sid.inpe.br/mtc-m21c/2019/10.04.12.14-RPQ

INTERNSHIP REPORT: SOFTWARE ENVIRONMENT FOR BERTHING MANEUVERS OF SATELLITES ENDOWED WITH ROBOTIC MANIPULATORS

Anderson Brazil Nardin

INPE
São José dos Campos
2019

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GBDIR)
Serviço de Informação e Documentação (SESID)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

sid.inpe.br/mtc-m21c/2019/10.04.12.14-RPQ

INTERNSHIP REPORT: SOFTWARE ENVIRONMENT FOR BERTHING MANEUVERS OF SATELLITES ENDOWED WITH ROBOTIC MANIPULATORS

Anderson Brazil Nardin

INPE
São José dos Campos
2019

ACKNOWLEDGEMENTS

I am grateful to many people. I would like to thank my advisor Dr. Evandro Marconi Rocco for his important advices and support.

I would also like to thank my colleagues from the On-Orbit Servicing and Autonomy group of the Space Flight Technology department at DLR, Heike Benninghoff, Florian Rems, Eicke-Alexander Risse, Kurt Schwenk and Ksenia Klionovska for their assistance and friendship, without them the results presented here would not have been possible.

Finally, but not less important, I thank CAPES for its financial support during the internship program (process number 88881.188613/2018-01).

ABSTRACT

The modeling of a robotic system in a space environment shall be thoroughly investigated, with emphasis on the disturbances its movement causes on the pose of the satellite, which serves as base to a robotic manipulator. These disturbances are considered torques generated by the actuation of the robotic mechanisms during berthing between artificial satellites. The movement of the satellite considered as the base of the robotic arm, due to the coupled manipulator’s moves, dynamically changes the distance between end effector and target point. A previous analysis suggests that the development of models that consider the dynamic correction of positioning errors and simultaneous, as well as cooperative, operation of the artificial satellite's and robot's control systems provides advantages in such missions. My study proposes using the available EPOS robots (European Proximity Operations Simulator) at DLR (German Aerospace Center) to exploit concepts of hardware-in-the-loop and real-time simulations. In this scenario, two physical robots play the role of chaser and target satellites involved in a berthing maneuver, while a virtual robotic manipulator coupled to the chaser satellite is emulated by the implemented software. The robotic arm which serves as an object of study in this work consists of a revolute manipulator with three rotating joints and three degrees of freedom in a Torsional - Rotational - Rotational (TRR) configuration moving in space. Such configuration gives it diverse applications and notable usefulness in the accomplishment of on-orbit servicing. The experiments were useful to prove the validity of the developed algorithms and this work achieved success in the task of creating a reliable software environment for tests of berthing maneuvers.

LIST OF FIGURES

Page

CONTENTS

Page

1 INTRODUCTION

This document follows an approach supported on the proposal approved by the doctorate board to which it was applied for. This report contains descriptions of the suggestions and studies conducted in the laboratories at DLR (German Aerospace Center).

The work conducted by Santos (2015) is an excellent example of a well-succeeded experience, conducted in the German Aerospace Center (DLR) laboratory. We will not discuss the merits of the study, nor describe it; instead, we are going to address some features of these laboratories (explored in the aforementioned work) in order to collaborate with the research. These features might be useful for the issues proposed here, considering a possible INPE/DLR cooperation in the same way as the cooperation established for the work of Santos (2015).

Putting it into context, Santos (2015), at a certain point in his research, addressed a scenario of a final rendezvous maneuver between spacecrafts, due to its complexity, in order to test its models. Additionally, he considered a promising use of the rendezvous and docking simulator with hardware-in-the-loop of the German Aerospace Center, which is called the European Proximity Operations Simulator (EPOS). This maneuver simulator has been used to test and validate proposed models.

The simulator in question uses two industrial robots to physically simulate the complete translational and rotational movement of two different satellites operating rendezvous and docking maneuvers. Moreover, all the guidance, navigation and control loop elements were developed and implemented in a simulation environment and tested in real-time at EPOS employing real sensors.

Lastly, the developed software presented effectiveness and robustness, proving to be able to generate reliable outcomes in both non-real-time and real-time simulations.

1.1. Objectives

This work can be understood as a natural progress of those outcomes obtained in Nardin (2015), where a simulation tool for berthing maneuvers was elaborated and had its functionalities properly explored.

Given the research power of the DLR facilities briefly shown here, and the enormous empirical and experimental capacity that EPOS represents, we can introduce the intended experimentation ideas.

The EPOS facilities' robots can be used to obtain outcomes that confirm those from computer simulation, adding realism and reliability to the whole research thanks to the inclusion of hardware in the loop of control and simulation. Summarizing, the effort of putting two robotic manipulators to emulate the moves to which the structure (formed by manipulator plus chaser satellite coupled) and the target satellite are subjected, promotes great benefits to research.

To better explain the approach: one of the robots would act as the actual chaser artificial satellite, equipped with its own control system, which serves as the base to the attached robotic manipulator – which, by its turn, is simulated by the computer-based software, seeking to achieve a target point within its work volume, and thus characterizing a berthing maneuver.

Each component system, being the robotic manipulator role played by the developed software simulator and the chaser satellite role played by one of the robots physically available, is going to be subjected to disturbance resulting from the other's movement. The second robot that compounds the EPOS facility comply with the target satellite role.

We can understand the ensemble formed by satellite and manipulator as a hybrid system formed by a physical emulated satellite, a robot performs its attitude, and a virtual simulated manipulator, a software provides its motion. Once both parts are attached together (virtually), such behavior, of mutual sensitivity, can be verified and realized for the hardware-in-the-loop (HIL) presence. In this case, the HIL concept would be explored, aiming to ascertain the maneuver effectiveness. For instance, a position sensor such as a camera system (and its proper image processing algorithm) should determine the actual position of the satellites.

In the end of this document, it will be presented how it was possible to test the developed space environment software simulator for berthing maneuvers through real-time (RT - VxWorks operating system) and hardware-in-the-loop simulations using the European Proximity Operations Simulator (EPOS).

2 MODELS AND WORK DESCRIPTION

2.1. Mathematical models

In a general view, it was developed a simulator for robotic systems in a space environment. The movement of the base satellite, due to the coupled manipulator’s moves, dynamically changes the distance between end effector and target point.

During the simulated maneuver, it is possible to verify how the center of mass of the ensemble, satellite that serves as base plus the robotic-three-jointed manipulator, changes along the time. Using the iterative Newton-Euler algorithm, it is proposed an approach based on the ongoing changes in moments of inertia matrix and its consequences to the servicer satellite dynamics as well.

If you move the manipulator arm during the berthing, the servicer's base will not be constant, since you change the center of mass and the moments of inertia of the coupled system. In fact, the changes in the center of mass and moments of inertia were already considered in this simulator. It is possible to see satellite and robot interfering with each other during a berthing maneuver.

2.1.1. Kinematics

The robotic arm adopted here is an anthropomorphic robot (Torsional – Rotational – Rotational, Figure 2.1). From which we obtained equations in order to solve the direct and inverse kinematics, i.e., given the desired position to the end effector we find the joint angles able to take the robot’s claw to that position and vice-versa.

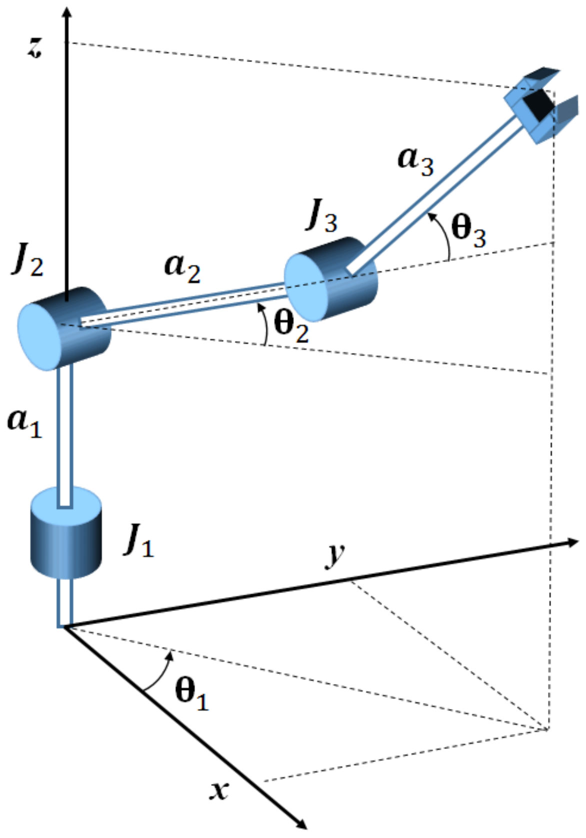


Figure 2.1 - Robotic arm TRR.

Source: Nardin (2015).

We can, using trigonometric relations, find Equations 2.1, 2.2 and 2.3 for inverse kinematics and Equations 2.4, 2.5 and 2.6 for direct kinematics, considering each joint respectively from *1* to *3*.

$$\theta_1 = \arctan \frac{y}{x} \tag{2.1}$$

$$\theta_2 = \arctan \left[\frac{(z-a_1)(a_2 + a_3\cos\theta_3) - \sqrt{x^2+y^2}\,a_3\sin\theta_3}{\sqrt{x^2+y^2}\,(a_2 + a_3\cos\theta_3) + (z-a_1)a_3\sin\theta_3} \right] \tag{2.2}$$

$$\theta_3 = \arccos \left(\frac{x^2 + y^2 + (z-a_1)^2 - a_2^2 - a_3^2}{2a_2a_3} \right) \tag{2.3}$$

$$x = [a_2 \cos\theta_2 + a_3 \cos(\theta_2 + \theta_3)]\cos\theta_1 \tag{2.4}$$

$$y = [a_2 \cos\theta_2 + a_3 \cos(\theta_2 + \theta_3)]\sin\theta_1 \tag{2.5}$$

$$z = a_1 + a_2 \sin\theta_2 + a_3 \sin(\theta_2 + \theta_3) \tag{2.6}$$

2.1.2. Moments of inertia

The moment of inertia tensor relative to the reference system *{A}* is expressed in Equation 2.7 (CRAIG, 2005).

$${}^{A}\mathbf{I} = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{bmatrix} \tag{2.7}$$

Where elements are given by:

$$I_{xx} = \iiint_V (y^2 + z^2)\rho dv \tag{2.8}$$

$$I_{yy} = \iiint_V (x^2 + z^2)\rho dv \tag{2.9}$$

$$I_{zz} = \iiint_V (x^2 + y^2)\rho dv \tag{2.10}$$

$$I_{xy} = \iiint_V xy\rho dv \tag{2.11}$$

$$I_{xz} = \iiint_V xz\rho dv \tag{2.12}$$

$$I_{yz} = \iiint_V yz\rho dv \tag{2.13}$$

Being ρ the volumetric mass density. Sometimes, it is necessary to calculate the moment of inertia tensor with relation to the center of mass, which is given by Equation 2.14.

$$\mathbf{P}_c = [x_c, y_c, z_c]^T \tag{2.14}$$

Given that *{C}* is the body center of mass system and *{A}* a non-particular system, we have through the Steiner theorem application (CRAIG, 2005):

$$^{A}\mathbf{I} = {}^{C}\mathbf{I} + m[\mathbf{P}_c^{\ T}\mathbf{P}_c\mathbf{I}_3 - \mathbf{P}_c\mathbf{P}_c^{\ T}] \tag{2.15}$$

Where $\mathbf{I}_3$ is the *3* by *3* identity matrix.

The moment of inertia tensor relative to the center of mass of a parallelepiped will be useful because that is the standard links' shape.

$$^{C}\mathbf{I} = \begin{bmatrix} \frac{m}{12}(h^2+l^2) & 0 & 0 \\ 0 & \frac{m}{12}(w^2+h^2) & 0 \\ 0 & 0 & \frac{m}{12}(l^2+w^2) \end{bmatrix} \tag{2.16}$$

Where *m, l, w* and *h* are body's mass, length, width and height, respectively.

2.1.3. Newton-Euler algorithm

If a rigid body has some acceleration in its center of mass it must have been caused by a force with respect to a system *i*, such that:

$$\vec{F}_i = m\dot{\vec{v}}_{C_i} \tag{2.17}$$

If this body is in rotation, with angular velocity and acceleration, then there exists a torque which acts on the body in order to cause a movement variation.

$$\vec{N}_i = {}^{C_i}\mathbf{I}\dot{\vec{\omega}}_i + \vec{\omega}_i \times {}^{C_i}\mathbf{I}\vec{\omega}_i \tag{2.18}$$

Where *{C}* has its origin on the link center of mass and the same orientation of the reference system *{i}*. We use the iterative Newton-Euler to calculate torques along the manipulator movement. The algorithm works in two stages: Outward, velocities and accelerations propagation, forces and torques calculation on each link from the first system to the last*;* Inward*,* forces and torques executed by each joint, from end effector to the manipulator's base (CRAIG, 2005; ELLERY, 2000). Equations for Outward:

$$^{i+1}\vec{\omega}_{i+1} = {}^{i+1}_{i}\mathbf{R}\,{}^{i}\vec{\omega}_i + \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} \tag{2.19}$$

$$^{i+1}\dot{\vec{\omega}}_{i+1} = {}^{i+1}_{i}\mathbf{R}\,{}^{i}\dot{\vec{\omega}}_i + {}^{i+1}_{i}\mathbf{R}\,{}^{i}\vec{\omega}_i \times \dot{\theta}_{i+1}\,{}^{i+1}\hat{Z}_{i+1} + \ddot{\theta}_i\,{}^{i+1}\hat{Z}_{i+1} \tag{2.20}$$

$$^{i+1}\dot{\vec{v}}_{i+1} = {}^{i+1}_{i}\mathbf{R}({}^{i}\dot{\vec{\omega}}_i \times {}^{i}\vec{P}_{i+1} + {}^{i}\vec{\omega}_i \times ({}^{i}\vec{\omega}_i \times {}^{i}\vec{P}_{i+1}) + {}^{i}\dot{\vec{v}}_i) \tag{2.21}$$

$$^{i+1}\dot{\vec{v}}_{C_{i+1}} = {}^{i+1}\dot{\vec{\omega}}_{i+1} \times {}^{i+1}\vec{P}_{C_{i+1}} + {}^{i+1}\vec{\omega}_{i+1} \times ({}^{i+1}\vec{\omega}_{i+1} \times {}^{i+1}\vec{P}_{C_{i+1}}) + {}^{i+1}\dot{\vec{v}}_{i+1} \tag{2.22}$$

$$^{i+1}\vec{F}_{i+1} = m_{i+1}\,{}^{i+1}\dot{\vec{v}}_{C_{i+1}} \tag{2.23}$$

$$^{i+1}\vec{N}_{i+1} = {}^{C_{i+1}}I_{i+1}\,{}^{i+1}\dot{\vec{\omega}}_{i+1} + {}^{i+1}\vec{\omega}_{i+1} \times {}^{C_{i+1}}I_{i+1}\,{}^{i+1}\vec{\omega}_{i+1} \tag{2.24}$$

Equations for Outward Inward*:*

$$^{i}\vec{f}_i = {}^{i}_{i+1}\mathbf{R}\,{}^{i+1}\vec{f}_{i+1} + {}^{i}\vec{F}_i \tag{2.25}$$

$$^{i}\vec{n}_i = {}^{i}\vec{N}_i + {}^{i}_{i+1}\mathbf{R}\,{}^{i+1}\vec{n}_{i+1} + {}^{i}\vec{P}_{C_i} \times {}^{i}\vec{F}_i + {}^{i}\vec{P}_{i+1} \times {}^{i}_{i+1}\mathbf{R}\,{}^{i+1}\vec{f}_{i+1} \tag{2.26}$$

In Ellery (2000) it is dedicated special attention to works that reflect the computational superiority of Newton-Euler formulation, against others such as Lagrangian. Figure 2.2 shows reference systems fixed to each joint center of mass (above) and the reference systems fixed to two successive joints (below).

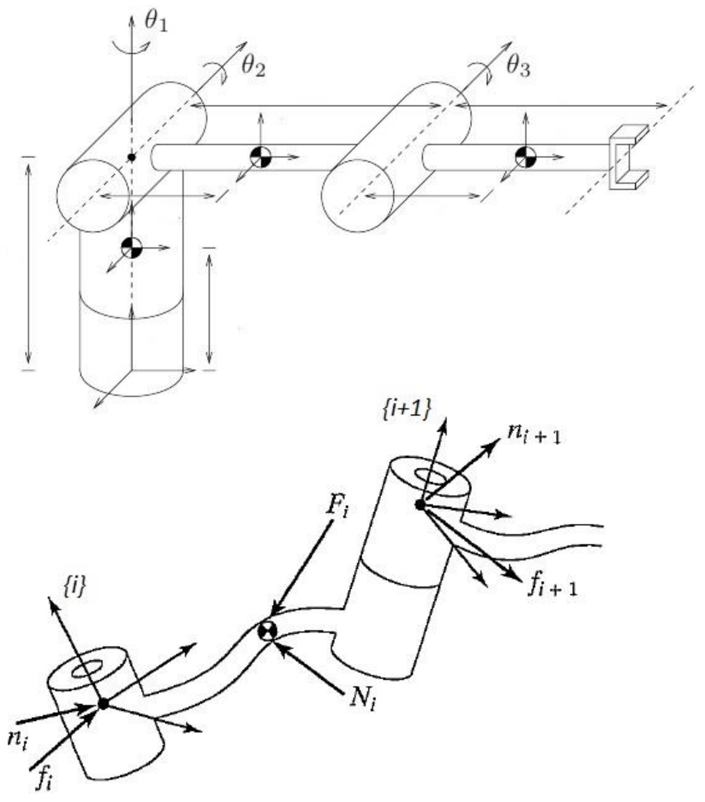


Figure 2.2 - System on the manipulator.

Source: Adapted from Craig (2005).

2.1.4. Satellite simulator

The satellite control system has a PID controller (Proportional – Integral - Derivative) with its respective gains K_P, K_I, K_D, which follows the control law defined by Equation 2.27, where $er(t)$ is the error signal. Figure 2.3 shows the control system configuration, while Figure 2.4 presents the ensemble formed by cubic shape base satellite plus robotic manipulator. The physical characteristics of each manipulator's link and satellite are provided by Table 4.1.

$$c(t) = K_P er(t) + K_I \int er(t)dt + K_D \frac{d}{dt} er(t) \qquad (2.27)$$

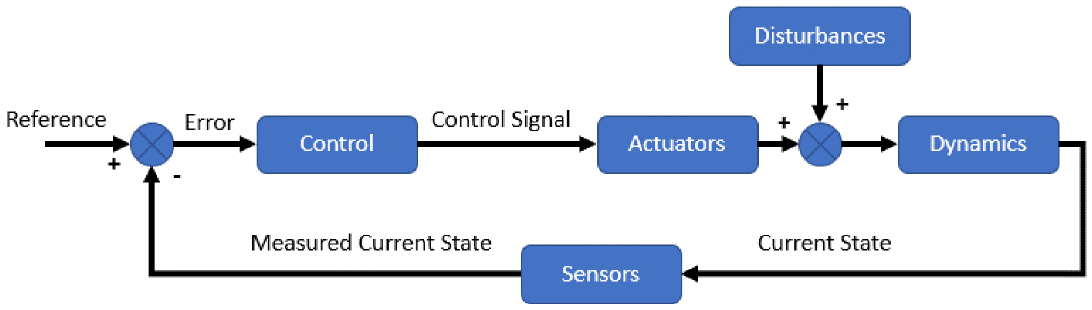


Figure 2.3 - Control configuration.

Source: Nardin (2015)

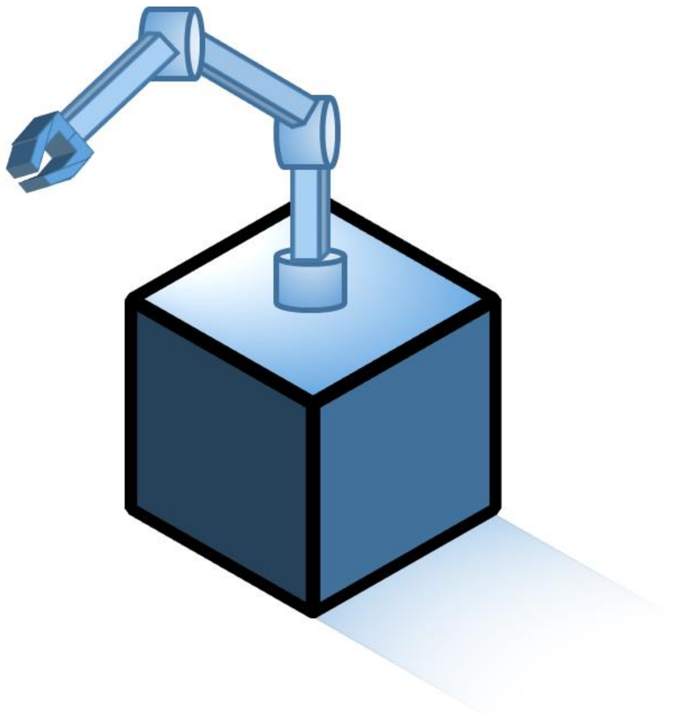


Figure 2.4 - Ensemble arm and satellite (out of scale).

Source: Nardin (2015)

	Length(m)	Width(m)	Height(m)	Mass (kg)
Link 0	0,5	1	1	40
Link 1	1	0,1	0,1	20
Link 2	1	0,1	0,1	20
Satellite	2	2	2	500

Table 2.1 - Physical characteristics

Source: Nardin (2015)

Using direct cosine matrix, we can obtain any final satellite attitude. For example, rotations around axis *X,Y,Z* (or *1-2-3* in this order) produce a matrix as seen in Equations 2.28 and 2.29 (HUGHES, 1986).

$$\mathbf{C}_{ba} = \mathbf{C}_3(\theta_3)\mathbf{C}_2(\theta_2)\mathbf{C}_1(\theta_1) \tag{2.28}$$

$$\mathbf{C}_{ba} = \begin{bmatrix} c_3c_2 & s_3c_1 + c_3s_2s_1 & s_3s_1 - c_3s_2c_1 \\ -s_3c_2 & c_3c_1 - s_3s_2s_1 & c_3s_1 + s_3s_2c_1 \\ s_2 & -c_2s_1 & c_2c_1 \end{bmatrix} \tag{2.29}$$

Where *c* represents cosine, *s* represents sine and indexes *1, 2, 3* represent angles $\theta_1, \theta_2, \theta_3$ respectively. Equation 2.30 reveals how is an attitude matrix considering an infinitesimal angle sequence (HUGHES, 1986).

$$\mathbf{C}_{ba} \cong \begin{bmatrix} 1 & \theta_3 & -\theta_2 \\ -\theta_3 & 1 & \theta_1 \\ \theta_2 & -\theta_1 & 1 \end{bmatrix} = \mathbf{I}_3 - \boldsymbol{\theta}^{\times} \tag{2.30}$$

A rigid body spins in the system *O'*, with angular velocity with relation to the inertial system *O,* as shown in Figure 2.5.

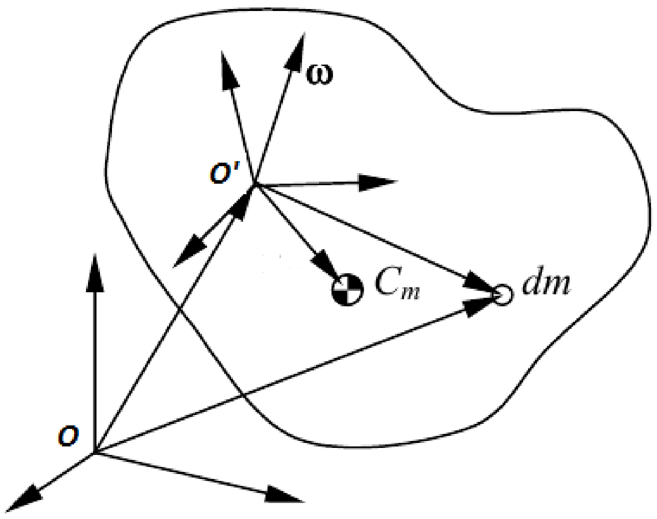


Figure 2.5 - Rigid body movement.

Source: Adapted from Hughes (1986).

If the origin of the system *O'* fixed to the body coincides with the center of mass, the dynamics equations of the satellite are deducted as follows (HUGHES, 1986). Where $\mathbf{p}$ is linear momentum, $\mathbf{h}_c$ is the angular momentum with relation to the center of mass, $\mathbf{v}_c$ is the center of mass linear velocity and $\boldsymbol{\omega}$ is the angular velocity.

$$\mathbf{p} = m\mathbf{v}_c \tag{2.31}$$

$$\mathbf{h}_c = \mathbf{I}\boldsymbol{\omega} \tag{2.32}$$

$$\dot{\mathbf{p}}^b = \mathbf{f} - \boldsymbol{\omega}^{\times}\mathbf{p} \tag{2.33}$$

$$\dot{\mathbf{h}}_c^b = \mathbf{g}_c - \boldsymbol{\omega}^{\times}\mathbf{h}_c \tag{2.34}$$

We define, $\mathbf{f}$ as the resulting of external forces and $\mathbf{g}_c$ as the resulting of external torques, both applied to the rigid body on its center of mass. Considering Equations 2.35 and 2.36, we find Equations 2.37 and 2.38.

$$\dot{\mathbf{p}}^b = m\dot{\mathbf{v}}_c^b \tag{2.35}$$

$$\dot{\mathbf{h}}_c^b = \mathbf{I}\dot{\boldsymbol{\omega}}^b \tag{2.36}$$

$$\dot{\mathbf{v}}_c^b = m^{-1}\mathbf{f} - \boldsymbol{\omega}^{\times}\mathbf{v}_c \tag{2.37}$$

$$\dot{\boldsymbol{\omega}}^b = \mathbf{I}^{-1}(\mathbf{g}_c - \boldsymbol{\omega}^{\times}\mathbf{I}\boldsymbol{\omega}) \tag{2.38}$$

Figure 2.6 shows vectors and reference systems used in order to obtain the target vector to the manipulator's tip. In a real mission, the robot's wrist could be equipped with a claw for example. There are uncountable tools which could be used on an end effector like this proposed here, depending on which kind of task should be performed. Being *T* the representation of a handle position in space, we can deduct Equations 2.39 and 2.41.

$$\vec{V}_{OT} = \vec{V}_{CT} + \vec{P}_{CM} \tag{2.39}$$

$$\mathbf{C}_{ca} = \prod_{i=1}^{n} \mathbf{C}_{ba_i} = \mathbf{C}_{ba_n} \mathbf{C}_{ba_{n-1}} \ldots \mathbf{C}_{ba_2} \mathbf{C}_{ba_1} \tag{2.40}$$

$$\vec{V}_{ag} = \mathbf{C}_{ca} \vec{V}_{OT} - \vec{P}_{SR} \tag{2.41}$$

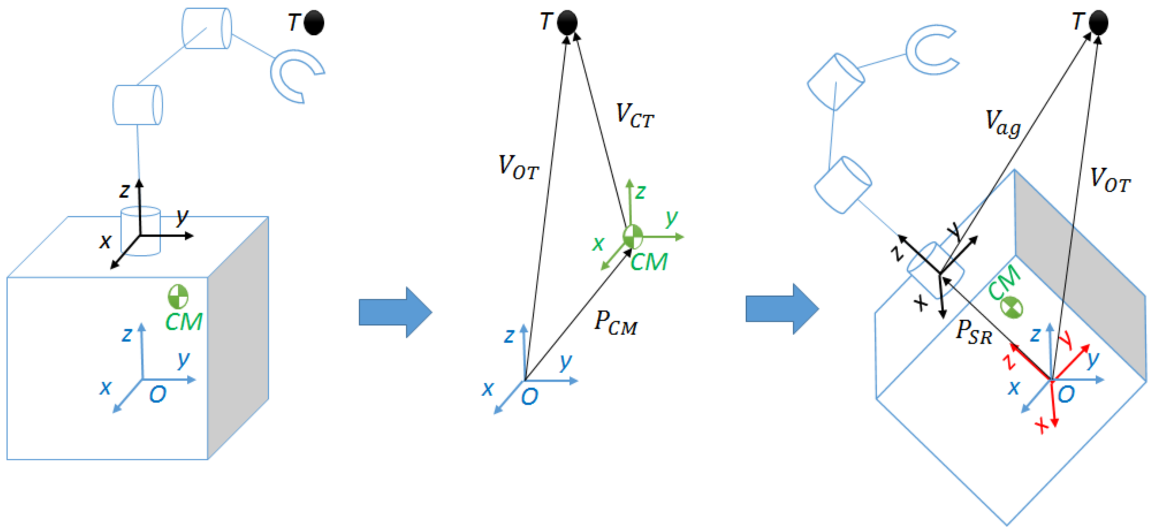


Figure 2.6 - Obtention of target vector to the end effector.

Source: Nardin (2015)

2.2. Simulation tool

We must remember that the aim of this study is to investigate the modeling of a robotic system in a space environment. We know also that the movement of the robot base, due to satellite repositioning through its actuators, alters dynamically the distance to the target point.

It is worth keeping in mind that the robotic arm, which serves as the object of study in this work, consists of a revolute manipulator with three rotating joints and three degrees of freedom in a Torsional-Rotational-Rotational (TRR) configuration moving in space. Such configuration brings diverse applicability and notable usefulness in the accomplishment of on-orbit servicing.

Figure 2.7 illustrates an animation frame generated while the developed simulator operates one of many maneuvers. This can be used to confirm that the manipulator has achieved a specified target point in space.

Figure 2,8 shows, in three-dimensional space, the torque in the satellite due to robot movement, while Figure 2,9 presents this with separate charts for each component, one for each dimension. These charts are depicted in function of time. Through those last charts, we have an idea of the order of magnitude of the torques that result from the robotic manipulator movement.

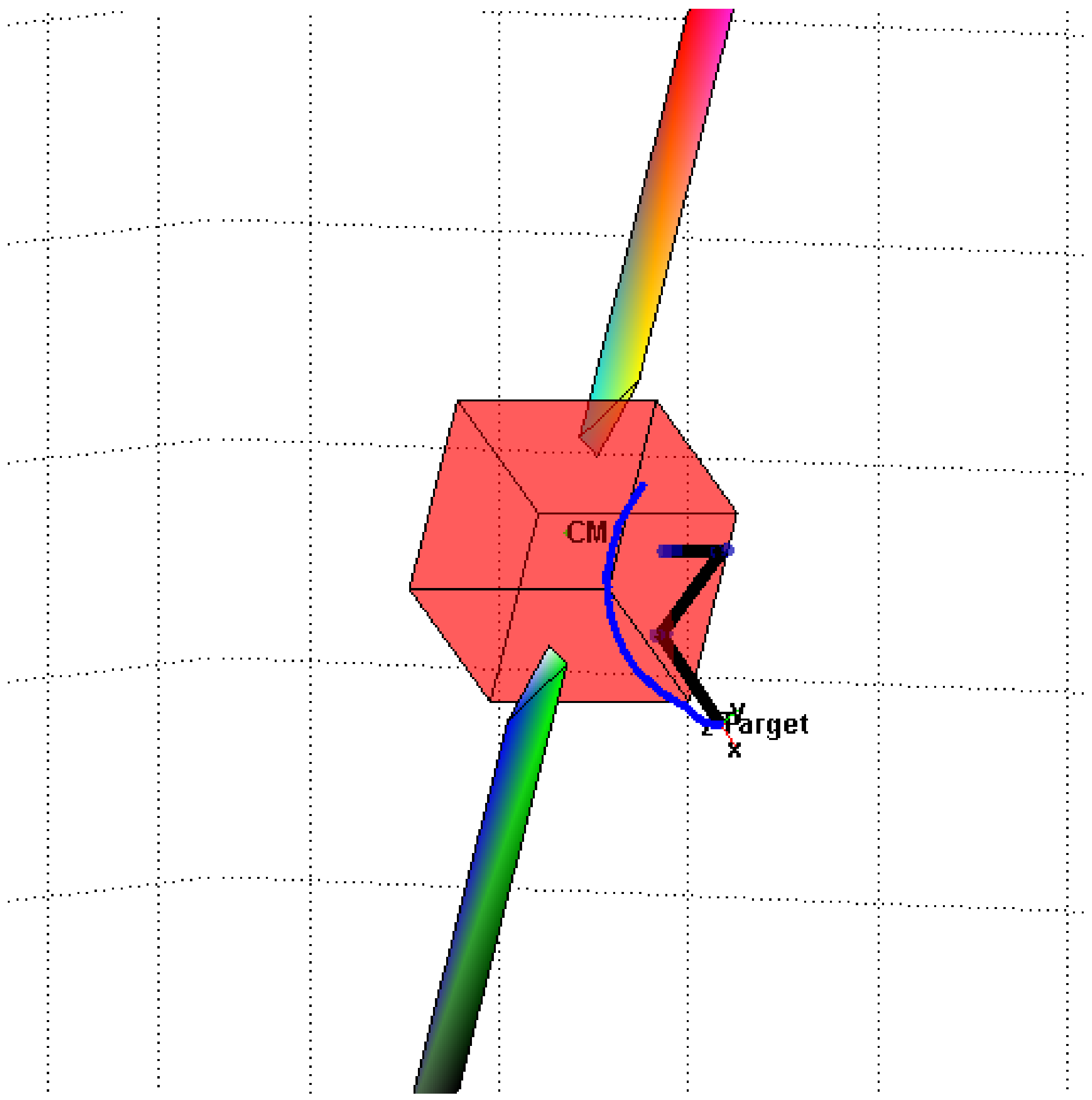


Figure 2.7 - Robotic Manipulator and Satellite.

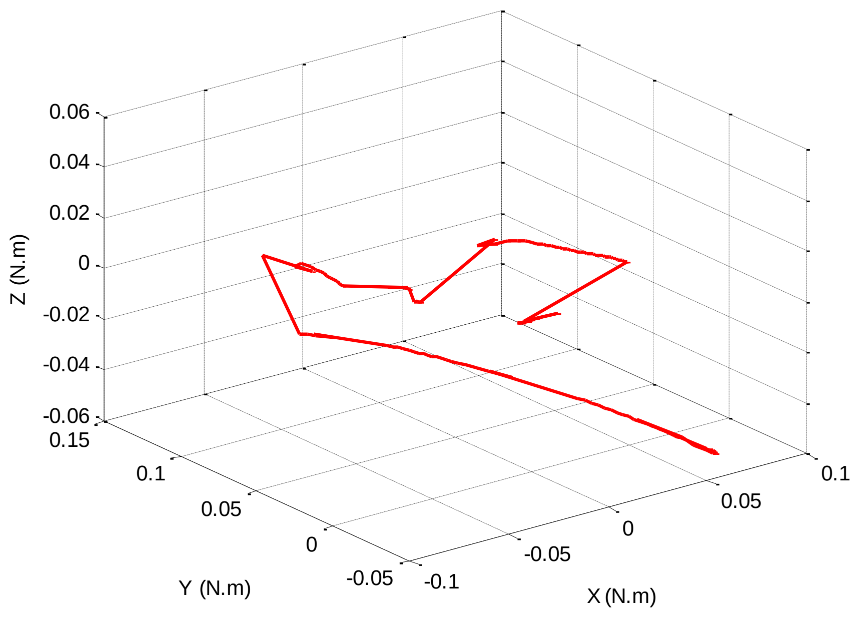


Figure 2.8 - Torque in the satellite in three dimensions.

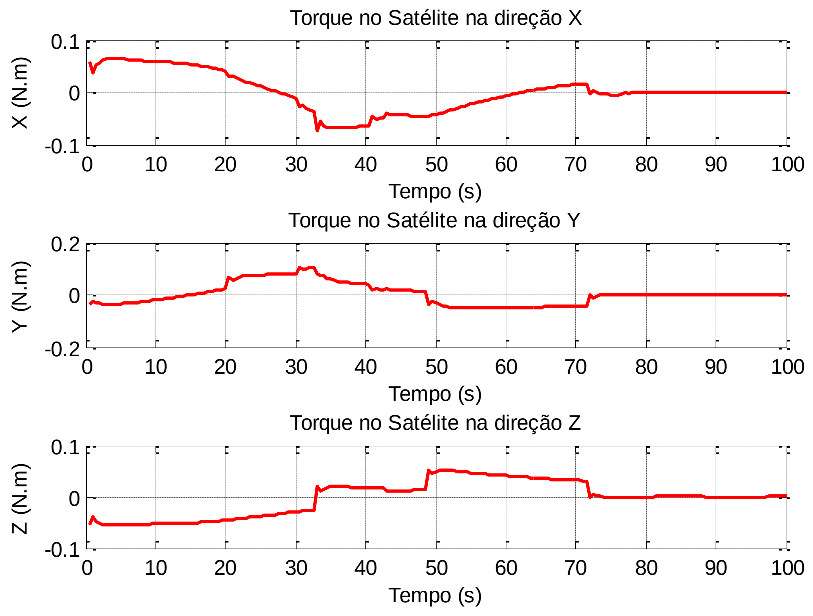


Figure 2.9 - Torque over time in different satellite axis.

Figures 2.10 and 2.11 show the symmetric behavior of torques from the satellite's attitude control system and those generated by robotic manipulator movements, respectively.

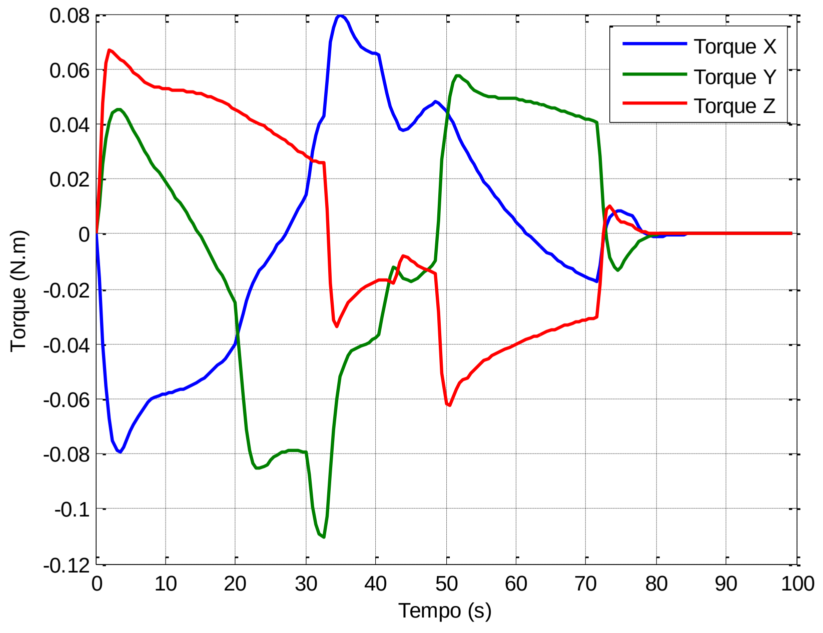


Figure 2.10 - Satellite GNC System Control.

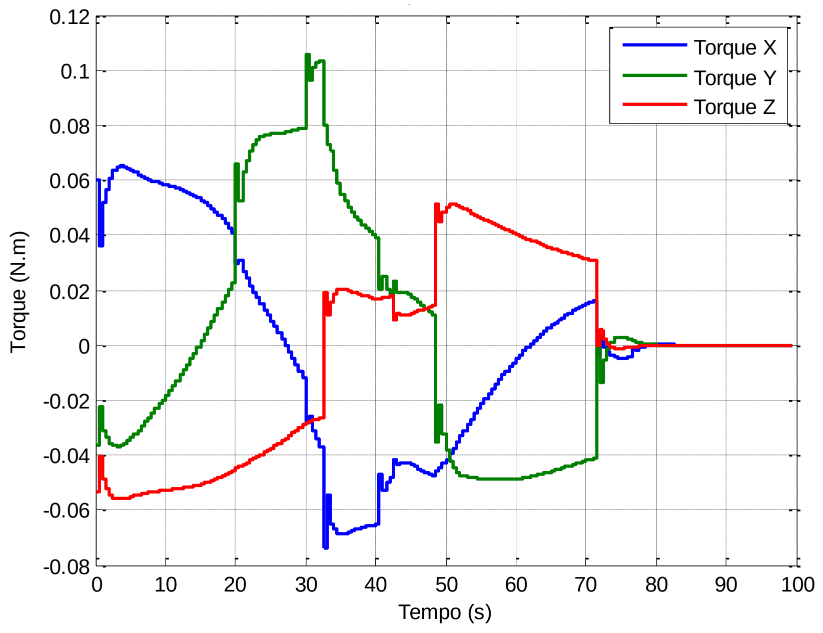


Figure 2.11 - Robot’s Torque.

2.3. Simulator with hardware-in-the-loop EPOS 2.0

Simulation with hardware-in-the-loop (HIL) is a technique that has been well developed and used in the design and validation of control systems. Its basic concept sums up to include real hardware (it can be understood as devices or any sort of hardware system) in real-time loop simulation. Simulations like these have been used for over forty years and had its origins in flight simulations. Recently, the usefulness of this kind of simulation has been noticed in several areas like automotive, robotics, space systems, etc.

The first version of EPOS was largely used for testing sensors and systems in rendezvous simulations. However, in order to improve the test and simulation capabilities, the new EPOS 2.0 system was built in 2009. The new EPOS provides test and verification capabilities for rotational and translational movements of two satellites: chaser and target. It is also useful for carrying out On-Orbit servicing (OOS).

The EPOS facility comprises two industrial robots. They are separated by 0 to 25 meters, which are utilized on realistic simulations of rendezvous and docking process (see Figure 2.12). In general, experiments of integration, testing and verification are performed in the Hardware-In-the-Loop EPOS at the German Aerospace Center (Deutsche Zentrum für Luft-und Raumfahrt - DLR) in Oberpfaffenhofen, Germany.

This facility has been described as an important tool and its potential has been exploited through many tests campaign, including end-to-end simulation environment for rendezvous and grasping of space debris (BENNINGHOFF et al.,2018).

Figure 2.13 presents the equipment’s layout. There is a rail system assembled on the floor in order to move one of the robots. The so-called Robot 1, a KUKA KR100HA, is assembled on a rail that simulates 6 degrees of freedom of a spacecraft, while Robot 2, a KUKA KR240, simulates other spacecraft with its own degrees of freedom. There is also a control and monitoring system that is able to manage the whole simulation.

Figure 2.14 illustrates a closed-loop simulation of a rendezvous. The system is based on measures of relative positioning between satellite models, and it can control the maneuver simulation.

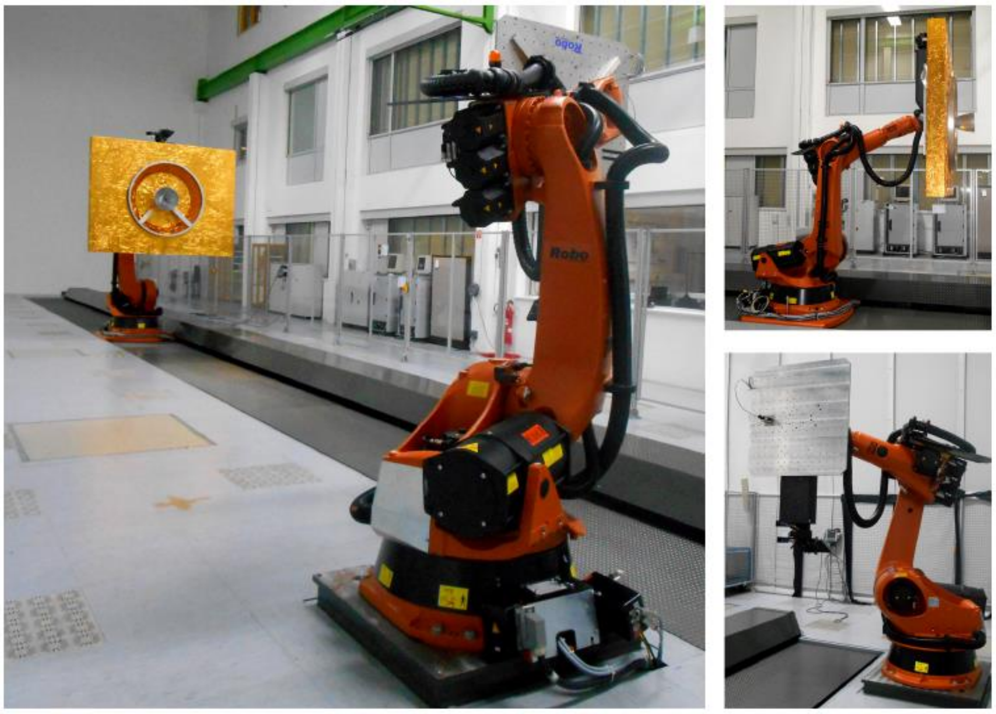


Figure 2.12 - EPOS Robots.

Source: Santos (2015).

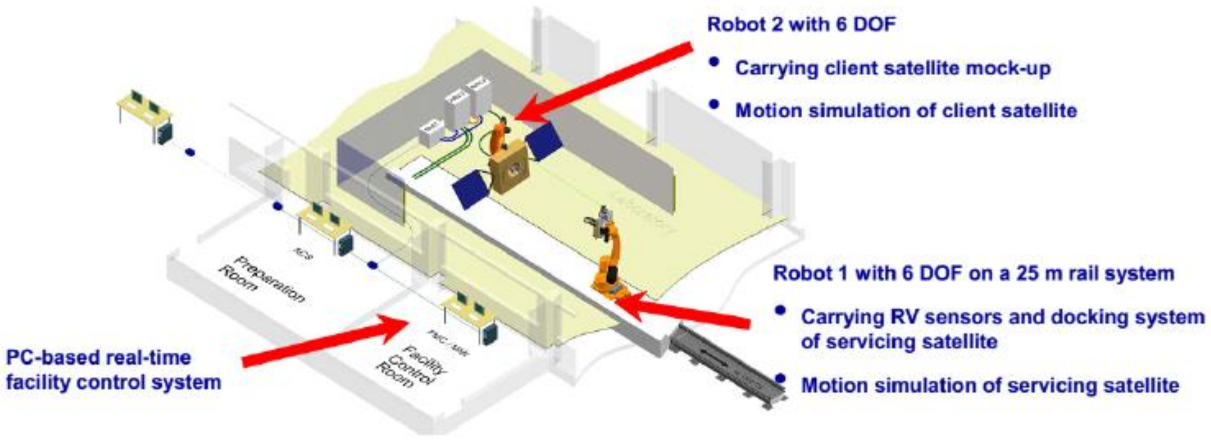

Robot 2 with 6 DOF
Carrying client satellite mock-up
Motion simulation of client satellite
Robot 1 with 6 DOF on a 25 m rail system
Carrying RV sensors and docking system of servicing satellite
Motion simulation of servicing satellite
PC-based real-time facility control system
Preparation Room
Facility Control Room

Figure 2.13 - EPOS system components.

Source: Santos (2015).

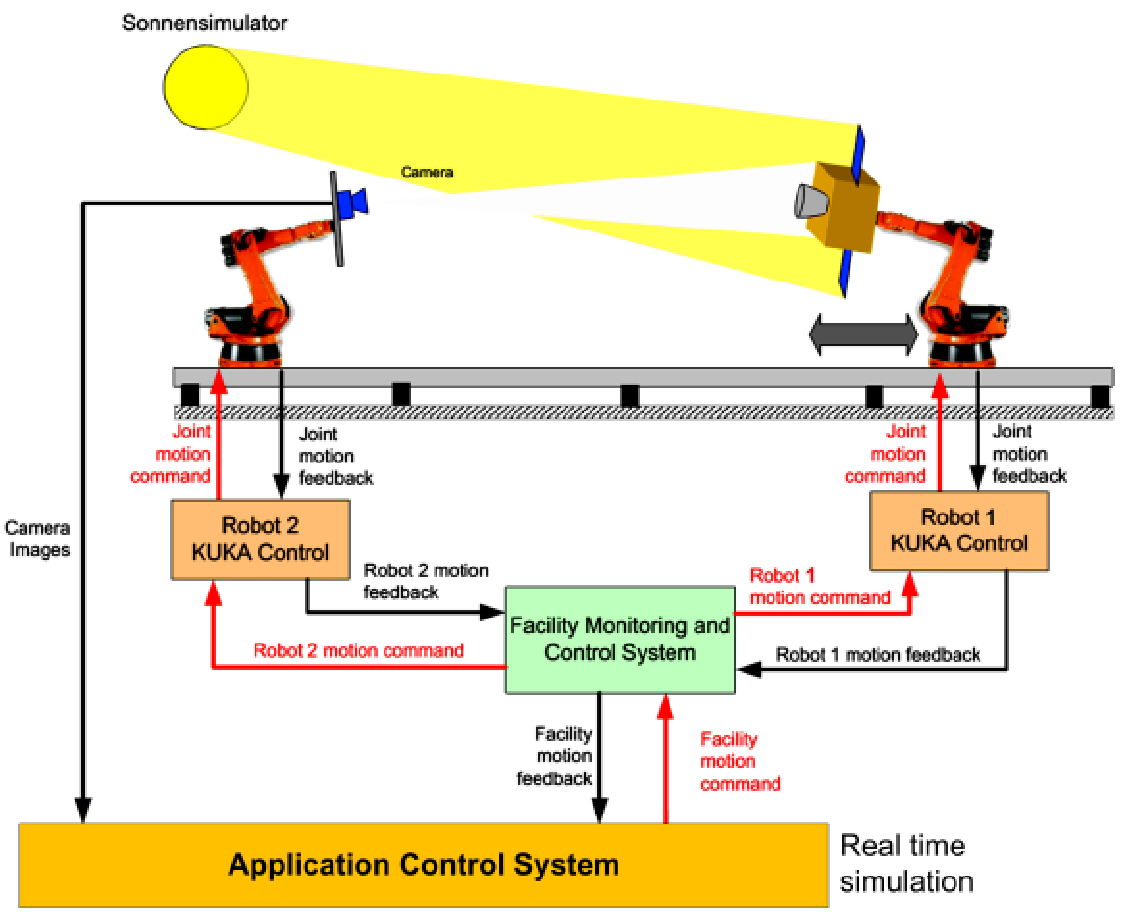


Figure 2.14 - EPOS configuration in a closed loop.

Source: Santos (2015).

This simulation facility can be used to demonstrate and test all outcomes obtained by computer environment simulations, such as those described here. Figure 2.15 presents a frame of the animated simulation performed by computer environment simulator designed by Nardin (2015).

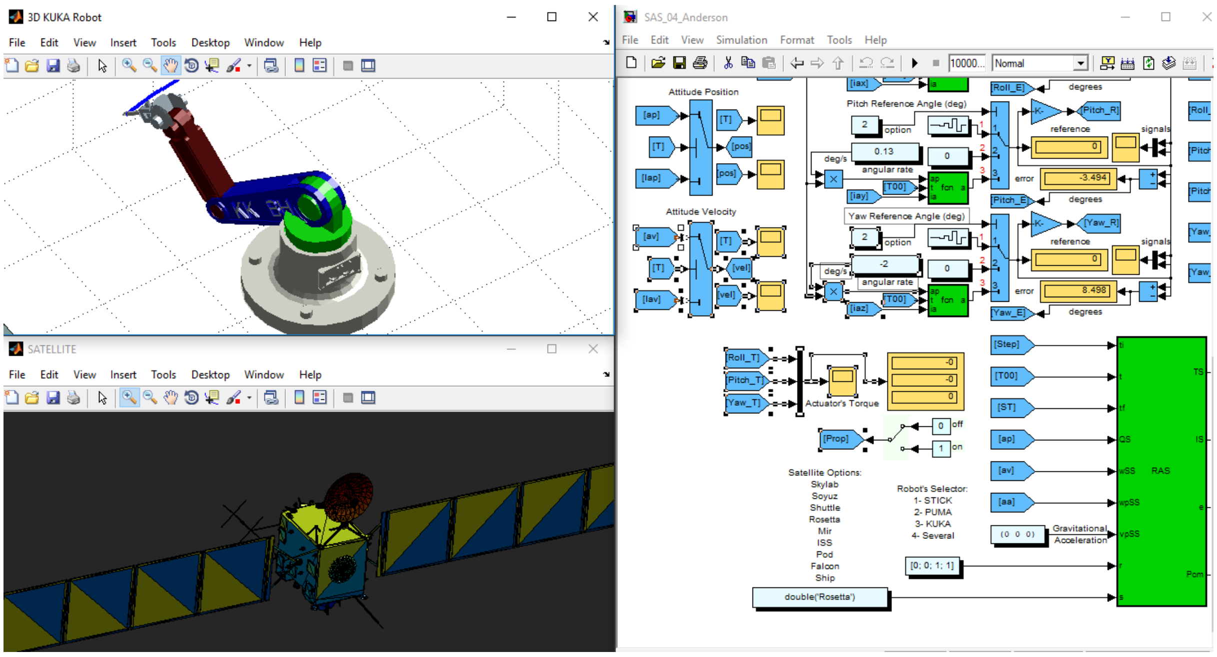


Figure 2.15 - Attitude Simulator fulfills a berthing.

2.4. Experiments

In the begging of the work through some meetings and after a general presentation describing my project to the On-Orbit Servicing and Autonomy group of the Space Flight Technology department at DLR, it was proposed a few modifications to be carried out in the software in order to improve chances of having it working correctly with the EPOS hardware. Indeed, those and many other modifications were done aiming to improve robustness and velocity. The proposed software was renamed Satellite Attitude and RObot Simulator – SAROS as an evolution of SAS (*Satellite Attitude Simulator*) developed by Rocco (2008), Rocco et al. (2011) and Rocco and Costa Filho (2015). Figure 2.16 shows the control system configuration.

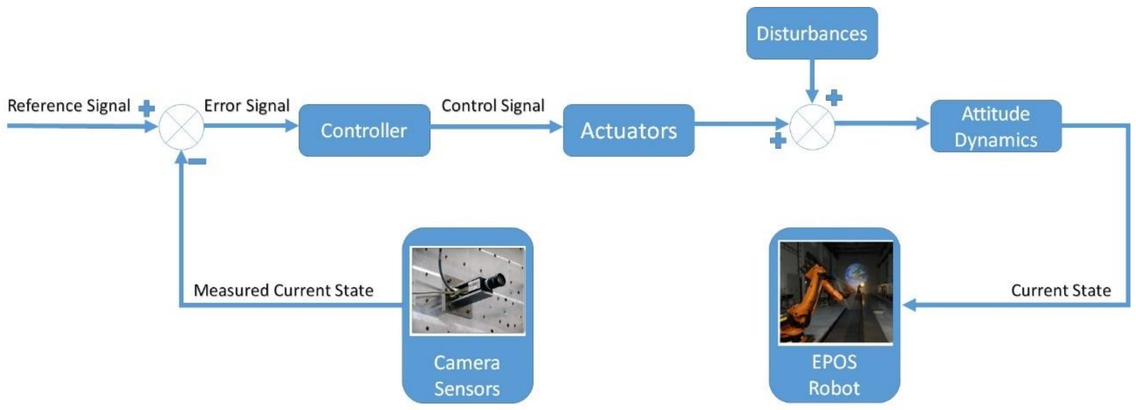


Figure 2.16 - Control System Configuration.

In order to be able to test the developed algorithms in the EPOS facility, it was composed a test routine. This plan of actions, when executed in the correct order, could improve results and at the same time avoid mistakes or unsafe actions.

The mentioned test plan was improved after some attempts to put it in practice. Each test was thought to be useful in some way to prove a software capability. After each test a document called “Test Protocol” formulated and proposed by Dr. Heike Benninghoff had to be properly filled in with a report format, see an example in Appendix A. In the end, we got the “Test Plan version 2.0”, which turned out to be worthy and consists of the steps showed in Appendix B.

Every software modification was properly documented using a tool for code sources management and software version control (*Git*), which has the intents: speed development, data integrity and support for distributed, non-linear workflows. The complete history of modifications fulfilled can be seen in Appendix C.

All robots’ movements in EPOS facility have been recorded through security cameras, Figure 2.17 shows the robot which performs the target satellite role in one of those frames from the recorded video and Figure 2.18 shows the robot responsible for chaser satellite’s movement.

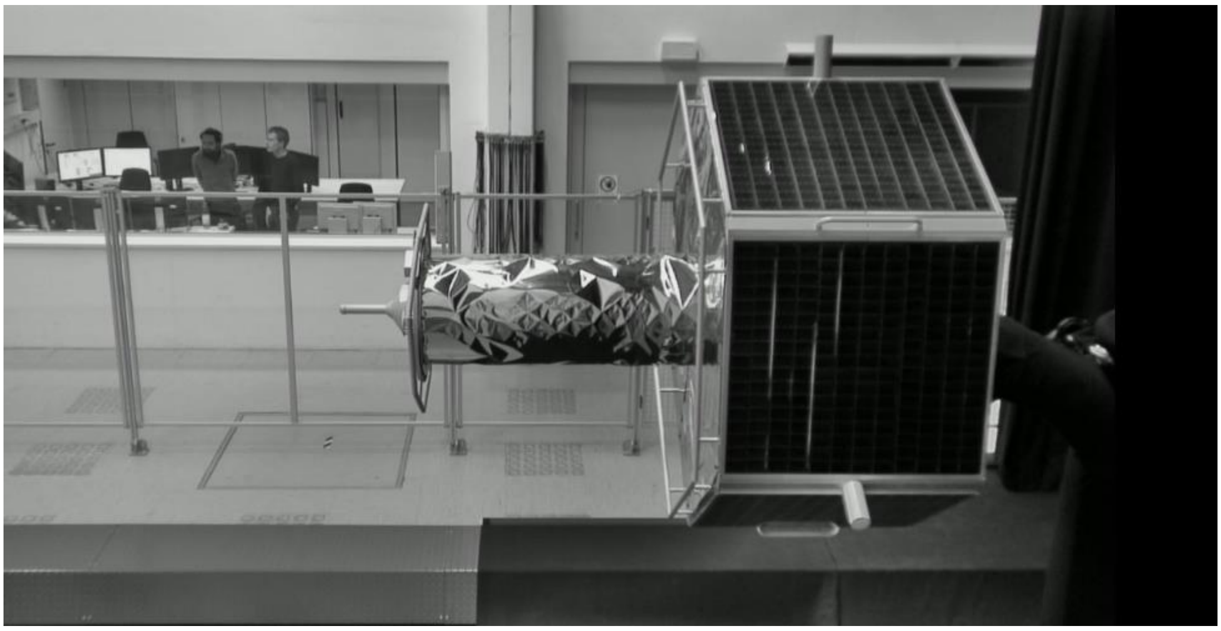


Figure 2.17 - Target Satellite and its robot.

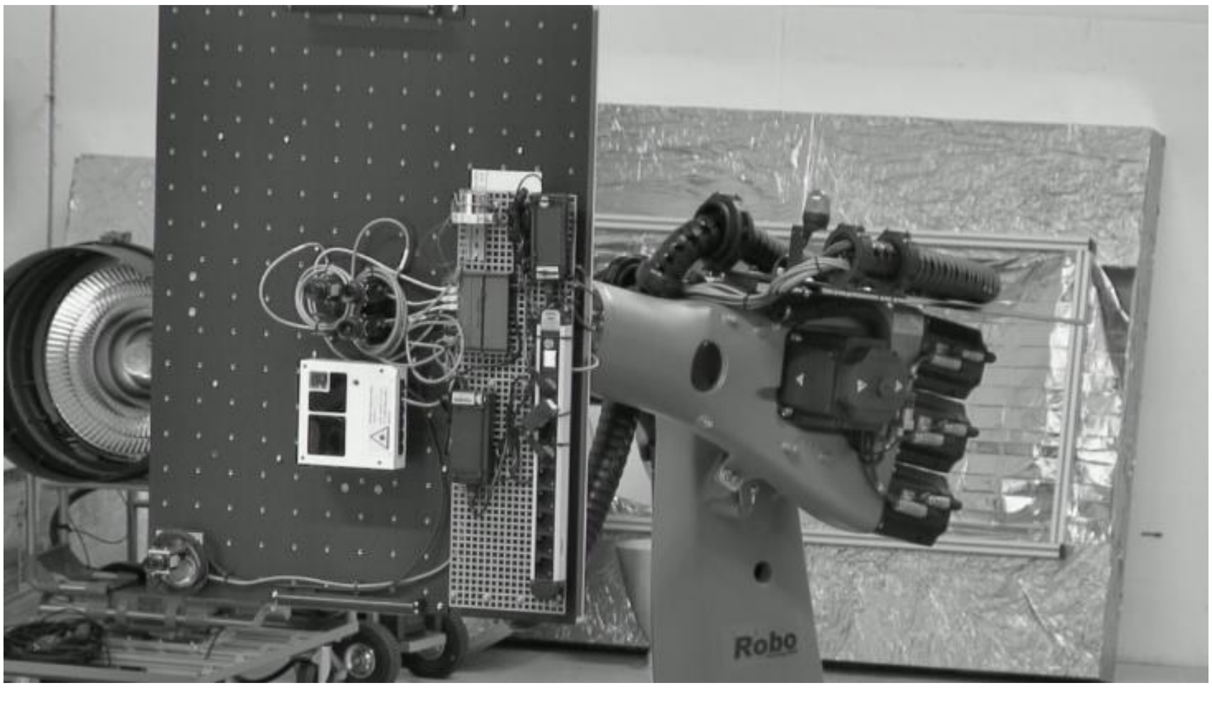


Figure 2.18 - Chaser Satellite and its robot.

3 SIMULATIONS AND RESULTS

It is necessary to say that the first two practical tests didn't work correctly due to problems originated in the source code and their results could not be used in this chapter. Despite those problems faced in the two first attempts, modifications were made, and we could test it again.

3.1. Simulation 1

Firstly, test 1 was done and as described in the Test Plan (Appendix B), it consists of maintaining the initial orientation with satellite control system and virtual manipulator control system turned off in order to demonstrate the possibility of connection between the developed system and devices of EPOS facility. We can verify the success of this test looking at the results generated when the collected data is put on a graph. Figure 3.1, 3.2 and 3.3 demonstrate how the chaser satellite orientation was maintained along all simulation.

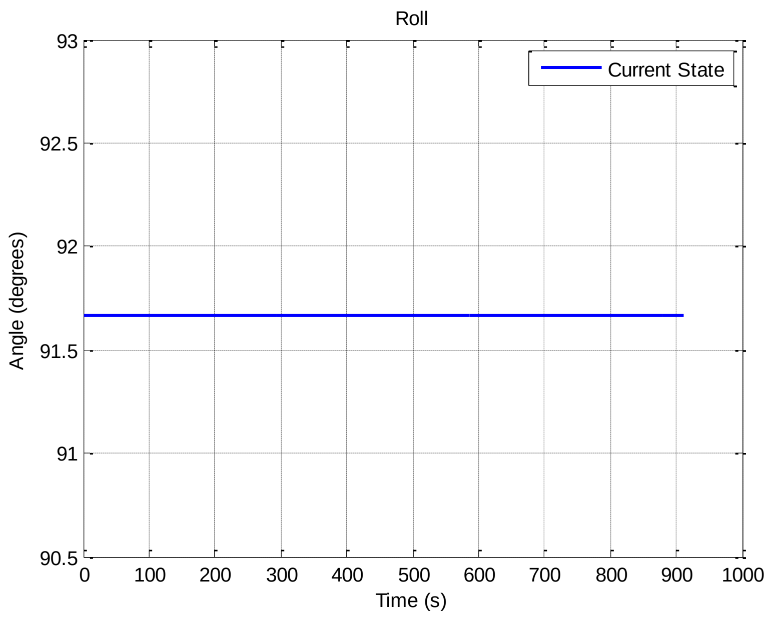


Figure 3.1 - Angle in roll (simulation 1).

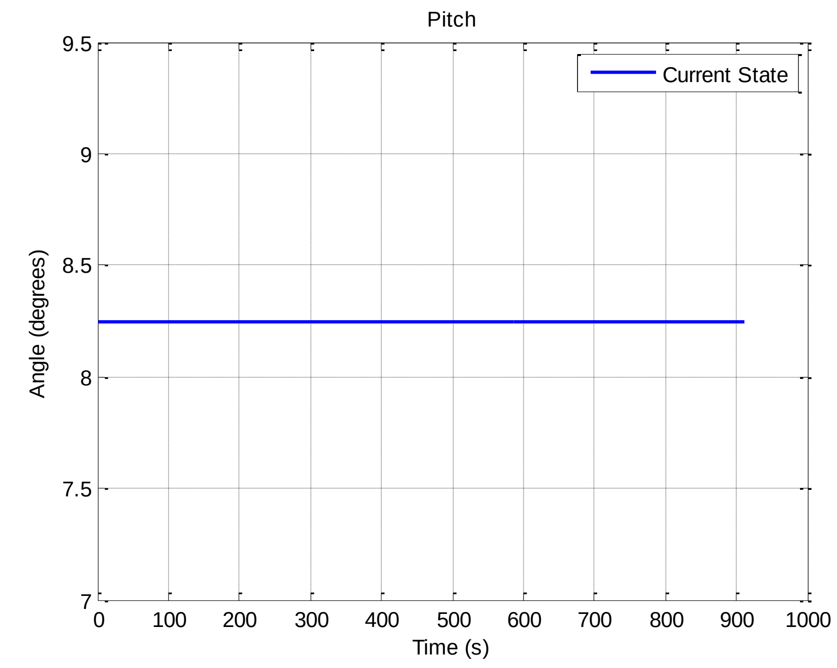


Figure 3.2 - Angle in pitch (simulation 1)

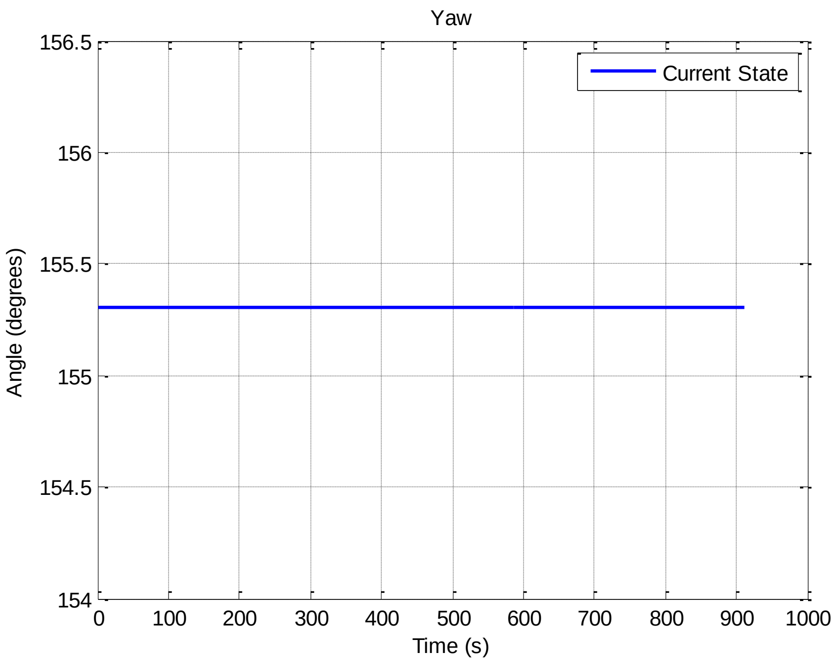


Figure 3.3 - Angle in yaw (simulation 1).

Besides maintaining the initial orientation, it was desirable that satellite and virtual manipulator control systems were maintained turned off as well. This demonstrates reliability when the software runs. We can verify this results in Figures 3.4 and 3.5.

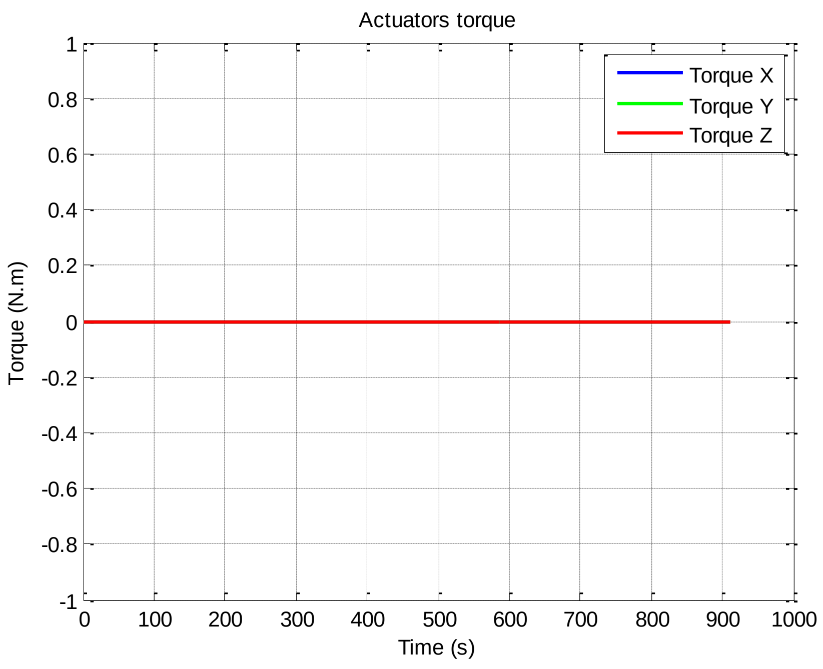


Figure 3.4 - Actuators torque (simulation 1)

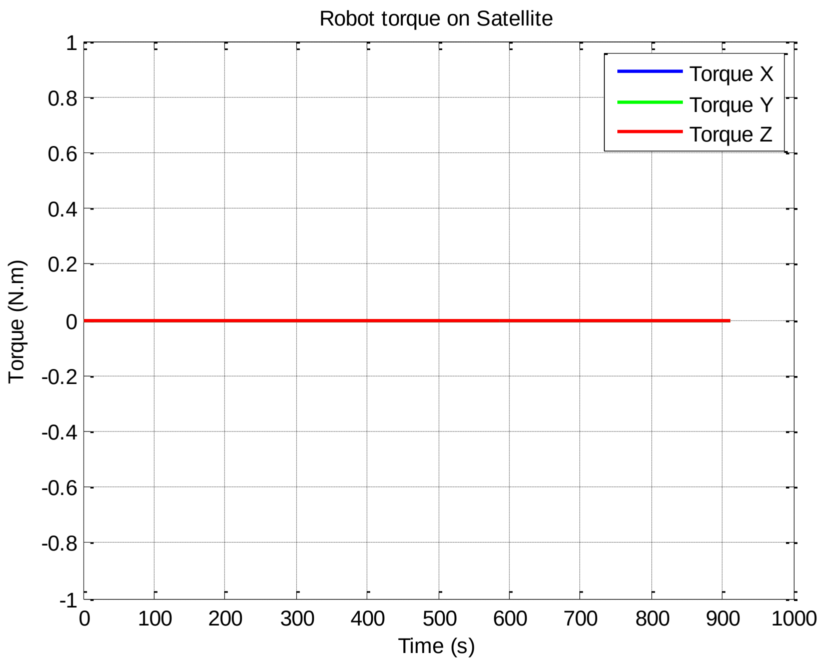


Figure 3.5 - Robot toque on Satellite (simulation 1).

3.2. Simulation 2

In the second simulation, our objective was to turn on the chaser satellite control system and then operate roll, pitch and yaw angles maneuver in order to show that the developed system can control the EPOS robots and emulate movements of the satellites. Figures 3.6, 3.7, 3.8 and 3.9 make us conclude this task has been properly executed.

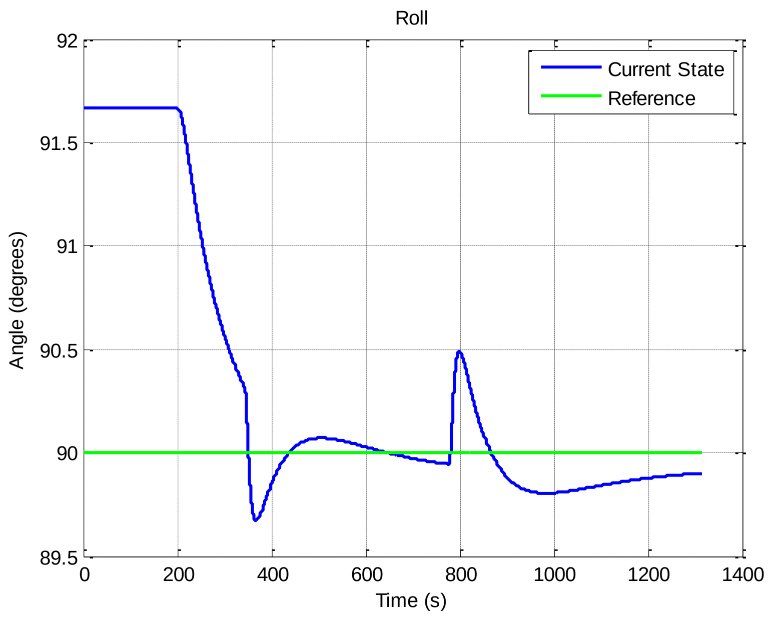


Figure 3.6 - Angle in roll (simulation 2).

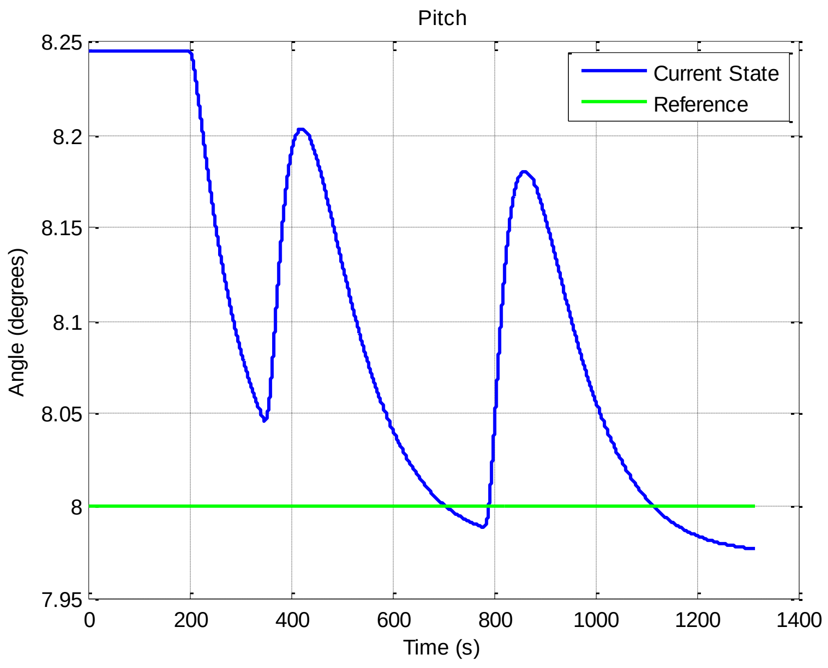


Figure 3.7 - Angle in pitch (simulation 2)

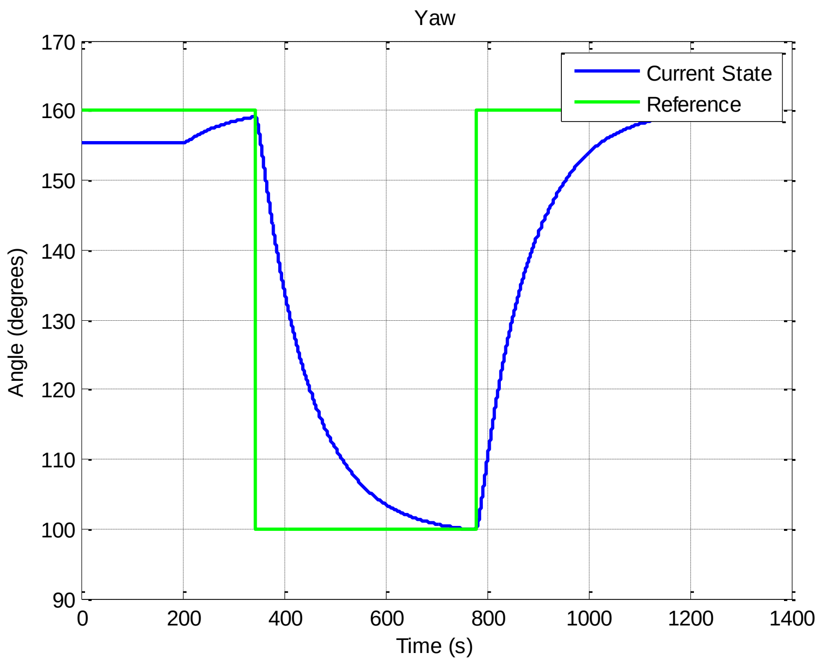


Figure 3.8 - Angle in yaw (simulation 2).

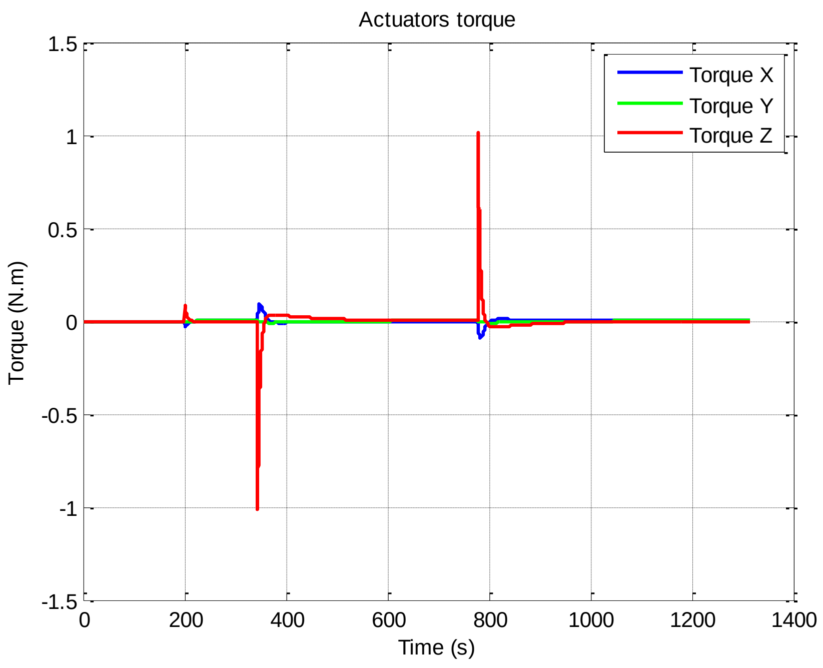


Figure 3.9 - Actuators torque (simulation 2)

Figure 3.10 shows us when the orientation control system was turned on and confirms that the robotic manipulator was maintained turned off.

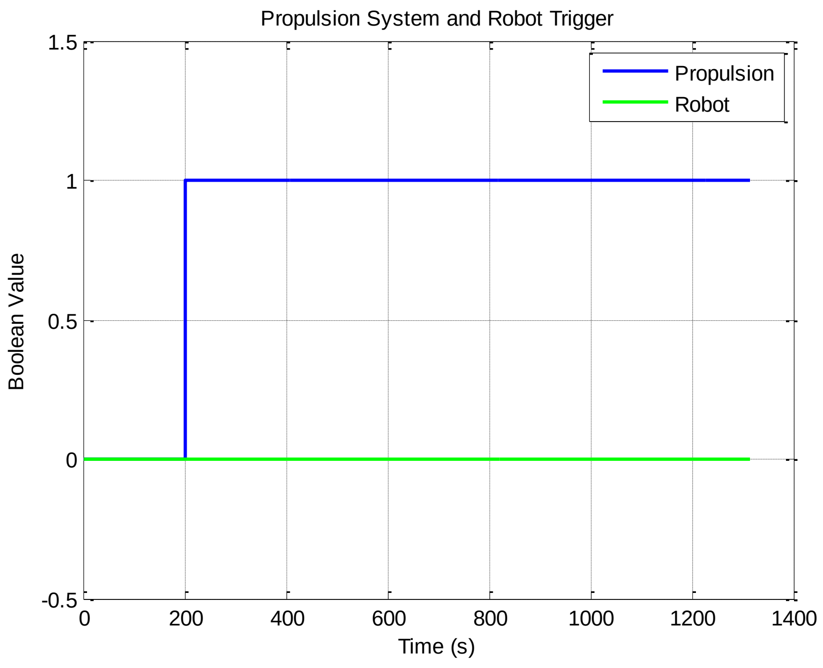


Figure 3.10 - Propulsion system and robot trigger (simulation 2)

3.3. Simulation 3

In this simulation, those tests 3,4 and 5 depicted in Test Plan (Appendix B) were executed. The general aim was to maintain the final orientation commanded to show the capacity of achieving an orientation by the satellite control system and its features as well. After this, turning off the satellite control system and turning on the manipulator control system. In this case, the manipulator shall take its end effector as close of the defined target vector as possible. In this case, the commanded target vector is out of the virtual manipulator workspace. And finally, turning on again the satellite's control system while the manipulator pursues the defined target vector. This test aims to show us how the virtual manipulator behaves in the scenario of cooperative motion between robot and satellite. Figure 3.11 shows us the order of actions.

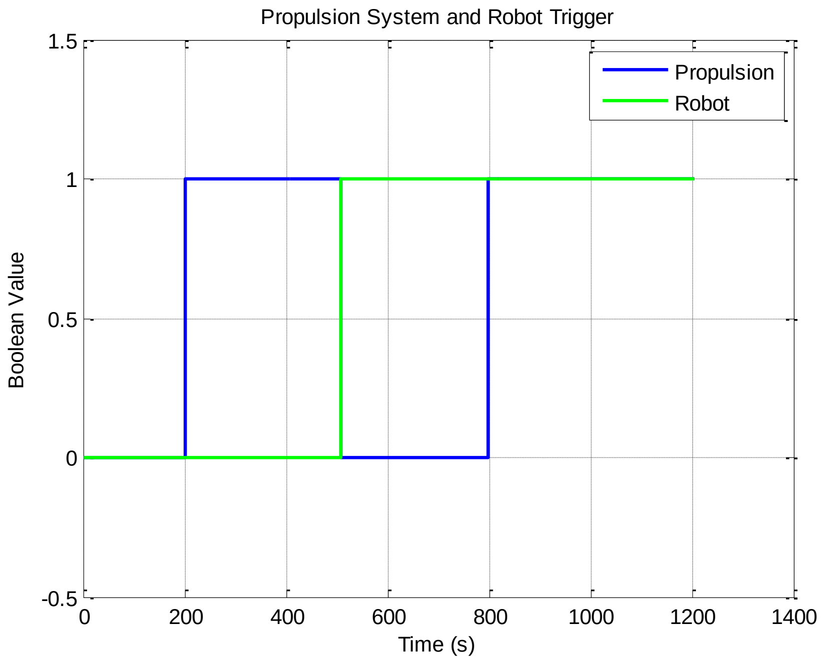


Figure 3.11 - Propulsion system and robot trigger (simulation 3)

As the satellite control system was turned on, we can see in Figures 3.12, 3.13, 3.14 and 3.15 its changes along time in each direction roll, pitch and yaw and its actuators torques respectively.

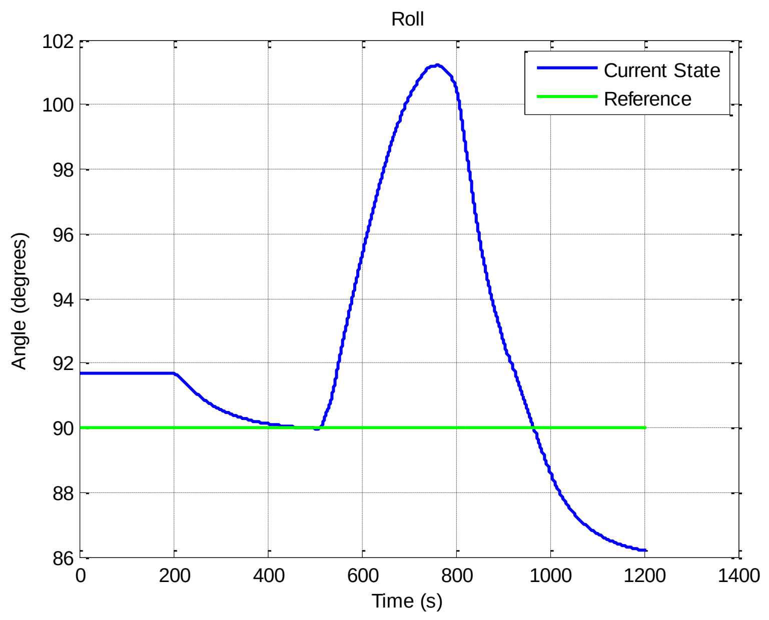


Figure 3.12 - Angle in roll (simulation 3).

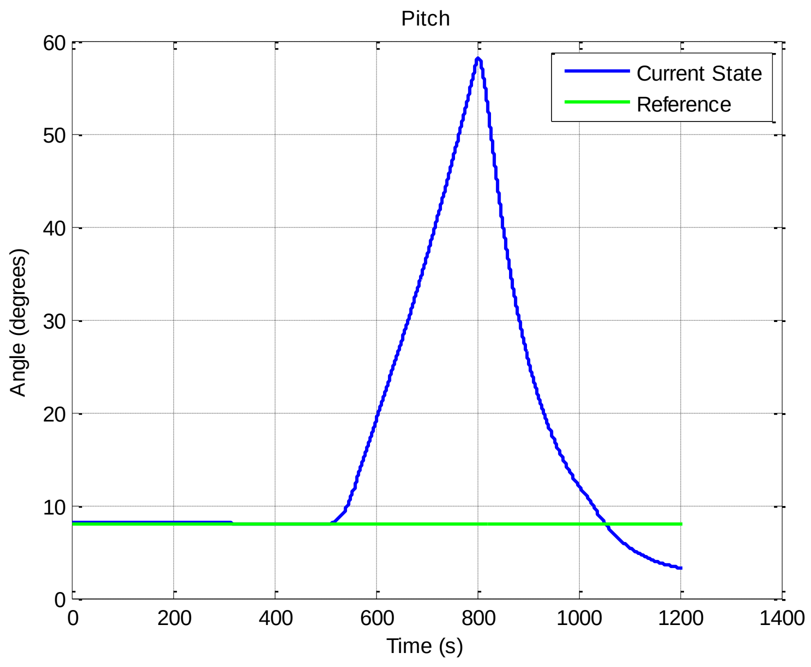


Figure 3.13 - Angle in pitch (simulation 3)

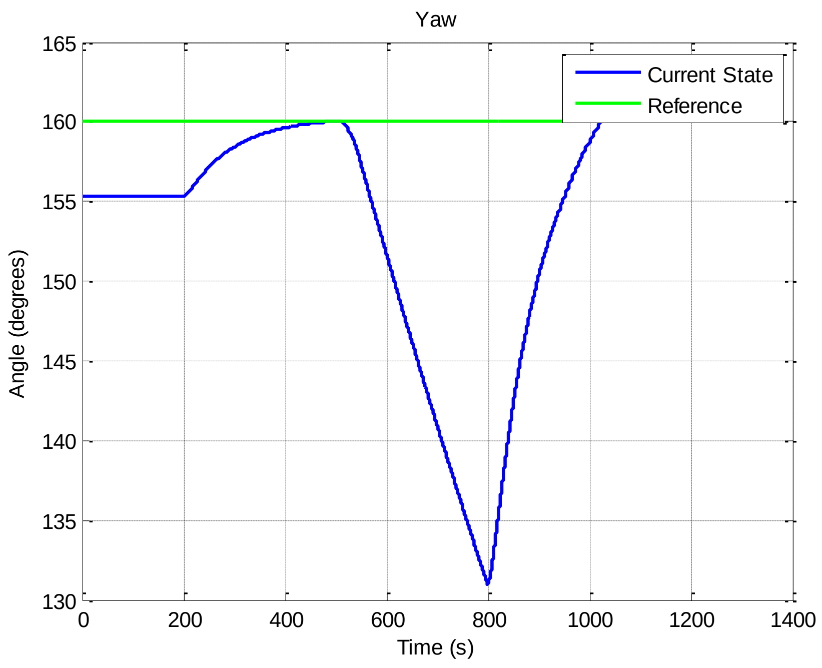


Figure 3.14 - Angle in yaw (simulation 3).

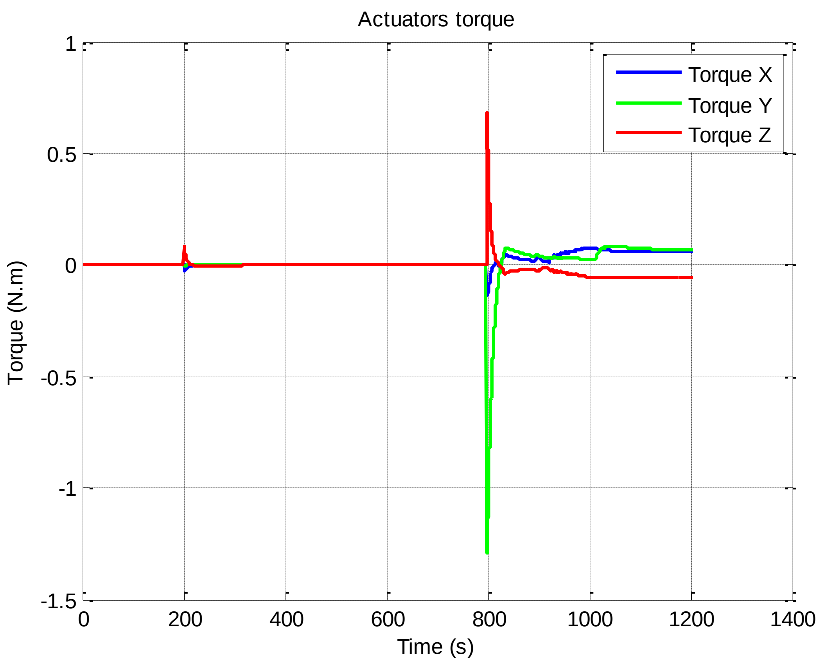


Figure 3.15 - Actuators torque (simulation 3)

As the virtual manipulator was trigged, Figures 3.16, 3.17 and 3.18 shows angular positions, velocities and accelerations of each robot's joints.

Figure 3.19 shows the torques which has been applied on the satellite base by the manipulator.

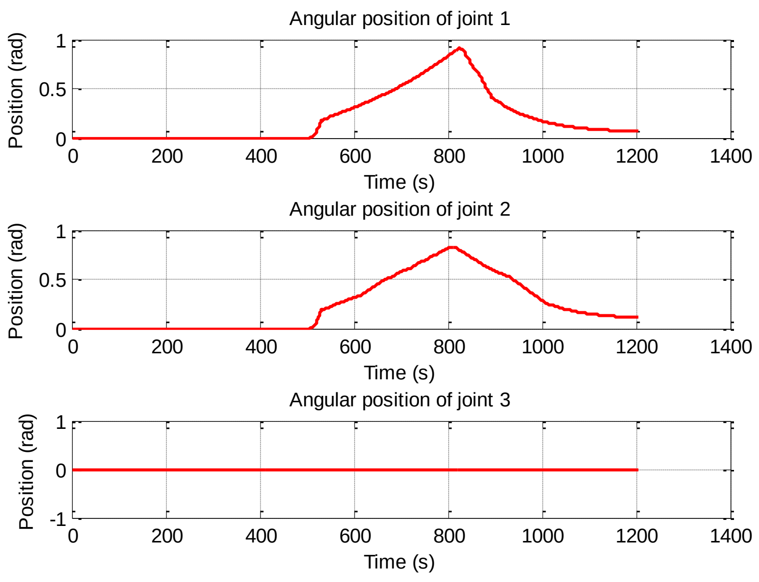


Figure 3.16 - Angular positions (simulation 3).

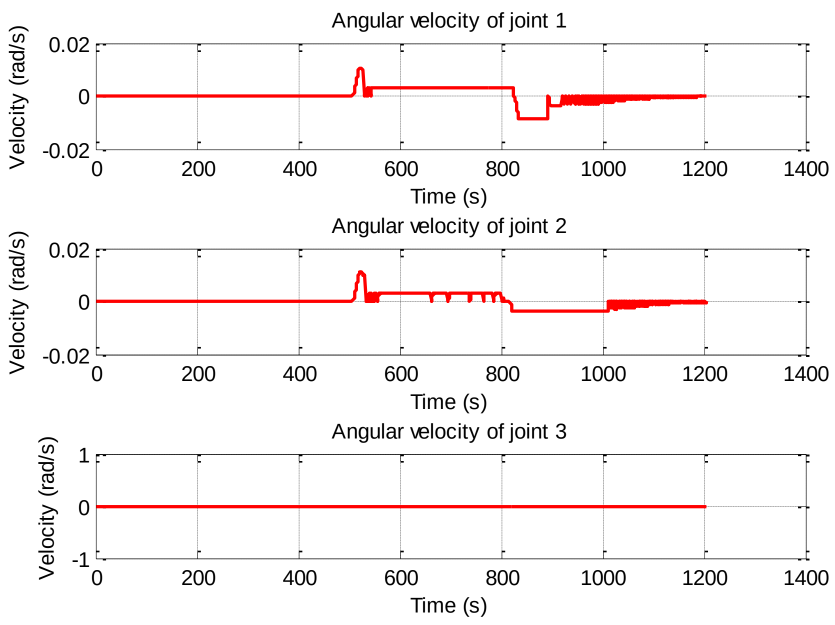


Figure 3.17 - Angular velocities (simulation 3)

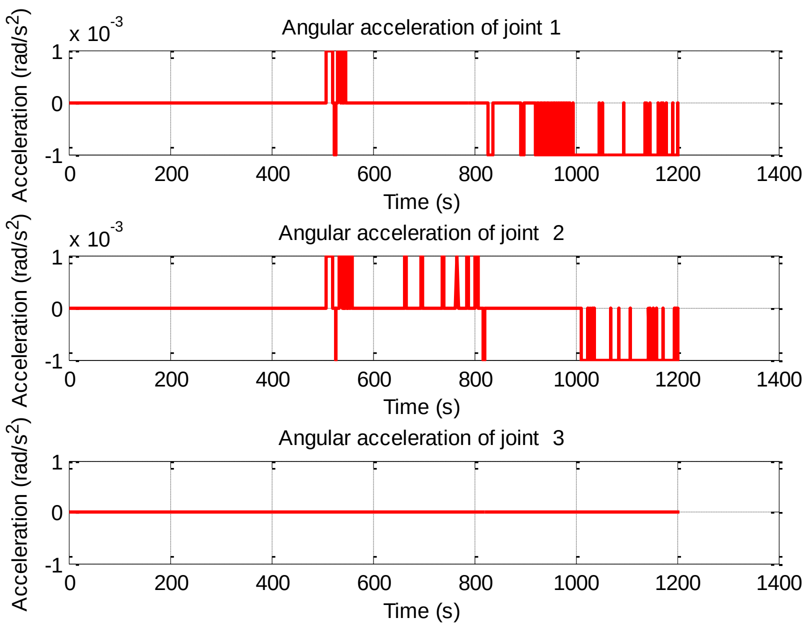


Figure 3.18 - Angular accelerations (simulation 3).

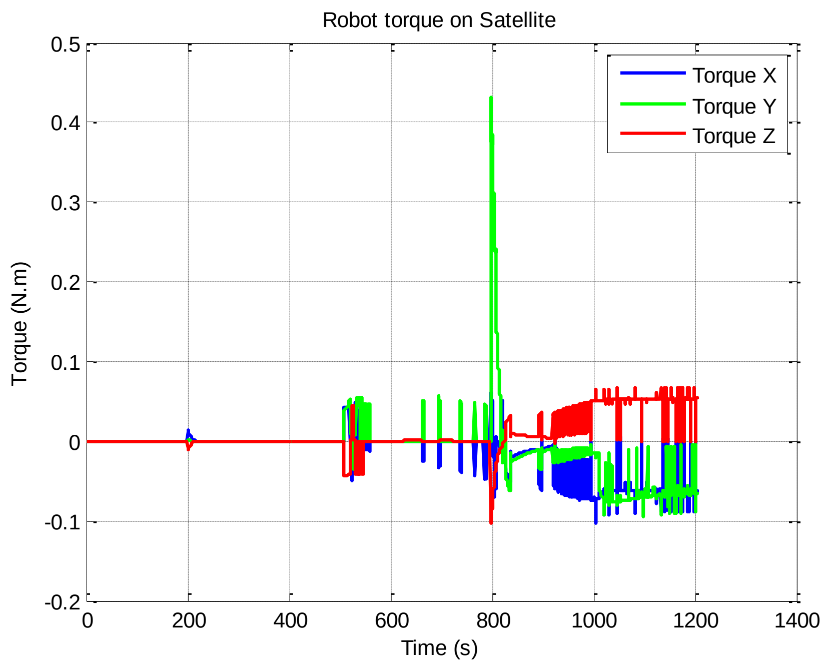


Figure 3.19 - Robot torque on satellite (simulation 3)

We can see it behaved exactly as it was expected in a situation where the target point is out of the defined workspace.

3.4. Simulation 4

In this simulation, the described test 6 is put into practice. The target vector is defined to be inside the manipulator workspace. With this simulation, we intend to demonstrate that the virtual manipulator control system is able to reach a point inside its workspace and maintain a given position. Figure 3.20 shows the order of actions.

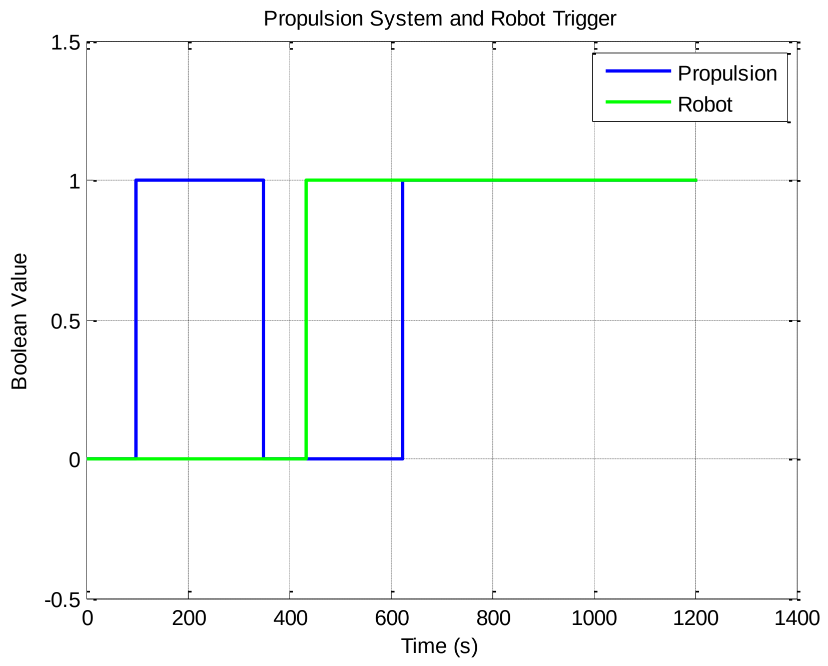


Figure 3.20 - Propulsion system and robot trigger (simulation 4)

We can see the satellite control system working with a sight from Figures 3.21, 3.22 and 3.23.

Figure 3.24 shows us how the moments of inertia of the arrangement, formed by satellite base plus robotic manipulator, changes along time, while Figure 3.25 shows the center of mass under the same point of view.

Figure 3.26 presents the torques applied on the satellite by the manipulator due to its movements.

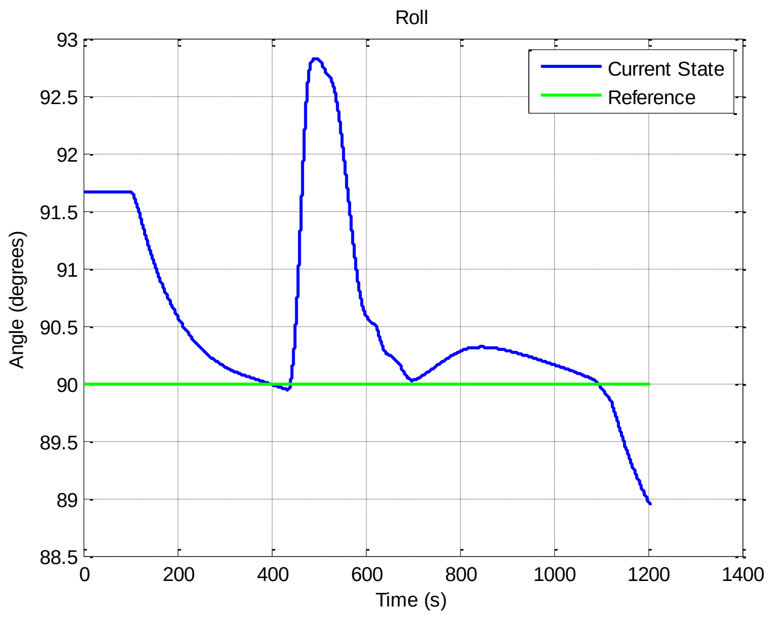


Figure 3.21 - Angle in roll (simulation 4).

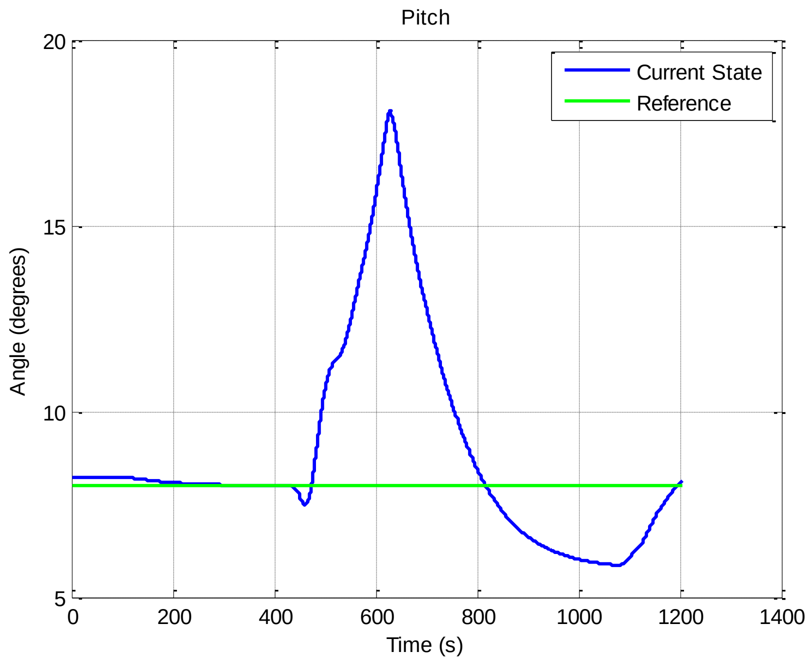


Figure 3.22 - Angle in pitch (simulation 4)

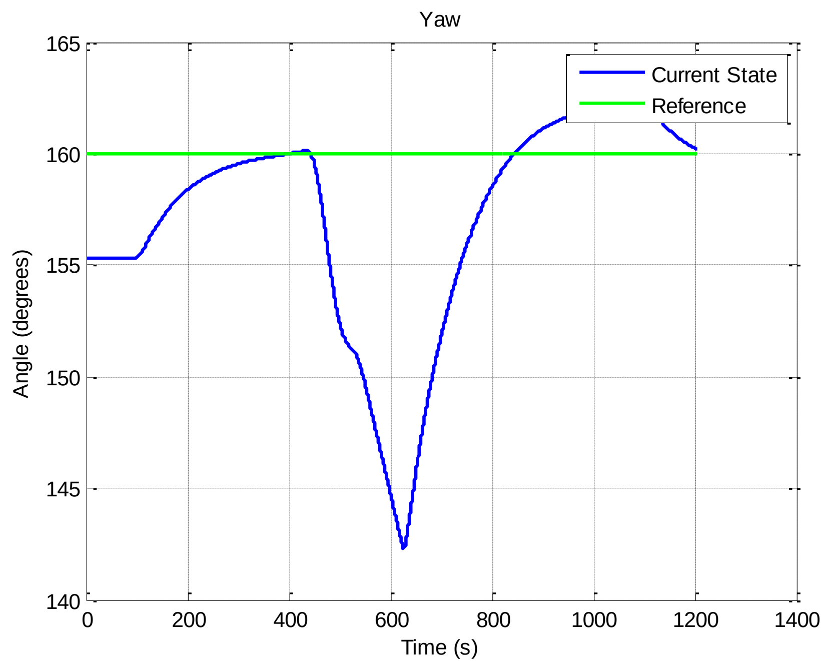


Figure 3.23 - Angle in yaw (simulation 4).

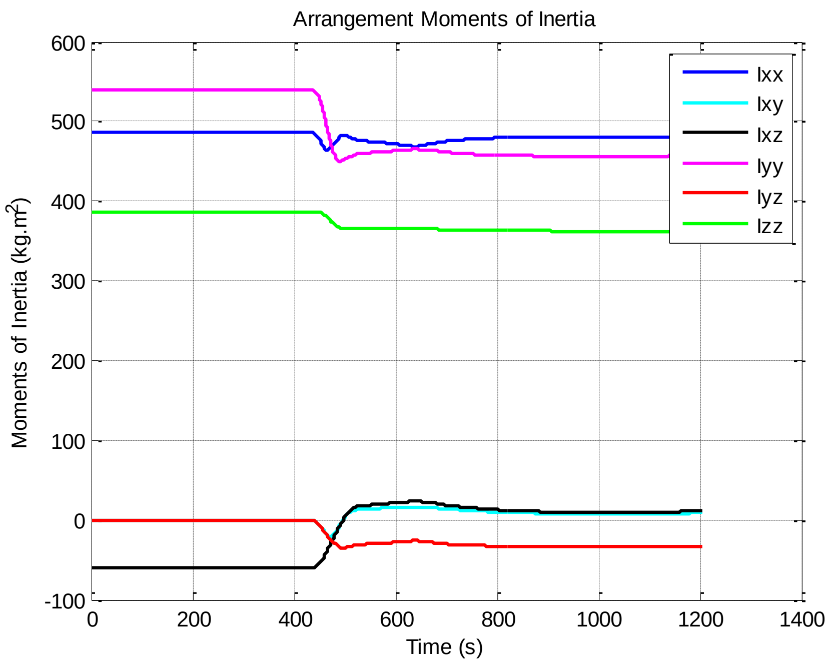


Figure 3.24 - Arrangement moments of inertia (simulation 4).

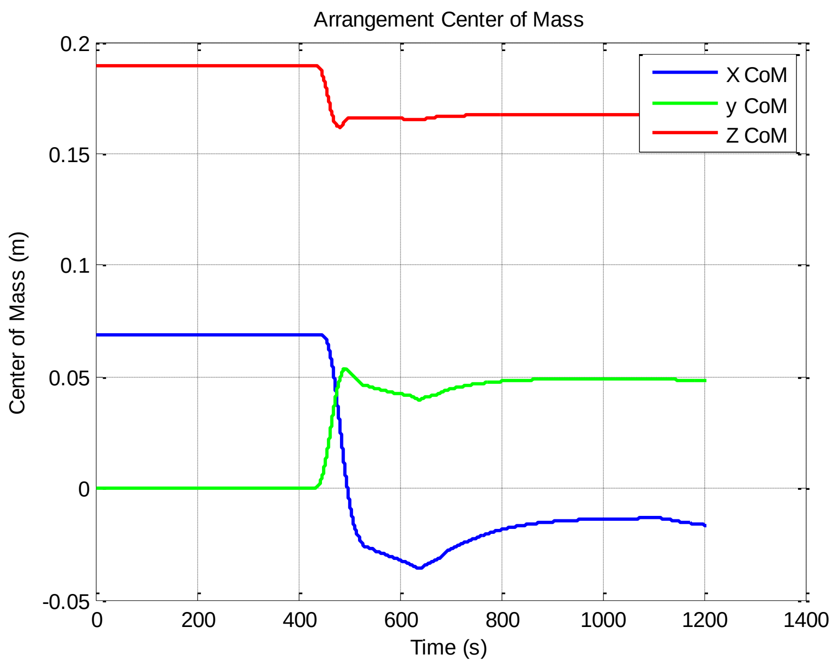


Figure 3.25 - Robot torque on satellite (simulation 4).

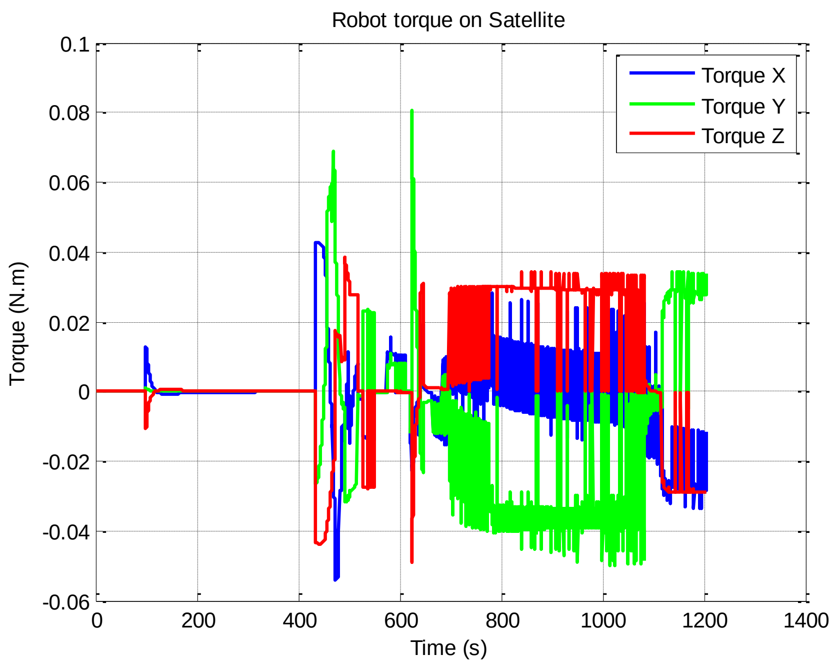


Figure 3.26 - Robot torque on satellite (simulation 4).

Finally, Figures 3.27 and 3.28 prove that the manipulator was able to achieve the target point since the distance error tended to zero and joints had velocity zero.

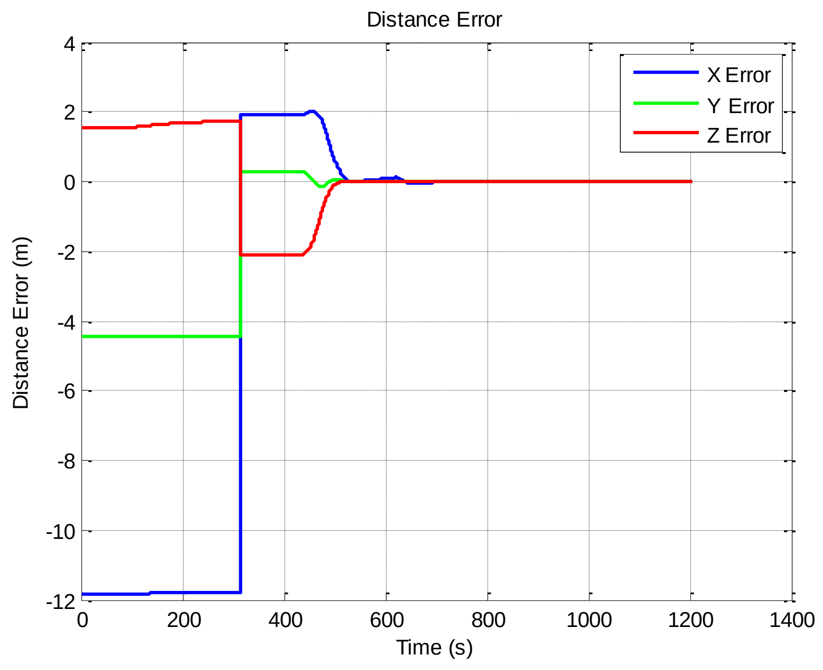


Figure 3.27 - Distance error to the target (simulation 4).

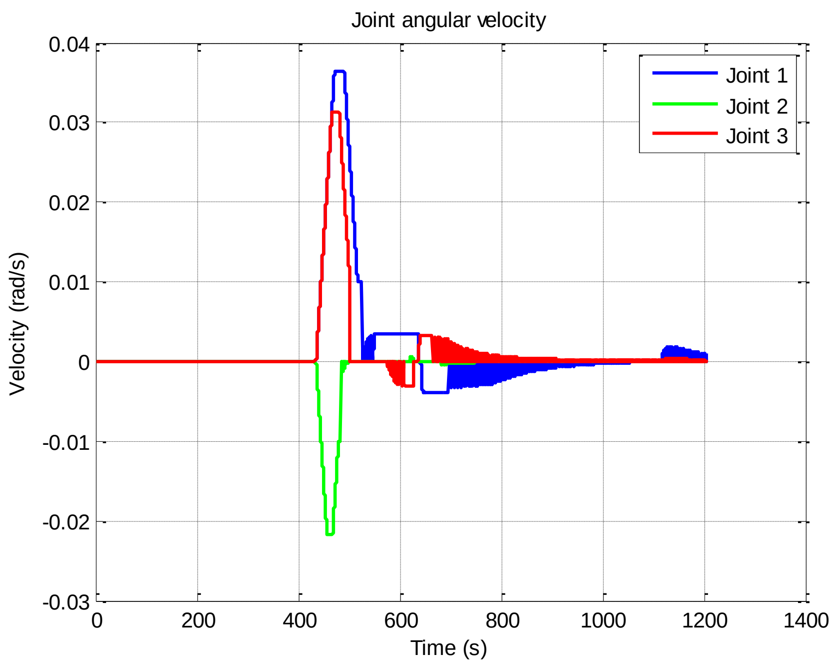


Figure 3.28 - Joint angular velocities (simulation 4).

3.5. Simulation 5

This was the most complete simulation since all steps and previous tests were performed in sequence in order to prove the robustness and reliability of the project. Figure 3.29 shows the triggering history between the virtual robot and the propulsion system which enables the satellite control system.

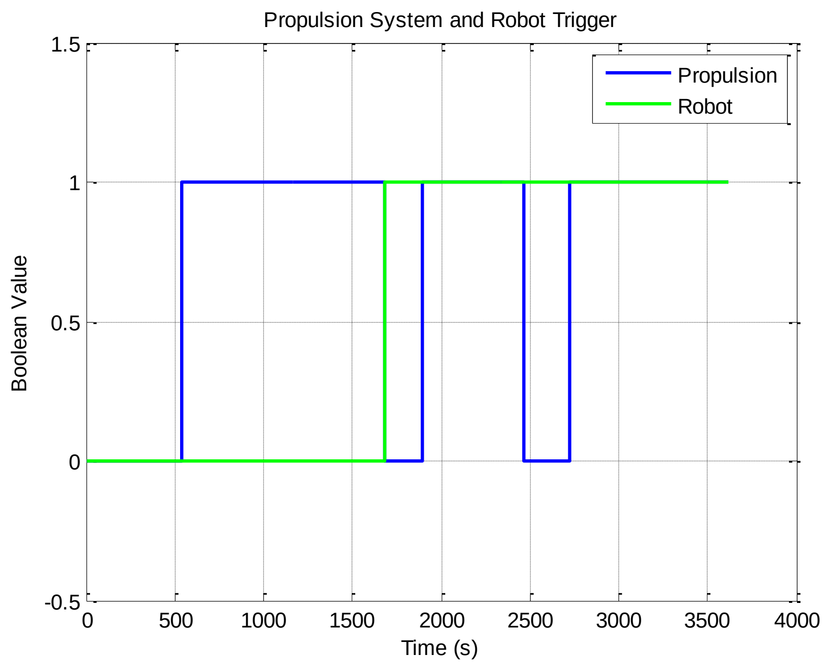


Figure 3.29 - Propulsion system and robot trigger (simulation 5)

In order to prove the target point was reached by the virtual manipulator end effector, we can see in Figure 3.30 the distance error tending to zero. That means the algorithm worked correctly, this can be confirmed by Figure 3.31 where the joints velocities are presented and at some point, all joints had null velocity at the same time.

In order to evaluate the satellite control system, Figure 3.32, 3,33 and 3,34 show its behavior along time due to multiple triggers. Figure 3.35 presents in a tridimensional space how the arrangement center of mass has changed along the simulation.

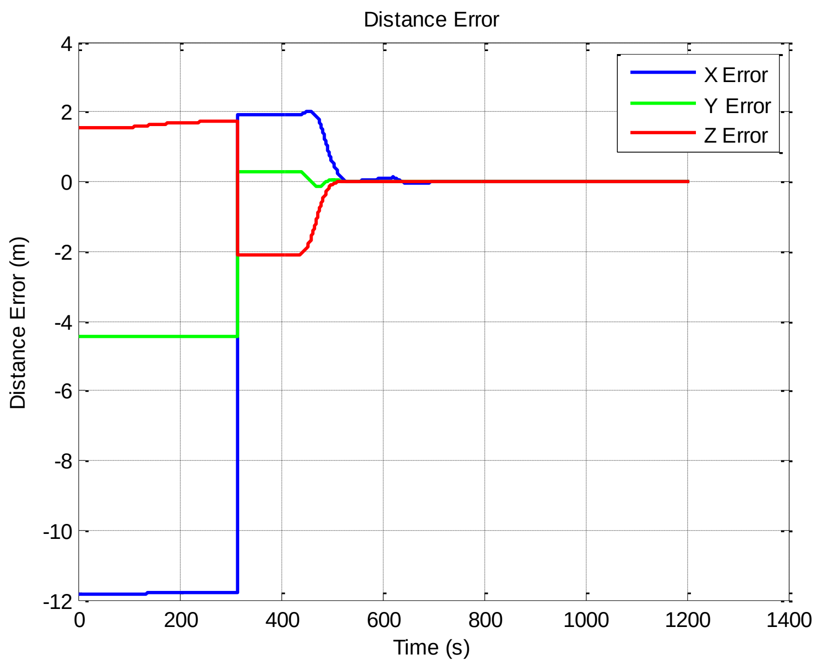


Figure 3.30 - Distance error to the target (simulation 5).

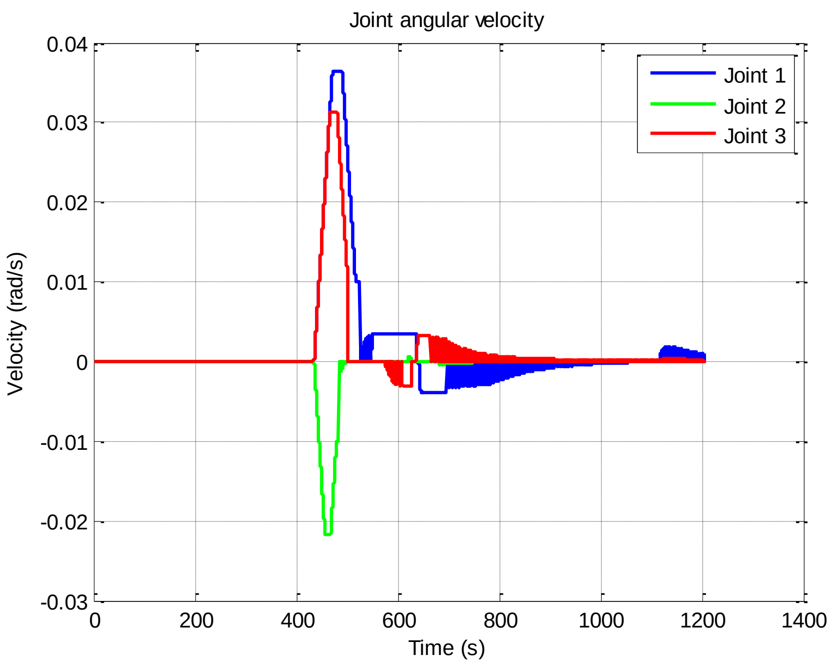


Figure 3.31 - Joint angular velocities (simulation 5).

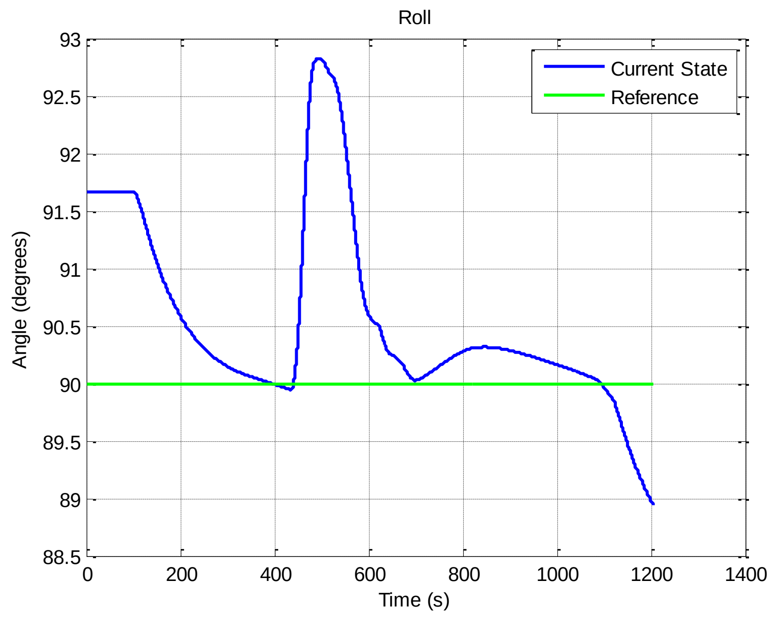


Figure 3.32 - Angle in roll (simulation 5).

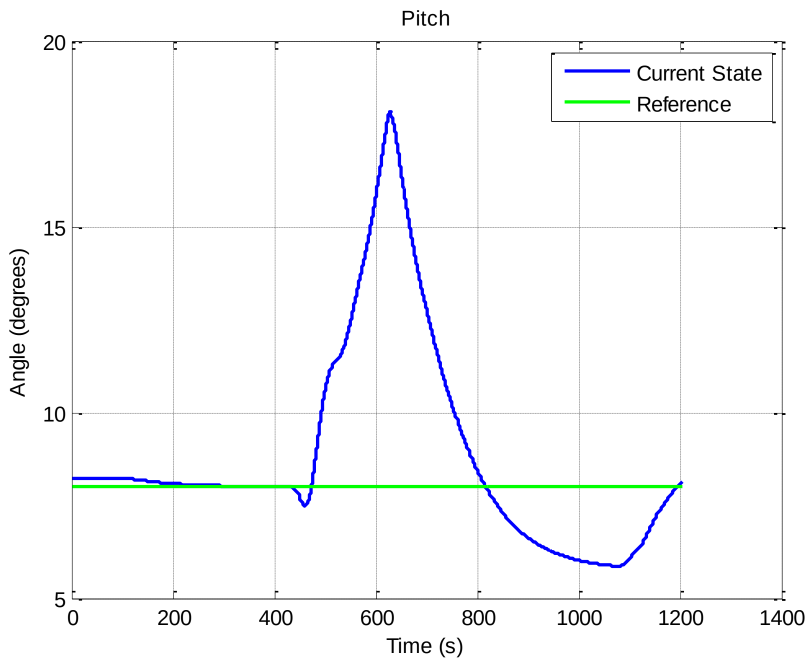


Figure 3.33 - Angle in pitch (simulation 5)

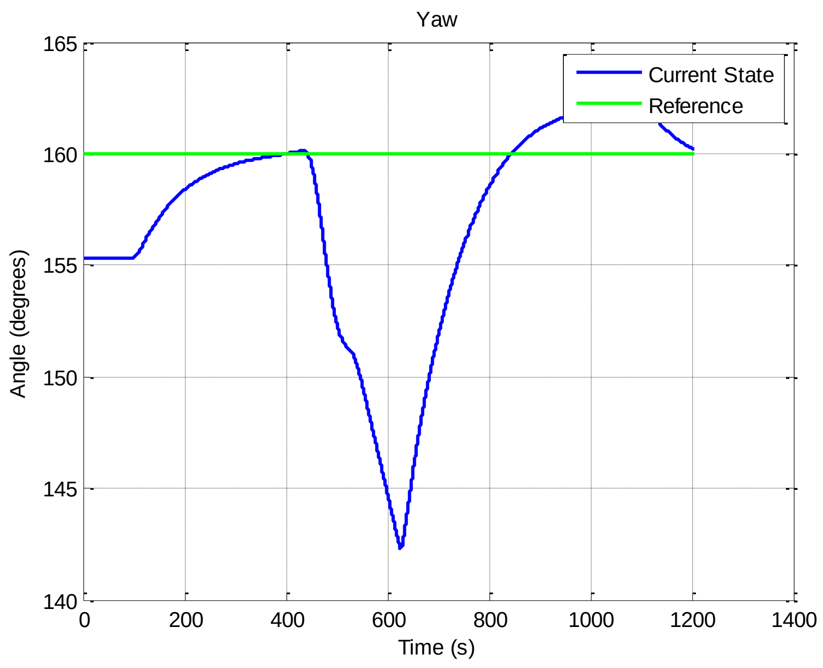


Figure 3.34 - Angle in yaw (simulation 5).

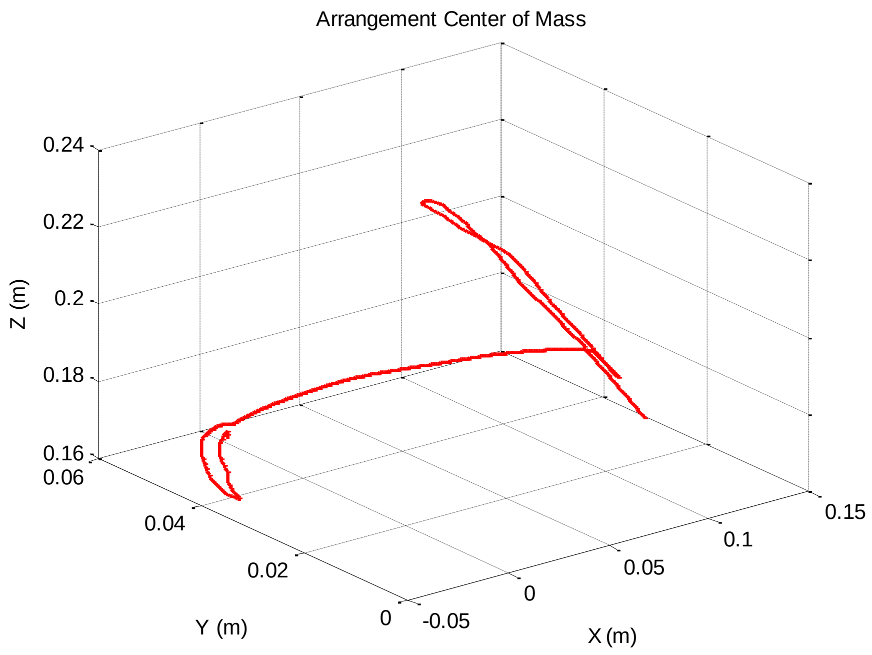


Figure 3.35 - Arrangement center of mass in 3D (simulation 5).

4 CONCLUSION

Along all internship, it was notable the value of On-Orbit activities and their impact for future missions. In this scenario, it is remarkable that laboratories which turn possible to test and verify results previously are fundamental tools for such critical and dangerous missions. This way, a reliable software environment where new ideas and planned routines can be performed is very useful, especially when it is able to work together with proper hardware exploiting the HIL concept.

The experiments described were useful to prove the validity of the developed algorithms and inspired the creation of a concept of “Phantom Limb” for spacecrafts endowed with robotic manipulators when tested in these circumstances since at some point we could see the robot, which played the role of a chaser satellite, moving despite its propulsion and control system was turned off. We can conclude that such movement was due to the virtual manipulator, which was not there physically, displacement along time.

In other words, the hardware was reacting to actions generated by a virtual part, emulated by software, and the total behavior was such as previously calculated by simulations executed only by software for the same inputs of target position. After all tests and simulations described, we can say this work achieved complete success in the task of creating a reliable software environment for tests of berthing maneuvers to be executed together with EPOS robots at DLR.

In the future, it would be interesting a study that proposes a multi-objective optimization approach, like in (ROCCO, 2002) and (ROCCO; E SOUZA; DE ALMEIDA PRADO, 2013), aiming to find a balanced solution among conflicting objectives, i.e., which satisfies a balanced request established among a set of conflicting objectives. Energy consumption, maneuver time, and movement precision would be treated as performance indexes to be optimized. Despite the difficulty of simultaneous optimization that they impose as objectives, this approach is necessary given the importance that they represent to artificial satellite maneuvers.

BIBLIOGRAPHY

BENNINGHOFF, Heike et al. End-to-end simulation and verification of GNC and robotic systems considering both space segment and ground segment. **Ceas Space Journal**, [s.l.], p.1-19, 23 jan. 2018. Springer Nature. http://dx.doi.org/10.1007/s12567-017-0192-2.

CRAIG, J.J. **Introduction to robotics**: mechanics and control. 3. ed. Upper Saddle River: Pearson Education Inc, 2005. 400 p.

ELLERY. A. **An introduction to space robotics**. Chichester. UK: Springer-Praxis Publishers, 2000. 672 p. ISBN (978-1-85233-164-1).

HUGHES, P. C. **Spacecraft attitude dynamics**. Mineola: Dover Publications, 1986. 564 p.

NARDIN, A. B. **Análise de manobras de atracação de satélites dotados de manipuladores robóticos.** 2015. 181 p. IBI: <8JMKD3MGP3W34P/3J259P2>. (sid.inpe.br/mtc-m21b/2015/02.18.23.42-TDI). Dissertation (Master of Science in Space Mechanics and Control) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2015. Available in: <http://urlib.net/8JMKD3MGP3W34P/3J259P2>.

ROCCO, E. M. **Manutenção orbital de constelações simétricas de satélites utilizando manobras impulsivas ótimas com vínculo de tempo**. 2002. IBI: <6qtX3pFwXQZ3r59YCT/H3MrR>. Tese (Doutorado em Mecânica Espacial e Controle) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2002.

ROCCO, E. M. Perturbed orbital motion with a PID control system for the trajectory. In: COLÓQUIO BRASILEIRO DE DINÂMICA ORBITAL, 14., 2008, Águas de Lindóia. **Anais...** Águas de Lindóia: ABCM, 2008.

ROCCO, E. M.; COSTA FILHO, A. C; CARRARA, V. Effect of the coupling between attitude and orbital control in maneuvers using continuous thrust.

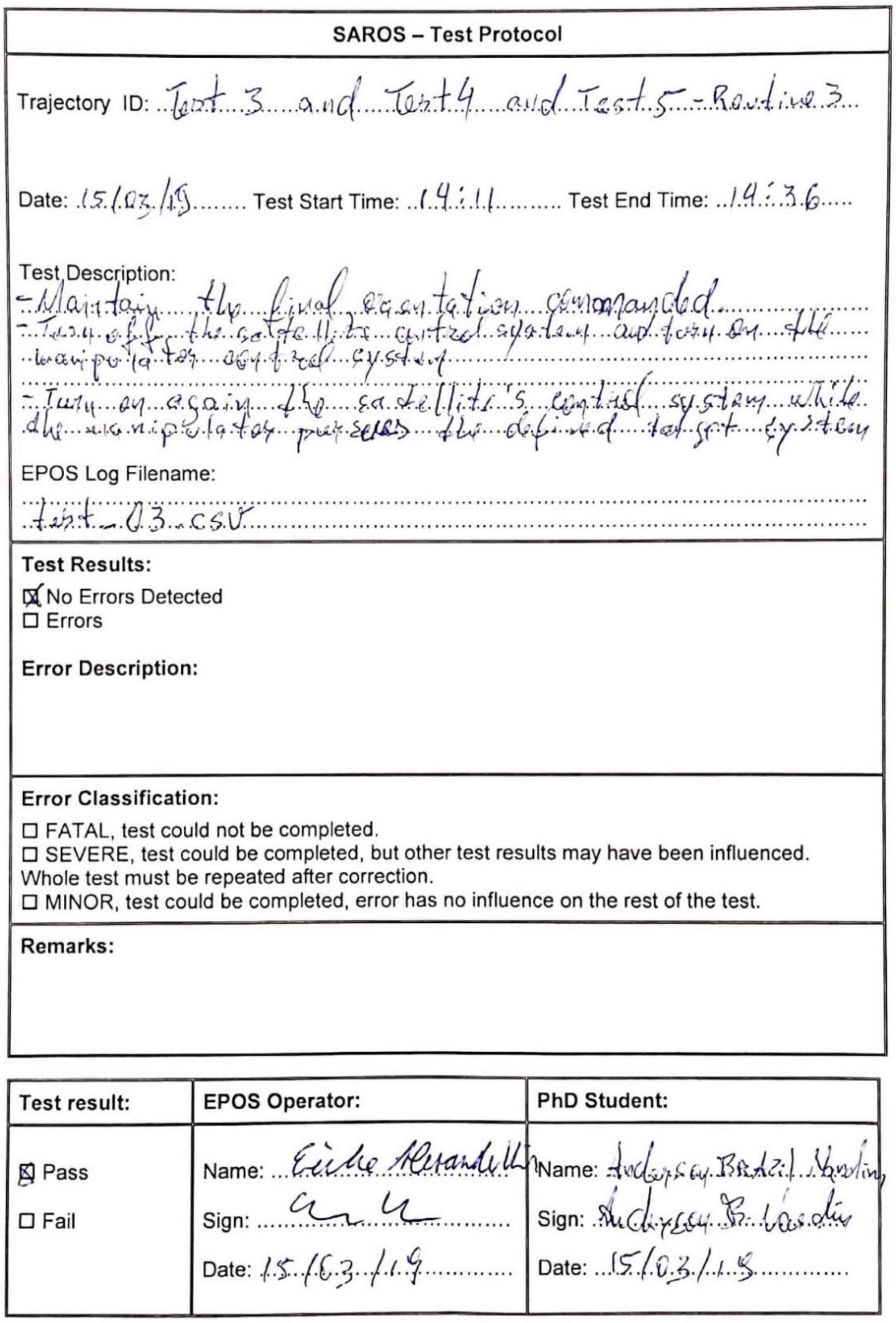
Minissimpósio: Aerospace Engineering. In: CONFERÊNCIA BRASILEIRA DE DINÂMICA, CONTROLE E APLICAÇÕES, 10., 2011, Águas de Lindóia. **Anais...** Águas de Lindóia: [s.n.], 2011.

ROCCO, E. M.; E SOUZA, M. L. DE O.; DE ALMEIDA PRADO, A. F. B. Station Keeping of Constellations Using Multiobjective Strategies. **Mathematical Problems in Engineering**, v. 2013, p. 1–15, 2013.

ROCCO, E. M.; COSTA FILHO, A. C. DA. **Avaliação dos desvios na trajetória originados pelo acoplamento entre o controle de atitude e de órbita em manobras orbitais com propulsão contínua**. . In: XXXV CNMAC - CONGRESSO NACIONAL DE MATEMÁTICA APLICADA E COMPUTACIONAL. 25 ago. 2015. Available in: <https://proceedings.sbmac.org.br/sbmac/article/view/585>.

SANTOS, W. G. **Discrete multi-objective optimization applied to the spacecraft actuators command problem and tested in a hardware-in-the-loop rendezvous simulator**. 2015. 160 p. IBI: <8JMKD3MGP3W34P/3HRTNES>. (sid.inpe.br/mtc-m21b/2015/01.30.12.04-TDI). Thesis (Doctorate in Space Engineering and Control) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2015. Available in: <http://urlib.net/8JMKD3MGP3W34P/3HRTNES>.

A SAROS - TEST PROTOCOL



SAROS – Test Protocol

Trajectory ID: Test 3 and Test 4 and Test 5 - Routine 3

Date: 15/03/19 Test Start Time: 14:11 Test End Time: 14:36

Test Description:
- Maintain the final orientation commanded
- Turn off the satellite's control system and turn on the manipulator control system
- Turn on again the satellite's control system while the manipulator pursues the defined target system

EPOS Log Filename:
test_03.csv

Test Results:
[✓ No Errors Detected]
☐ Errors

Error Description:

Error Classification:
☐ FATAL, test could not be completed.
☐ SEVERE, test could be completed, but other test results may have been influenced. Whole test must be repeated after correction.
☐ MINOR, test could be completed, error has no influence on the rest of the test.

Remarks:

Test result:	EPOS Operator:	PhD Student:
[✓ Pass] ☐ Fail	Name: Eike Alexander [illegible] Sign: [illegible] Date: 15/03/19	Name: Andersson Brazil Nardin Sign: [illegible] Date: 15/03/19

Figure A.1 - SAROS - Test Protocol.

B TEST PLAN VERSION 2.0

Initialization procedure:

Step 1: git fetch and git checkout my_feature it makes the local repository up to date with my newest modifications.

Step 2: click on init folder and run extepos_ref_init_e2e_local_scenario_1.m.

Step 3: change the current folder in Matlab environment to extepos_ref (where the file RAS.m can be found).

Step 4: Certify that the Real-Time PC is ready (Heike see it in the console COM2 in FCS - Facility Control System).

Step 5: click on the model and press Control+B in order to build the program.

Step 6: click on connect to target to establish connection with the compiled version of the program.

Step 7: Start the model.

Step 8: the model “ExtEPOS Monitor Displays” confirms if it is engaged. It means the commanded orientation was achieved by the robots’ facility.

Observation 1: before starting and stopping each test, inform Heike to Start/Stop data logging.

Test 1:

Maintaining the initial orientation with satellite control system and virtual manipulator control system turned off.

-This test demonstrates the possibility of connection between the developed system and devices of the EPOS facility.

Test 2:

Turn on the chaser satellite control system and then operate roll, pitch and yaw angle maneuver.

-this test shows that the developed system is able to control the EPOS robots and emulate movements of the satellites.

Test 3:

Maintain the final orientation commanded.

-this procedure shows the capacity of achieving an orientation by the satellite control system and its features as well.

Test 4:

Turn off the satellite control system and turn on the manipulator control system.

-the manipulator shall take its end effector as close of the defined target vector as possible. In this case, the commanded target vector is out of the virtual manipulator workspace.

Test 5:

Turn on again the satellite's control system while the manipulator pursues the defined target vector.

-this test aims to show us how the virtual manipulator behaves in the scenario of cooperative motion of manipulator and satellite.

Test 6:

After test 3 and before test 4, the target vector is defined to be inside the manipulator workspace.

-this test intends to demonstrate that the virtual manipulator control system is able to reach a point inside its workspace and maintain a given position.

Observation 2: After each test, the logging file can be found in C:\windriver\workspace.

Observation 3: After each test, the test protocol sheet must be filled in.

C SAROS - GIT COMMIT HISTORY

```
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Fri Mar 15 10:36:50 2019 +0100

    BN: adjusted blocks position

Author: ACS-User <noemail@email.de>
Date:   Thu Mar 14 18:36:03 2019 +0100

    BN: implemented maximum to file blocks regarding EPOS

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Thu Mar 14 15:59:16 2019 +0100

    BN: all to file blocks were defined array

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Thu Mar 14 11:42:04 2019 +0100

    BN: added all necessary to file blocks

Author: ACS-User <noemail@email.de>
Date:   Wed Mar 13 13:03:13 2019 +0100

    BN: added rate transition and to file blocks to some outputs

Merge: 5210938 e9aaf67
Author: ACS-User <noemail@email.de>
Date:   Tue Mar 12 16:02:29 2019 +0100

    Merge branch 'saros_develop' into saros_local_sim_pass

Merge: dde8083 69fa865
Author: ACS-User <noemail@email.de>
Date:   Tue Mar 12 16:00:00 2019 +0100

    Merge branch 'feature_tests_implementation' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Tue Mar 12 11:25:44 2019 +0100

    BN: prepared inputs for tests routine

Merge: 8fbff49 dde8083
Author: ACS-User <noemail@email.de>
```

```
Date:   Thu Mar 7 14:36:01 2019 +0100

    Merge branch 'saros_develop' into saros_local_sim_pass

Merge: 8fbff49 cb35d8f
Author: ACS-User <noemail@email.de>
Date:   Thu Mar 7 14:35:08 2019 +0100

    Merge branch 'bug_fixed_non-feasible_orientation' into saros_develop

Author: ACS-User <noemail@email.de>
Date:   Thu Mar 7 14:34:18 2019 +0100

    BN: changed ti variable in RAS.m

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Wed Mar 6 17:50:31 2019 +0100

    BN: added robot trigger recording

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Wed Mar 6 10:05:46 2019 +0100

    BN: improved test for initialization

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Fri Mar 1 18:37:30 2019 +0100

    BN: organized output displays

Author: ACS-User <noemail@email.de>
Date:   Thu Feb 28 13:09:20 2019 +0100

    BN: fixed the wrong outputs of saros and displays

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Thu Feb 28 10:21:02 2019 +0100

    BN: adjusted limits of actuators saturation block

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Tue Feb 26 10:10:01 2019 +0100

    BN: added rotation matrix orthogonalization block and adjusted control gains

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
```

```
Date:   Mon Feb 25 10:42:43 2019 +0100

    BN: added a few displays in order to compare variables'values

Merge: a089bd7 860d4a4
Author: ACS-User <noemail@email.de>
Date:   Fri Feb 22 14:27:46 2019 +0100

    Merge branch 'saros_develop' into saros_local_sim_pass

Merge: 7bd2b48 122b075
Author: ACS-User <noemail@email.de>
Date:   Fri Feb 22 13:23:17 2019 +0100

    Merge branch 'bug_fixed_RAS_correction' into saros_develop

Author: ACS-User <noemail@email.de>
Date:   Fri Feb 22 13:20:38 2019 +0100

    BN+HB: updated initial parameters for EPOS

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Fri Feb 22 10:19:36 2019 +0100

    BN: corrected mask options for target reference

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Wed Feb 20 10:59:27 2019 +0100

    BN: adjusted mask configurations

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Wed Feb 20 10:41:48 2019 +0100

    BN: exchanged extrinsic function for embedded function

Merge: e7d80d5 ce610b2
Author: ACS-User <noemail@email.de>
Date:   Mon Feb 18 14:05:24 2019 +0100

    Merge branch 'bug_fixed_clear_all_fix_in_LoadFcn' into saros_develop

Author: ACS-User <noemail@email.de>
Date:   Mon Feb 18 13:47:17 2019 +0100

    HB+BN: modified LoadFcn in Callbacks of SAROS block
```

```
Merge: dbf653a f289fe6
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 18 08:41:56 2019 +0100

    Merge branch 'feature_contants_usage' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 18 08:40:15 2019 +0100

    BN: initial parameters were defined as constants

Merge: 6e7fb32 f6e234e
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 18 08:09:52 2019 +0100

    Merge branch 'feature_initial_orientation' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 18 07:48:00 2019 +0100

    BN: added switch to enable target initial position

Merge: 647f141 a1e600f
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 18 08:07:17 2019 +0100

    Merge branch 'bug_fixed_initial_parameters' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Thu Feb 14 10:29:52 2019 +0100

    BN: initial orientation variables were passed to SAROS

Merge: db63e14 152ea31
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 18 08:04:52 2019 +0100

    Merge branch 'feature_initial_parameters' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 11 11:23:23 2019 +0100

    BN: removed order of magnitude factor

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
```

```
Date:   Mon Feb 11 10:55:56 2019 +0100

    BN: initial parameters were properly defined

Merge: 8ea31b0 2c42533
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 11 08:48:54 2019 +0100

    Merge branch 'bug_fixed_callback' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 11 07:54:55 2019 +0100

    BN: modified stopfcn callback

Merge: 2c70cd1 cee11c6
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 11 08:36:46 2019 +0100

    Merge branch 'tech_debt_translation' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Thu Feb 7 15:19:47 2019 +0100

    BN: translated matlab code comments

Merge: a089bd7 bc7a948
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 11 08:31:05 2019 +0100

    Merge branch 'feature_io_implementation' into saros_develop

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Wed Feb 6 13:33:51 2019 +0100

    BN: vector correction to use ECI system

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Wed Feb 6 13:03:00 2019 +0100

    BN: fixed SAROS output links order

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Tue Feb 5 16:24:27 2019 +0100

    BN: adjusted path to SAROS files in the root folder
```

```
Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Tue Feb 5 11:20:44 2019 +0100

    BN: Generic IO block replaced by SAROS model

Author: Brazil Nardin <Anderson.BrazilNardin@dlr.de>
Date:   Mon Feb 4 16:37:17 2019 +0100

    BN: Added simple IO block

    Implemented simple IO block which receives target position and provides
chaser position and attitude.
```