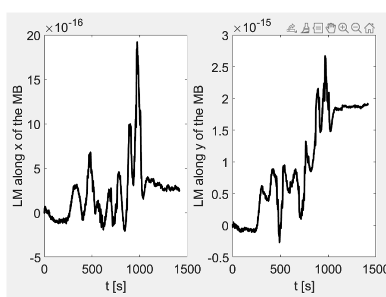


Figure 6: Linear Momentum.

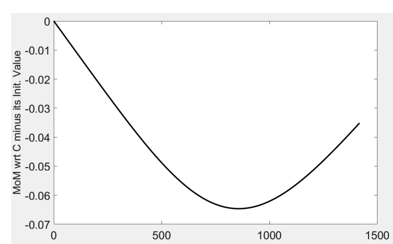


Figure 7: Moment of Momentum w.r.t. C.

Another test has been conducted with the manipulator aligned with the local vertical and with the same initial conditions as before (Section 2.2.2). Besides the conservation of the linear momentum and the position of the system CoM w.r.t. the inertial reference frame, it resulted that the moment of momentum w.r.t. the system CoM conserves since the principal axes of each link are aligned with the $\hat{h_1}$ versor ($\underline{\underline{I}}^i_{\mathrm{Ci}} \cdot \hat{h}_1$ is parallel to $\hat{h_1}$), therefore the moment acting on each link is equal to zero, and so $\sum_{i=0}^{5} \underline{g}^i_{Ci} = \underline{0}$.

References

[1] M. Romano, *Orbital Robotics and Distributed Space Systems, Class Notes*. Politecnico di Torino, 2023

[2] T. Wolf, *Deutsche Orbitale Servicing Mission*. Deutsches Zentrum für Luft- und Raumfahrt, 2012

[3] SpaceX, *Falcon User's Guide*. SpaceX, 2021

[4] S. Berrone, S. Pieraccini, *Numerical methods and scientific computing, Class Notes*. Politecnico di Torino, 2022