Chapter 4

Uncertainty Quantification in Inverse Kinematics Computation of Space Robots by Neural Networks

Chapter details

Chapter DOI:

https://doi.org/10.4322/978-65-86503-88-3.c04

Chapter suggested citation / reference style:

Santos, Rogerio R., et al. (2022). "Uncertainty Quantification in Inverse Kinematics Computation of Space Robots by Neural Networks". *In* Jorge, Ariosto B., et al. (Eds.) *Uncertainty Modeling: Fundamental Concepts and Models*, Vol. III, UnB, Brasilia, DF, Brazil, pp. 89–118. Book series in Discrete Models, Inverse Methods, & Uncertainty Modeling in Structural Integrity.

P.S.: DOI may be included at the end of citation, for completeness.

Book details

Book: Uncertainty Modeling: Fundamental Concepts and Models

Edited by: Jorge, Ariosto B., Anflor, Carla T. M., Gomes, Guilherme F., & Carneiro, Sergio H. S.

Volume III of Book Series in:

Discrete Models, Inverse Methods, & Uncertainty Modeling in Structural Integrity

Published by: UnB City: Brasilia, DF, Brazil Year: 2022

DOI: https://doi.org/10.4322/978-65-86503-88-3

Uncertainty Quantification in Inverse Kinematics Computation of Space Robots by Neural Networks

Rogerio R. Santos[1*], Ijar M. da Fonseca[2], and Domingos A. Rade[3]

[1]Division of Mechanical Engineering, Technological Institute of Aeronautics - ITA, Brazil. E-mail: rsantos9@gmail.com
[2]Division of Aeronautical and Aerospace Engineering, Technological Institute of Aeronautics - ITA, Brazil. E-mail: ijar@ita.br
[3]Division of Mechanical Engineering, Technological Institute of Aeronautics - ITA, Brazil. E-mail: rade@ita.br

*Corresponding author

Abstract

Service satellite are used in a variety of applications, justified by scientific, economic, strategic and social benefits. The perspective of working with space equipment is largely based on the assumption that there are robotic mechanisms capable of handling multiple payloads. Normally, robot path planning involves determining the inverse kinematics configuration, since the workspace is expressed in Cartesian coordinates and the robot control is performed through joint angles. The present study discusses strategies for calculating the inverse kinematics of a space serial manipulator. The results of a classical industrial kinematics model and a multibody dynamics formulation are compared in the context of space operation. A Neural Network model is considered to quantify the uncertainty about the end-effector positioning and joint angle values. It was found that the Neural Network is a robust predictor that, together with a Nonlinear Programming strategy, represents an effective methodology for inverse kinematics calculations. This result contributes to the establishment of algorithms that will increase the operational autonomy of space equipment in different scenarios of orbital operation.

Keywords: Robot Path Planning, Berthing, Satellite Dynamics, Machine Learning, PyTorch, OpenModelica

1 Introduction

The movement autonomy of space robots is a contemporary challenge of great relevance. It consists in the ability of fulfilling goals without the need of external control operators. There is a wide range of applications for service satellites, and its relevance is justified by scientific, economic and strategic benefits.

Long-distance space travel requires the existence of space deposits along the way. The service satellites are enabling agents of such infrastructure. Another activity destined for the service satellite is the manipulation of objects in orbit, enabling the removal of space debris and maintenance of space vehicles in orbit. Additionally, as the positioning of satellites in the Geostationary Earth Orbit (GEO) depends on the availability of a free and viable position in orbit, there is commercial and security interest in removing failed or unused satellites for reuse from its orbital position. Additionally, the maintenance of geostationary satellites increases their lifespan and leads to better use of space.

The repair of space equipment is another important application of service satellites, for example, the Hubble Space Telescope's maintenance service mission (1993). The lifespan of valuable resources can be extended through maintenance and upgrade actions. From this perspective, we see the need to develop the equivalent of a "technical assistance network" in orbit to solve problems of critical failures in space equipment, leading to the interruption or complete loss of the space mission.

For the activities of providing services in orbit, in addition to the strategic importance regarding the domain of space exploration technology, it also involves the economic aspect of great relevance already identified by companies around the world. We live in a new phase of the space race characterized by the participation of private companies, focused on the economic exploitation of space resources.

NASA's Artemis program (Smith et al. [2020]) will send astronauts to the Moon through a global initiative in the engineering, technology development, and process improvement necessary to safely conduct human exploration to the Moon. This initiative includes the contribution of companies and international partners to the exploration and development of the Moon.

Companies such as SpaceX (Mróz et al. [2022]) and Blue Origin have been involved in the development of space transport equipment, focusing on economic viability through the reuse of launch vehicles. Boeing provides a series of resources oriented towards space exploration. Among the resources offered, there is the provision of a global positioning system infrastructure (Global Positioning System IIF) (Kuang et al. [2017]) composed of a large number of satellites. Boeing Commercial Satellites designs and markets communication and cargo satellites for commercial purposes, such as telecommunication infrastructure, broadband, and scientific applications (Skinner et al. [2013]).

Airbus presents on its website the possibility of space equipment maintenance services to be offered in the coming years. The "Airbus O. CUBED Services" will be operated by a category of space vehicle called "SpaceTug". The SpaceTug family of service satellites contains advanced technologies for robotics, electric propulsion, computer vision-based navigation, as well as approaching and intercepting strategies. On-orbit services (OOS) will be offered in three main categories:

- Inspection and maintenance service for telecommunications satellites;
- On-orbit servicing for geostationary orbit (GEO) and low-Earth orbit (LEO);

- Debris Removal Service from the Space vehicles orbits.

On-orbit vehicle maintenance and orbital debris removal services should operate autonomously (Aglietti et al. [2020]) based on intelligent strategies for moving and manipulating objects. The feasibility of this vision is largely based on the existence of robotic mechanisms that are able to act effectively in the manipulation of different objects. The view on the commercial importance of space exploration is confirmed by the large number of satellites in operation today. According to ESPI [2020a], the number of satellites operating in Earth orbit in March 2019 was 2062, distributed as follows:

- Communication: 773;
- Earth observation: 768;
- Scientific research: 105;
- Navigation: 138;
- Technology: 265;
- Other missions: 13.

In ESPI [2020b] the policies of different European countries for the development of space programs are presented, together with justifications on the strategic importance of such an initiative. According to the information in the report, two concerns common to the countries involved are:

- Development of technologies that lead to the autonomy of space operations, in order to establish situational awareness for action at the international level, that is, the ability to recognize any possible issues once you arrive at the scene and act proactively to avoid a negative impact;
- Expansion of knowledge and training of people involved in space activities as well as maintenance of the competence of specialists in the area of space security.

In this context, from a strategic point of view, the development of space exploration technologies is fundamental, in order to contribute to the maintenance of national sovereignty, as well as to contribute to the development of cutting-edge technologies with great industrial and commercial impact.

The paper is organized as follows. The Section 2 presents the main characteristics for robot kinematics representation. Inverse kinematics of space robots is discussed in Section 3. The Section 4 presents a popular interpolation scheme for path planning. The Section 5 outlines the principles involved in torque calculation, while the feature-rich approach of multibody dynamics is discussed in Section 6. Numerical experiments that quantify uncertainty are detailed in Section 7. The Section 8 summarizes the conclusions of the present study.

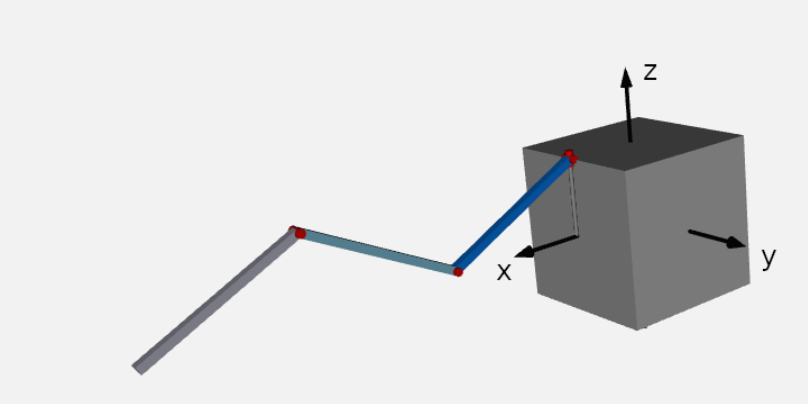


Figure 1: Space robot composed by a satellite and a serial manipulator.

2 Space Robots

The concept of space robot considered in the present paper consists of a space satellite with a serial manipulator attached to it. The schematic representation of the spaceship is given in Figure 1.

The kinematic model of the industrial serial manipulator describes the influence of joint angles in the placement of the end-effector. Usually, an $n-$link manipulator is mathematically represented by a set of homogeneous transformations $A_i, i = 1, \ldots, n$ that describe the Cartesian positioning from the base $P_{base}(x, y, z)$ to the end-effector $P_{end}(x, y, z)$. Following this convention, $A_1 = T_0^1, A_2 = T_1^2, \ldots, A_n = T_{n-1}^n$ and the serial manipulator can be represented by the matrix

$$T_0^n(q) = T_0^1(q_1)T_1^2(q_2)\ldots T_{n-1}^n(q_n) \tag{1}$$

where q_i is the $i-$th joint coordinate, that is, $q_i = \theta_i$ for rotational joint or $q_i = d_i$ for translation joint (Craig [2005]).

In the context of applications, the paper by da Fonseca [2021] presents an overview of the space age and the first robotic probes preceding the landing of the astronauts on the Moon. It discusses space robotics fundamental concepts, classification, and safety-critical aspects for space robotics applications. The space robotics state of the art presented focuses on the International Space Station (ISS) Extravehicular Activities (EVA), the planetary explorations (like the current Mars exploration), JAXA's Hayabusa mission that rendezvoused and landed on the 25143 Itokawa asteroid, and in the ESA/DLR's Rosetta spacecraft carrying its Philae robotic module which landed on the 67P / Churyumov-Gerasimenko comet. The article includes a discussion of on-orbit servicing and rendezvous & docking/berthing (RVD/B).

2.1 Robot Kinematics

To determine the inverse kinematics of a serial robot manipulator involves the computation of the joint angles from a given Cartesian position and orientation of the end-effector.

This computation is of primary importance in the programming and control of robot manipulators. The corresponding procedure has to do with nonlinear algebraic equations,

considering that there is no analytical closed-form solution for a general robot structure. Consequently, it is necessary to establish a way to compute the inverse kinematics for a wide class of robots and related tasks that are assigned to them.

When the end-effector moves along a continuous trajectory, it is not possible to switch from one kind of solution to another, arbitrarily. Uchiyama [1979] pointed out that the trajectories of multiple solutions intersect at singular points and solutions can be switched only at those points. The singular points of a robotic mechanism are defined as the singular points of the implicit function, namely, the joint angles that make the Jacobian matrix not to have full rank. Besides, singular points can bring control difficulties and poor positioning accuracy.

Moreover, when working into complex environment, the manipulator can be subjected to work into restricted workspaces, both for sided constraints (like walls and floor), and for internal constraints (as the geometry of manipulated object). Effective optimization strategies to overcome such difficulties are proposed by dos Santos et al. [2008b], dos Santos et al. [2010], and Santos et al. [2022], among others.

Adopting the Denavit-Hartenberg convention (Spong and Vidyasagar [1989], Craig [2005]) to define A_i, any serial robot can be represented by Equation(1). Hence, given the base P_{base} and joint $q_i(d, \theta)$ values, the Cartesian position of the end-effector P_{end} is given by

$$P_{end} = T_0^n(q).P_{base}. \tag{2}$$

2.2 Inverse Kinematics

Adopting the Denavit-Hartenberg convention (Craig [2005], Spong and Vidyasagar [1989]) to define A_i, any serial robot can be represented by Equation 1. Hence, given the base P_{base} and joint $q_i(d, \theta)$ values , it is possible to determine the end-effector cartesian position P_{end} through the equality

$$P_{end} = T_0^n(q).P_{base}. \tag{3}$$

To solve the inverse kinematics problem means to solve the inverse problem, i.e., given P_{base} and P_{end} find $q_i(d, \theta)$ for which Equation 3 holds.

One approach to solve the inverse kinematics problem is to find a closed-form solution by using algebra and geometry. Although this approach is generally more desirable in applying the solution to real-time control of robots, it is not always possible to obtain closed-form solutions for manipulators with an arbitrary mechanism due to the complexity of Equation 3.

Rather, the class of manipulators mechanisms for which the closed-form solutions are guaranteed is very limited. The algebraic approach to closed-form solutions is related to finding q through various algebraic transformations of Equation 3; the geometric approach means finding q by using geometrical heuristics to take advantage of the special structure of the manipulator.

The paper by da Fonseca et al. [2019] considers the forward and inverse kinematics for a space robot. The main focus is to compute the position and orientation of manipulators' end-effectors relative to their platform. The set of coordinate frames follows the convention for frames that appears in the literature for ground robot manipulators. The kinematics related to the spacecraft attitude is added in the formulation, providing

an overview of the kinematics equations for spacecraft-type manipulators. The inertial, orbital, and body-fixed coordinate frames are included in the kinematics study.

2.3 Inverse kinematics as an optimization problem

Considering the manipulator kinematics described by Equation 3 and an end-effector point defined as P_{end}, then it is desired to get $q(d, \theta)$ values that obey Equation 3. However,

$$T_0^n(q).P_{base} = P_{end} \;\Rightarrow\; T_0^n(q).P_{base} - P_{end} = \vec{0} \;\Rightarrow$$
$$|T_0^n(q).P_{base} - P_{end}| = 0 \tag{4}$$

Therefore, the optimal solution q^* (i.e. minimal) of the objective function given by

$$f_1(d, \theta) = |T_0^n(q(d, \theta)).P_{base} - P_{end}|^2 = 0 \tag{5}$$

is also solution of Equation 3, because $q(d, \theta)$ satisfies Equation 5 if, and only if, it satisfies Equation 3. It is an expression for the misalignment between the base and tool frames, that is, the transformation matrix align both systems of coordinates, the symbol $|.|$ is a vector norm, and the squared refers to the quadratic error (dos Santos et al. [2005], dos Santos et al. [2010]). Then, the inverse kinematics parameter $q(d, \theta)$ can be achieved through an optimization procedure by using Equation 5 as the objective function.

This approach has the main advantage of representing a generic procedure that can be developed for different architectures of serial robot manipulators. As will be shown, this approach leads to efficient computation of the inverse kinematics (positioning and orientation) by using global and local optimization methods.

Along the investigation of the optimization problem, there are two kinds of solution points (Luenberger and Ye [2008]): *local minimum points*, and *global minimum points*. The definitions below are related to these solution points.

Definition A point $q^* \in \Omega$ is said to be a *relative minimum point* or a *local minimum point* of f over Ω if there is an $\epsilon > 0$ such that $f(q) \geq f(q^*)$ for all $q \in \Omega$ and $|q - q^*| < \epsilon$. If $f(q) > f(q^*)$ for all $q \in \Omega,\ q \neq q^*,\ |q - q^*| < \epsilon$, then q^* is said to be a *strict relative minimum point* of f over Ω.

Definition A point $q^* \in \Omega$ is said to be a *global minimum point* of f over Ω if $f(q) \geq f(q^*)$ for all $q \in \Omega$. If $f(q) > f(q^*)$ for all $q \in \Omega,\ q \neq q^*$, then q^* is said to be a *strict global minimum point* of f over Ω.

As the inverse kinematics solution $q(d, \theta)$ of Equation 5 is also a global minimum, it is possible to obtain the inverse kinematics by means of a global optimization procedure.

The local minimum of Equation 5 can be easily obtained by classical nonlinear optimization methods, such as the Gradient method, the Conjugate Gradient method, and the Quasi-Newton based methods, among others. However, it can result a minimum which does not satisfy $f_1(q) = 0$. Then, Equation 5 does not hold. To avoid this situation, the initial point is changed and the optimization is performed until the global minimum is achieved.

To improve the robustness of the process, it is possible to use a global minimizer. There are many methods devoted to find a global minimum of a nonlinear objective function, despite that to find a global minimum is not an easy task. Well known methods such as genetic algorithms (Le Grand and Merz Jr. [1993]), differential evolution algorithm (Storn and Price [1997]), simulated annealing (Romeijn and Smith [1994]), or other heuristic (Choze et al. [2022a]) could be used in this case. The main characteristic of these methods is that the global (or near global) optimum is obtained through a high number of functional evaluations.

3 Inverse Kinematics of Space Robots

To solve the inverse kinematics problem means to solve the inverse problem, i.e., given P_{base} and P_{end}, find $q_i(d, \theta)$ for which Equation 2 holds. It will require the solution of the inverse problem

$$\min_{\theta_i} |T_0^n(q(\theta_i))P_{base} - P_{end}| \tag{6}$$

The Denavit-Hartenberg parameters of the manipulator, as seen in Figure 1, is shown in Table 1. The parameter a refers to a fixed translation while α represents a fixed rotation

Table 1: Denavit-Hartenberg parameters.

Joint	a (m)	α (rad)	d (m)	θ (rad)
1	0	1.57	0	θ_1
2	1	0	0	θ_2
3	1	0	0	θ_3
4	1	0	0	θ_4

of the coordinate system. The joint value d express a translation joint while θ represents the angle of a rotational joint (Craig [2005], Spong and Vidyasagar [1989]).

The classical formulation of the industrial device, Equation 2, contains the hypothesis that the base of the manipulator is fixed. Since the satellite is traveling in orbit and may floats under the reaction forces and torque arising from the joints' motion control, the system dynamics (satellite plus the attached manipulator) is more complex than that of manipulators dynamics on ground. Therefore, to contemplate the dynamic effects along the movement, a higher fidelity model is considered.

The high fidelity model is build using the OpenModelica software (Fritzson et al. [2005]), which provides a multibody dynamics library and will compute the coupled effects between the robot and the satellite while floating in the space. The parameters are shown in Table 2.

Given the displacement Δ of the satellite, the position $P_{effector}$ of the end-effector and the position P_{target} of the target, the inverse problem for obtaining the inverse kinematics is given by

$$\min_{\theta_i} |\{P_{effector}(\theta_i) + \delta(\theta_i)\} - P_{target}| \tag{7}$$

which minimum value corresponds to the Euclidean distance between the end-effector and the target. The disturbance due to multiple dynamic effects and coupling between satellite and robot is accounted for by the δ parameter (Fonseca et al. [2021]).

Table 2: Satellite and robot parameters for dynamics computation.

Device	Dimension$_{x,y,z}$ (m)	Mass (kg)
Satellite	$1 \times 1 \times 1$	90
Link 1	$1 \times 0.05 \times 0.05$	0.3
Link 2	$1 \times 0.05 \times 0.05$	0.3
Link 3	$1 \times 0.05 \times 0.05$	0.3

4 Path Planning

An important aspect of the robotic device is the ability to compute a trajectory that describes the desired motion of a manipulator in multidimensional space. It consists in a time history of position, velocity, and acceleration for each degree of freedom of the manipulator.

Multiple strategies are available and in the simplest case the mission will specify the desired goal position and orientation of the end-effector and the exact shape of the path to get there, the duration, the velocity profile, and other details will be automatically established by an algorithm.

The desired outcome of a plan will be represented in terms of the state and actions that are executed. The main goals of a trajectory planning system are:

1. Feasibility: find a plan that causes arrival at a final state, regardless of its efficiency;
2. Optimality: find a feasible plan that optimizes performance following one or more performance indices, in addition to arriving in a goal state.

Furthermore, in order to guarantee smooth paths, some sort of constraints may be included on the spatial and temporal aspects of the path through the usage of via points, which consists in the establishment of intermediate points between the initial and the final configurations.

The initial position of the manipulator is known according to a Cartesian reference and also in the form of a set of joint angles. A planning strategy shall provide a set of joint angles whose value at t_0 is the initial position of the joint and the set of joint angles at t_f is the desired goal position of that joint. In making a smooth motion through a polynomial function, at least four constraints will arise. Two constraints are due to the selection of initial and final values:

$$\begin{aligned} \theta(0) &= \theta_0 \\ \theta(t_f) &= \theta_f \end{aligned}$$

Two additional constraints are due to the continuity in velocity, which in this case means that the initial and final velocities are zero:

$$\begin{aligned} \dot{\theta}(0) &= 0 \\ \dot{\theta}(t_f) &= 0 \end{aligned}$$

These four constraints can be satisfied by a polynomial of third degree. Satisfying such constraints uniquely specify a particular cubic polynomial. A cubic has the form

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

and the joint velocity and acceleration are:

$$\begin{aligned}\dot{\theta}(t) &= a_1 + 2a_2 t + 3a_3 t^2 \\ \ddot{\theta}(t) &= 2a_2 + 6a_3 t\end{aligned}$$

Combining the equations of velocity and acceleration with the four desired constraints yields four equations in four unknowns:

$$\begin{aligned}\theta_0 &= a_0 \\ \theta_f &= a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 \\ 0 &= a_1 \\ 0 &= a_1 + 2a_2 t_f + 3a_3 t_f^2\end{aligned}$$

Solving the equations for a_i, it follows that the explicit values of constants are

$$\begin{aligned}a_0 &= \theta_0 \\ a_1 &= 0 \\ a_2 &= \frac{3}{t_f^2}(\theta_f - \theta_0) \\ a_3 &= -\frac{2}{t_f^3}(\theta_f - \theta_0)\end{aligned}$$

and uniquely defines a cubic polynomial.

In the case $t_0 = 0$ s, $t_f = 30$ s, $\theta_0 = 0$ rad and $\theta_f = \frac{\pi}{2}$ rad, the a_i parameters are $a_0 = 0, a_1 = 0, a_2 = 0.0052$ and $a_3 = -0.0001$. The polynomial $\theta(t)$ is shown in Figure 2.

This concept can be generalized through an interpolation scheme. For this, the intermediate reference r_1 and the final reference point r_2 are added to the analysis. By moving the position of the reference points, a spline interpolation scheme is applied to the references r_0, r_1 and r_2. The concept is shown in Figure 3. Different interpolation data will lead to a different physical movement.

5 Torque and mechanical power

The Lagrangian $L = K - P$ is defined by the difference between the kinetic energy K and the potential energy P of the system. The dynamics of the system can be described by the Lagrange equation as given by:

$$u_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} \tag{8}$$

where, q_i are the generalized coordinates (θ_i for rotational joints and d_i for prismatic joints); $\dot{q}_i$ are the generalized velocities (angular velocity $\dot{\theta}_i$ for rotational joints and $\dot{q}_i$ for prismatic joints); and u_i are the generalized forces.

By using Equation 8, the generalized forces u_i can be written as:

$$u_i = \sum_{j=1}^{n} D_{ij}\ddot{q}_j + I_{ai}\ddot{q}_i + \sum_{i=1}^{n}\sum_{k=1}^{j} C_{ijk}\dot{q}_j\dot{q}_k + G_i \tag{9}$$

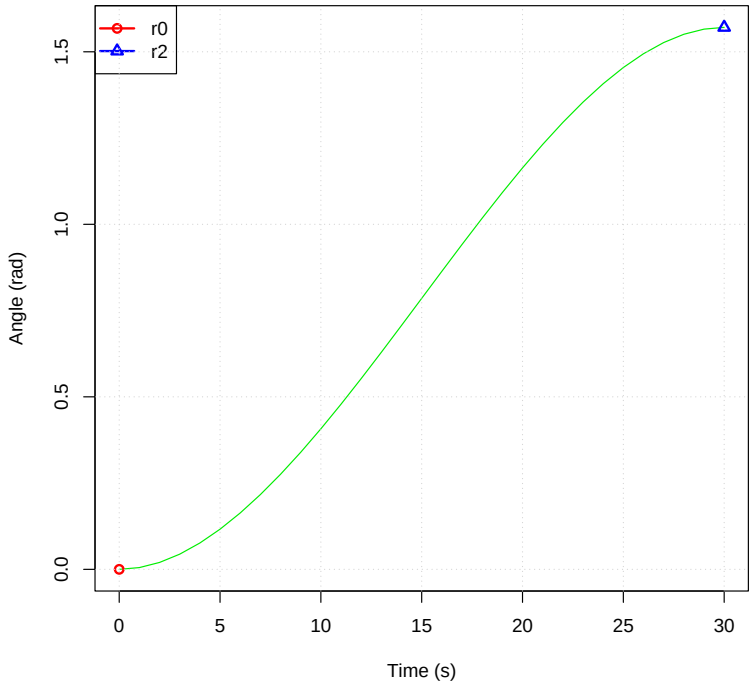


Figure 2: Cubic interpolation of joint coordinates.

where, D_{ij} is the inertia matrix of the system; C_{ijk} is the matrix that takes into account Coriolis and centrifugal effects; I_{ai} represent the inertia of the actuators, and G_i is the vector of gravitational forces.

Due to the relationship that exists between energy and force, the minimal energy can be estimated from the generalized force $u_i(t)$ that is associated to each joint i at the time instant t. The mechanical power can be used for design purposes, as defined by:

$$f = \int_{t_0}^{t_f} \sum_{i=1}^{n} \left(u_i^T(t)\dot{q}_i(t)\right)^2 dt \tag{10}$$

This expression is representative of the phenomenon under study because it considers both the kinematic and dynamical aspects of the trajectory, simultaneously (Steffen Jr. and Saramago [1997], Saramago and Steffen Jr. [1998]).

Should be pointed out that by using a discrete set with limited number of intermediate reference points, the interpolation schema provides a trajectory with relevant information in terms of mechanical energy (dos Santos et al. [2008a]).

In the scenario of space missions, the joint forces and torques may incur in disturbance over the satellite and end-effector movements. The paper by da Fonseca et al. [2018] deals with the dynamic analysis of the close approach phase of a space robot to a target spacecraft. The computer simulations of the equations of the dynamics (relative orbital and attitude equations of motion) yields the grasping conditions and the results are compared with the inverse kinematics grasping operation implemented in a lab experiment.

An alternative to describe the spacecraft dynamics is to use software that requires only a geometric description of the system, as described in the next section.

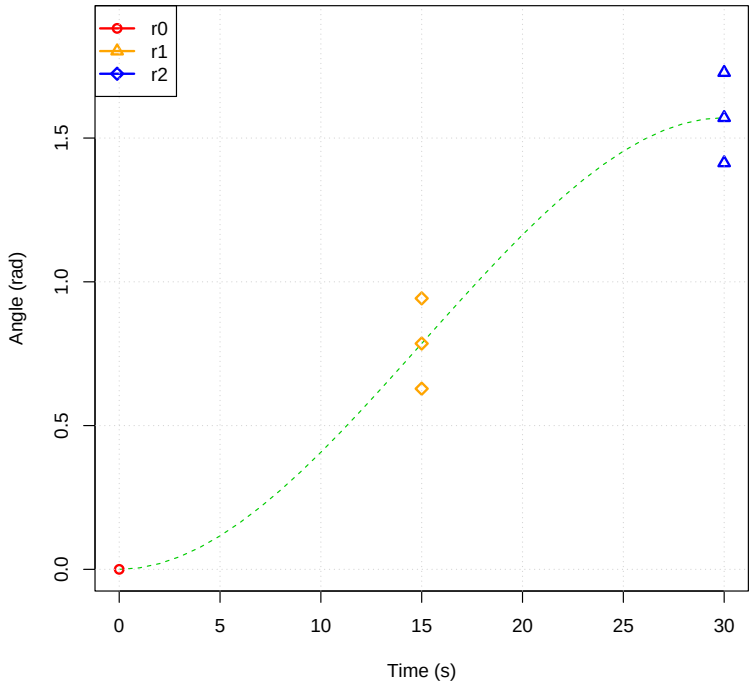


Figure 3: Spline interpolation of joint coordinates through reference points r_1 and r_2.

6 Multibody modeling approach with OpenModelica software package

The paper Pulecchi and Lovera [2006] discusses the suitability of Modelica language for modeling Attitude and Orbit Control Systems (AOCS). The ability to use multiple coordinate frames through visual connectors, the existence of rich MultiBody Library to describe spacecraft dynamics, and the ability to describe multi-physics domains are some key aspect cited by the authors.

The Modelica language allows defining models in a declarative way, in hierarchical modules, combining several formalisms: ordinary differential equations (ODE), differential-algebraic equations (DAE), bond graphs, and finite state automata.

The multiple-domain capability of the language allows combining electrical, mechanical, hydraulic, and thermodynamic model components on the same application Fritzson et al. [2005]. By default, OpenModelica transforms a Modelica model into an ODE representation to perform a simulation by using numerical integration methods. DASSL is the default solver in OpenModelica. It is an implicit, high-order, multi-step solver with a step-size control (Petzold [1982]).

Elementary objects such as mass, spring, damper, and joints can be represented by visual elements, and do not require any explicit knowledge of the mathematical equations. An example of a 4-joint, 3-link robotic manipulator is shown in Figure 4.

As a result, the satellite body, joints, links and other mechanical and electrical elements are represented by building blocks. In this context, there is no need to write the dynamics equations explicitly. Code 1 shows a fragment of the automatically generated

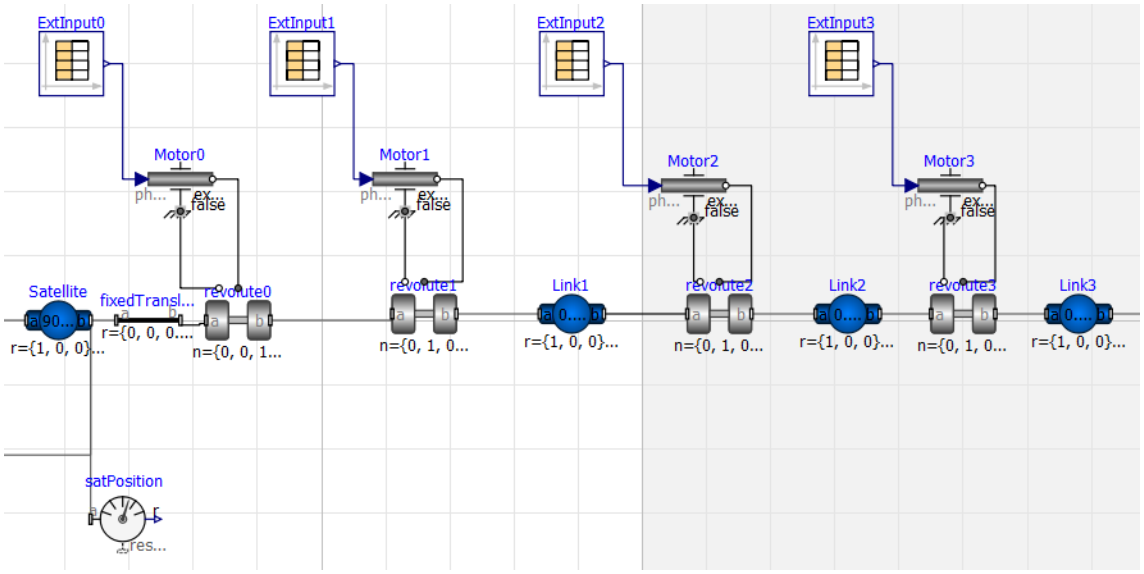


Figure 4: Fragment of a Space Robot model in OpenModelica.

equations. The corresponding Denavit-Hartenberg parameters are given in Table 1.

The satellite and robotic arm of the present work rely on a computational model built in OpenModelica software. A more in-depth discussion of software features and aerospace applications can be found in the works of Fritzson et al. [2006], Elmqvist et al. [1998], Lovera and Pulecchi [2006], Looye [2008], Briese et al. [2017], Moorman and Looye [2002], and Chen et al. [2017].

Some commercial front-ends for Modelica include AMESim (Siemens Software), Dymola (Dassault Systemes), Wolfram SystemModeler (Wolfram Research), SimulationX (ESI ITI GmbH), MapleSim (Maplesoft) and CATIA Systems Dynamic Behavior (Dassault Systemes).

Code 1: Code fragment of a Space Robot model in OpenModelica.

```
model SpaceRobot
inner Modelica.Mechanics.MultiBody.World world(axisLength =
   ↪  1, gravityType = Modelica.Mechanics.MultiBody.Types.
   ↪ GravityTypes.UniformGravity, n = {0, 0, -1});
Modelica.Mechanics.MultiBody.Joints.Revolute revolute1(n =
   ↪ {0, 1, 0},phi(displayUnit = "rad", fixed = false,
   ↪ start = 0), useAxisFlange = true);
Modelica.Mechanics.MultiBody.Joints.Revolute revolute2(n =
   ↪ {0, 1, 0},phi(displayUnit = "rad"), useAxisFlange =
   ↪ true);
(...)
Modelica.Mechanics.MultiBody.Parts.BodyShape Link1(m = 0.3,
   ↪  r = {1, 0, 0}, r_CM = {0.5, 0, 0}, shapeType = "box
   ↪ ", sphereDiameter = 0.01);
Modelica.Mechanics.MultiBody.Parts.BodyShape Link2(color =
   ↪ {135, 206, 235},m = 0.3, r = {1, 0, 0}, r_CM = {0.5,
   ↪ 0, 0}, shapeType = "box", sphereDiameter = 0.01);
Modelica.Mechanics.MultiBody.Sensors.AbsolutePosition
   ↪ effectorPosition(resolveInFrame = Modelica.Mechanics.
   ↪ MultiBody.Types.ResolveInFrameA.world);
Modelica.Mechanics.MultiBody.Parts.BodyShape Satellite(
   ↪ color = {200, 200, 200},height = 0.99, length = 0.99,
   ↪ m = 90, r = {1, 0, 0}, r_CM = {0.5, 0, 0}, shapeType
   ↪ = "box", sphereDiameter = 0.1, width = 0.99,
   ↪ widthDirection = {0,  0,1});
(...)
equation
connect(world.frame_b, traZ.frame_a);
connect(Satellite.frame_b, satAngles.frame_a);
connect(Satellite.frame_b, satPosition.frame_a);
connect(revolute3.frame_b, link3.frame_a)
(...)
connect(Link1.frame_b, revolute2.frame_a);
connect(Link2.frame_b, revolute3.frame_a);
connect(revolute0.frame_b, revolute1.frame_a);
connect(Satellite.frame_b, fixedTranslation.frame_a);
annotation(uses(Modelica(version = "3.2.2")),Diagram);
end SpaceRobot;
```

7 Uncertainty Quantification

The satellite and the robotic serial manipulator are coupled as shown in Figure 5. Four joint coordinates and tree links are considered. The Denavit-Hartenberg parameters (Table 1) that describe this robot are associated with eight reference points $r_1, \ldots, r_8$, as shown in Table 3.

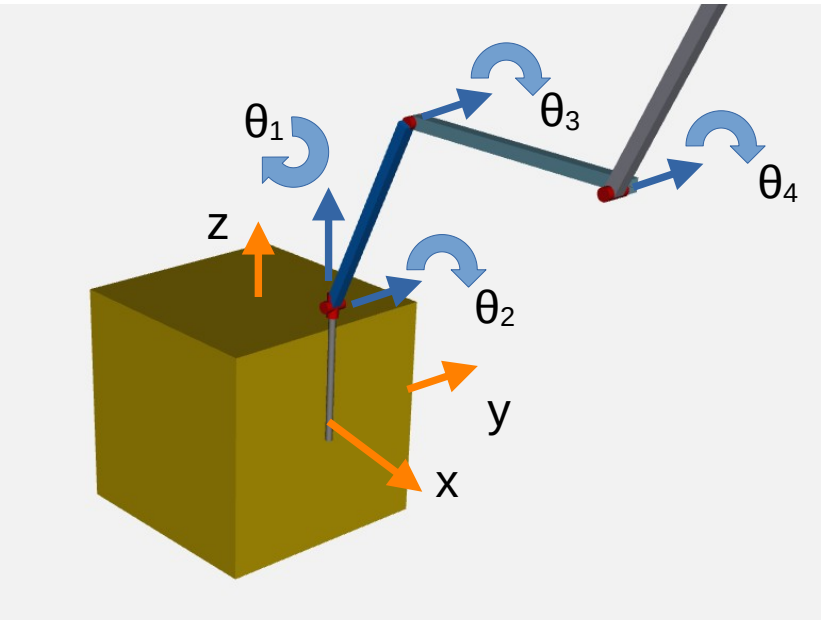


Figure 5: Space robot manipulator.

Table 3: Joint coordinate nomenclature.

Joint	Design variables	Joint coordinate
1	r_1, r_2	$\theta_1(r_1, r_2, t)$
2	r_3, r_4	$\theta_2(r_3, r_4, t)$
3	r_5, r_6	$\theta_3(r_5, r_6, t)$
4	r_7, r_8	$\theta_4(r_7, r_8, t)$

The berthing operation involves rendezvous between two orbiting space vehicles followed by berthing (grasping our capture operation). The relative attitude implies synchronizing the attitude motion of both spacecraft so that the relative attitude is zero. The same is valid fort the relative position. When the relative velocity is zero, the position of the center of mass of both spacecraft stays at the same distance from each other.

In the present work, the berthing box (Fehse [2003]) is centered at the position $(x, y, z) = (2.50, 0.00, 0.70)$ m in relation to the satellite origin and encompasses the uncertainty of $(\Delta x, \Delta y, \Delta z) = (\pm 0.21, \pm 0.21, \pm 0.21)$ m.

The numerical computations where performed using the R software (R Core Team [2020]) and considered the following step:

- Nonlinear programming algorithm (Johnson [2020]) solving the inverse problem (Equation 6);
- Nonlinear programming algorithm solving the inverse problem (Equation 7);
- Neural Network algorithm (Paszke et al. [2019]) learning the inverse kinematics solution from historical data.

These steps are evaluated as detailed in Algorithm 1.

Algorithm 1: Inverse kinematics computation.

$t_0 \leftarrow 0$
$t_f \leftarrow$ final time
$P_{end} \leftarrow$ Cartesian coordinates of target
$(\theta_1, \theta_2, \theta_3, \theta_4)_0 = (0, 0, 0, 0)$
`/* Generate estimate from low fidelity model */`
$\tilde{\theta}_i \leftarrow \min_{\theta_i} \left| T_0^n(q(\theta_i)) P_{base} - P_{end} \right|, i = 1, \ldots, 4$
`/* Optimize high fidelity model */`
Initial guess $\leftarrow \tilde{\theta}_i$
$\vec{\theta_f} \leftarrow \min_{\theta_i} \left| T_0^n(q(\theta_i)) \left\{ P_{base}(\theta_i) + \Delta(\theta_i) \right\} - P_{end} \right|, i = 1, \ldots, 4$
`/* Save historical designs */`
$n \leftarrow$ number of optimization iterations
$\vec{\theta}_{hist} = (\vec{\theta_f})_k, k = 1, \ldots n$
$\vec{P}_{hist} = (\vec{P}_{end})_k, k = 1, \ldots n$
$\vec{\Delta}_{hist} = (\vec{\Delta}(\theta_f))_k, k = 1, \ldots n$
`/* Neural network training */`
$RNN \leftarrow$ training from historical data $\{\vec{\theta}_{hist}, \vec{P}_{hist}, \vec{\Delta}_{hist}\}$.

In the following, 36 cases with distinct target points P_{end} were analyzed. The results provided by the DH algorithm are summarized in Table 4. The error corresponds to the distance between the end-effector and the target, computed through the high fidelity model (Equation 7). The measurement of the distance between the end-effector and the target confirmed that the stationary point was an optimal design.

The error dispersion, summarizing the distance between the end-effector and the target for all cases, is shown in Figure 6. After the optimization process, the final result (Figure 6, on the right side) was below the threshold $\varepsilon = 0.004$ m. The choice of a Nonlinear Programming strategy has a direct influence on the convergence rate and on the accuracy of the result. A comprehensive exposition of techniques and features can be found in the chapter "An overview of Linear and Non-linear Programming methods for Structural Optimization" (Choze et al. [2022b]) of this book series.

Table 4: Number of iterations and final error obtained from DH strategy.

Case	Target (m)	DH Iter.	DH Error (m)
1	(2.500, 0.000, 0.700)	141	0.004
2	(2.710, 0.210, 0.910)	121	0.003
3	(2.710, 0.210, 0.490)	174	0.003
4	(2.710, -0.210, 0.910)	111	0.003
5	(2.710, -0.210, 0.490)	153	0.003
6	(2.290, 0.210, 0.910)	117	0.003
7	(2.290, 0.210, 0.490)	151	0.002
8	(2.290, -0.210, 0.910)	127	0.003
9	(2.290, -0.210, 0.490)	139	0.003
10	(2.471, 0.050, 0.600)	145	0.003
11	(2.600, -0.023, 0.708)	136	0.003
12	(2.576, -0.051, 0.607)	150	0.003
13	(2.486, 0.103, 0.688)	144	0.003
14	(2.428, 0.007, 0.653)	137	0.003
15	(2.478, -0.102, 0.616)	139	0.003
16	(2.559, -0.011, 0.670)	145	0.003
17	(2.568, 0.090, 0.675)	136	0.004
18	(2.583, 0.056, 0.721)	137	0.003
19	(2.405, 0.075, 0.741)	166	0.003
20	(2.535, -0.078, 0.777)	134	0.004
21	(2.501, -0.061, 0.635)	148	0.002
22	(2.596, 0.035, 0.803)	117	0.002
23	(2.518, 0.023, 0.658)	141	0.003
24	(2.417, -0.070, 0.757)	141	0.003
25	(2.461, -0.088, 0.693)	139	0.002
26	(2.398, -0.037, 0.623)	148	0.003
27	(2.456, 0.070, 0.745)	132	0.003
28	(2.493, -0.015, 0.731)	130	0.002
29	(2.525, -0.096, 0.700)	150	0.003
30	(2.507, 0.013, 0.783)	136	0.003
31	(2.545, 0.040, 0.630)	137	0.002
32	(2.442, -0.031, 0.767)	137	0.003
33	(2.423, -0.003, 0.714)	141	0.003
34	(2.552, -0.050, 0.760)	141	0.002
35	(2.443, 0.087, 0.644)	126	0.002
36	(2.531, 0.066, 0.796)	132	0.003

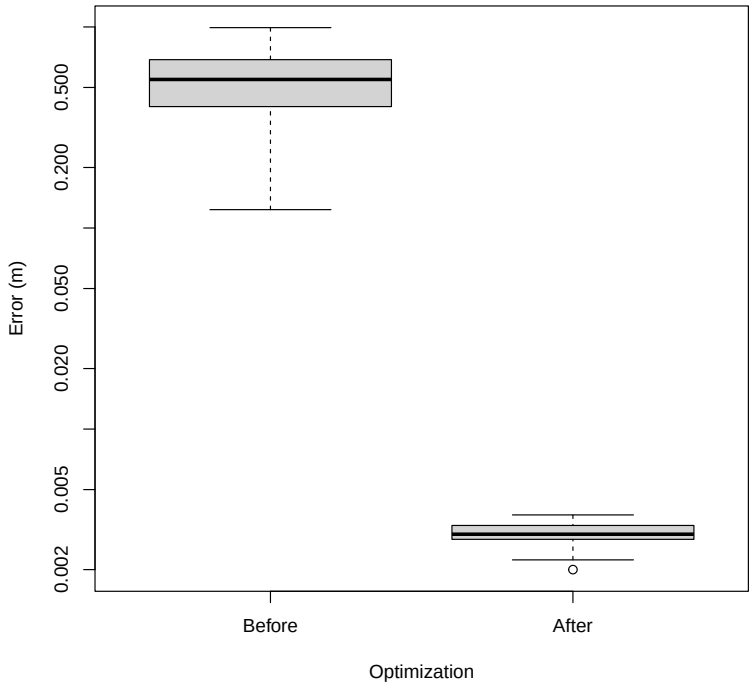


Figure 6: Error dispersion before and after the optimization process.

Code 2: Code of a Neural Network architecture to estimate the Cartesian position error.

```
class Net(T.nn.Module):
 def __init__(self):
  super(Net, self).__init__()
  self.hid1 = T.nn.Linear(8, 2400)
  self.outp = T.nn.Linear(2400, 1)

 def forward(self, x):
  x = T.relu(self.hid1(x))
  x = self.outp(x)
  return x
```

7.1 Error due to model simplification

Once the satellite is traveling in space, it may present oscillations influenced by the robotic movement, and is subject to the influence of external forces from the environment. As a result, the hypothesis of fixed robotic base (Equation 1) is not feasible in some situations. The Neural Network model, as shown in Code 2, is considered to estimate the discrepancy between the model with a fixed base and the model with a floating base.

The uncertainty that arises from this approach is evaluated by the computation of the same joint design $\vec{\theta}(r,t)$ by using the high fidelity model (Figure 4), and the difference between results of Equations 6 and 7 is shown in Figure 7. There is a similar pattern in terms of error dispersion along x, y and z dimensions, as shown in Figure 8.

The error is negligible in some situations but relevant in another. As a result, the optimization problem 6 given an useful estimate of joint angles at the expense of low computational power requirements. Notice that the median of the error (horizontal heavy line on the plot) is similar in all tree dimensions.

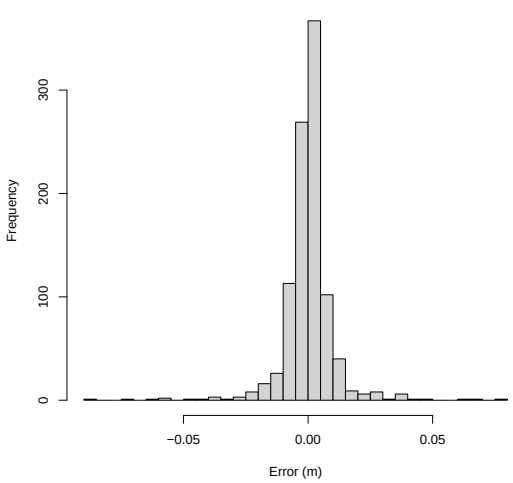
The most important reference points are those representing the final joint angles. Joint angles at the final time $\theta_i(t_f)$ corresponds to a Cartesian position of the end-effector. The cumulative error between the target and the end-effector is shown in Figure 9.

The z-axis has large deviation than other dimensions. According to the present convention (Figure 5) there is a higher displacement in the vertical axis while evaluating the target set (Table 4). The procedure is further detailed in Algorithm 2.

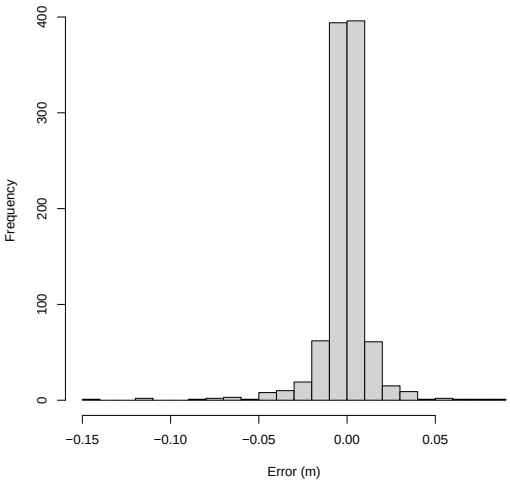
Algorithm 2: Inverse kinematics computation through the mathematical model.

$t_0 \leftarrow 0$
$t_f \leftarrow$ final time
$P_{end} \leftarrow$ Cartesian coordinates of target
$(\theta_1, \theta_2, \theta_3, \theta_4)_0 = (0, 0, 0, 0)$
`/* Generate estimate from low fidelity model */`
$\vec{\theta_f} \leftarrow \min_{\theta_i} \left| T_0^n(q(\theta_i)) P_{base} - P_{end} \right|, i = 1, \ldots, 4$

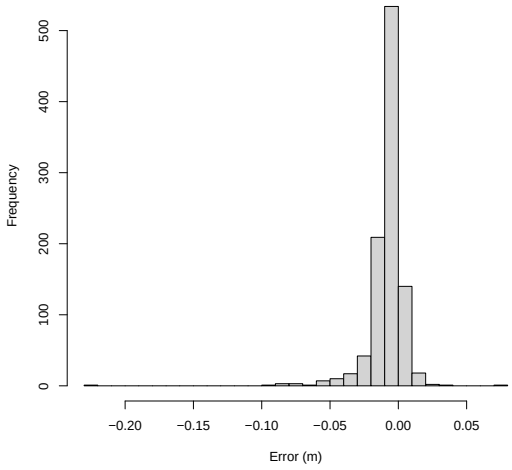
The main advantage of this procedure is the low computational cost of the CPU, which allows the evaluation of a multitude of scenarios under a low computational budget.



(a) X-axis



(b) Y-axis



(c) Z-axis

Figure 7: Historical error histogram along the x, y and z axes.

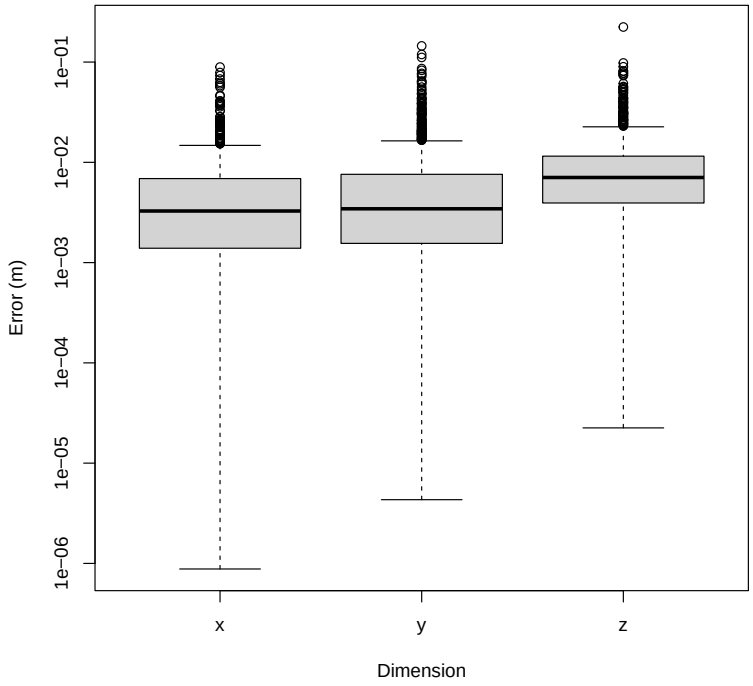


Figure 8: Comparison of error dispersion along the x, y and z axes.

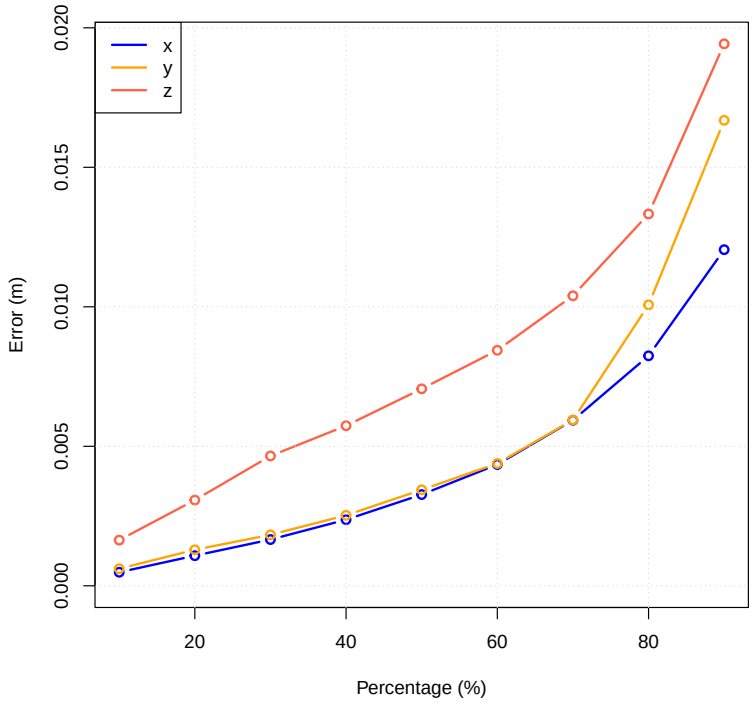


Figure 9: Cumulative error along the x, y and z axes.

Code 3: Code of a Neural Network architecture to estimate the joint angle value.

```
class Net(T.nn.Module):
 def __init__(self):
  super(Net, self).__init__()
  self.hid1 = T.nn.Linear(10, 2700)
  self.hid2 = T.nn.Linear(2700, 1031)
  self.hid3 = T.nn.Linear(1031, 394)
  self.hid4 = T.nn.Linear(394, 150)
  self.hid5 = T.nn.Linear(150, 57)
  self.hid6 = T.nn.Linear(57, 22)
  self.outp = T.nn.Linear(22, 1)

 def forward(self, x):
  x = T.relu(self.hid1(x))
  x = T.relu(self.hid2(x))
  x = T.relu(self.hid3(x))
  x = T.relu(self.hid4(x))
  x = T.relu(self.hid5(x))
  x = T.relu(self.hid6(x))
  x = self.outp(x)
  return x
```

7.2 Error on meta-modeling estimate

The calculation of the inverse kinematics though Equation 7 generated a history of 4949 joint coordinates θ and positions of end-effector P_{end}.

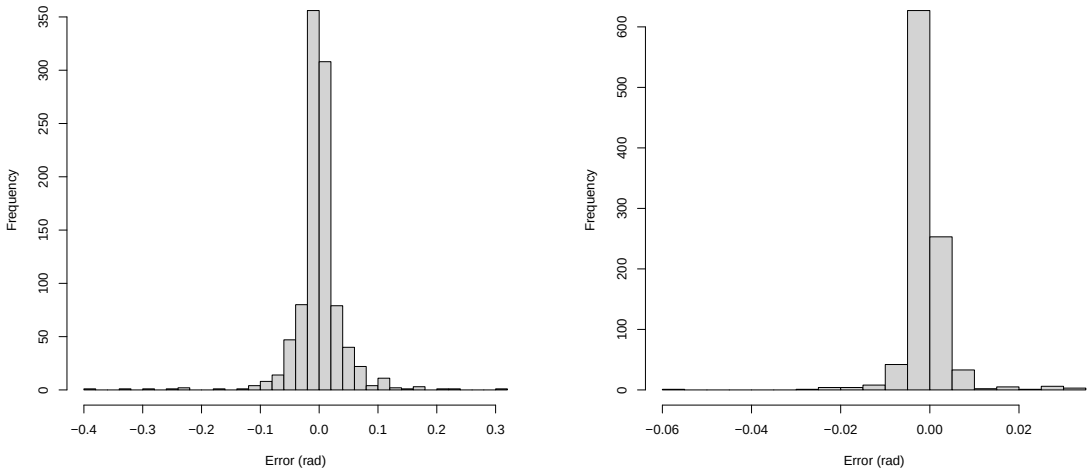
This section will discuss the existence of a meta-model that can learn from this data and predict the inverse kinematics configuration of other target points. The training data will be based exclusively on the historical data set acquired from the optimization procedure done in the previous section.

The values used to train a Neural Network consist of a subset of 3959 sample points, which corresponds to 80% of the data available. The meta-model will predict the optimal value of each reference point $\vec{r} = (r_1, \ldots, r_8)$. These points are used to calculate the value of the joint coordinates $\theta(\vec{r}, t)$ in the time interval $t \in [t_0, t_f]$. The model is tested against a subset of 990 points not used in the training phase, which corresponds to 20% of the data. The general architecture of the Neural Network is presented in the Code 3, following the syntax of PyTorch.

The difference between reference points $\vec{e} = \vec{r}_{opt} - \vec{r}_{rnn}$ computed through optimization and the Neural Network is summarized in Figure 10. The majority of the results are near zero. The mean value and standard deviation are $(\bar{e}_1, \sigma_1) = (0.0000, 0.0421)$ and $(\bar{e}_2, \sigma_2) = (-0.0007, 0.0050)$ for r_1 and r_2, respectively.

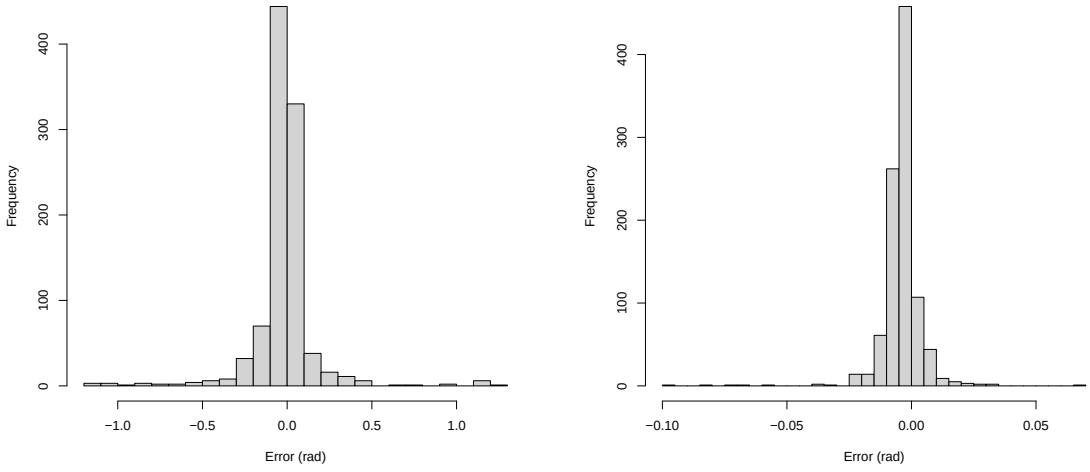
Continuing with the analysis of the second joint, the mean value and the standard deviation are $(\bar{e}_3, \sigma_3) = (-0.0152, 0.1937)$ and $(\bar{e}_4, \sigma_4) = (-0.0039, 0.0085)$ for r_3 and r_4, respectively. The histogram is shown in Figure 11.

The analysis of the third joint led to the mean value and standard deviation of $(\bar{e}_5, \sigma_5) =$



(a) Point r_1 (b) Point r_2

Figure 10: Historical error histogram of joint 1.



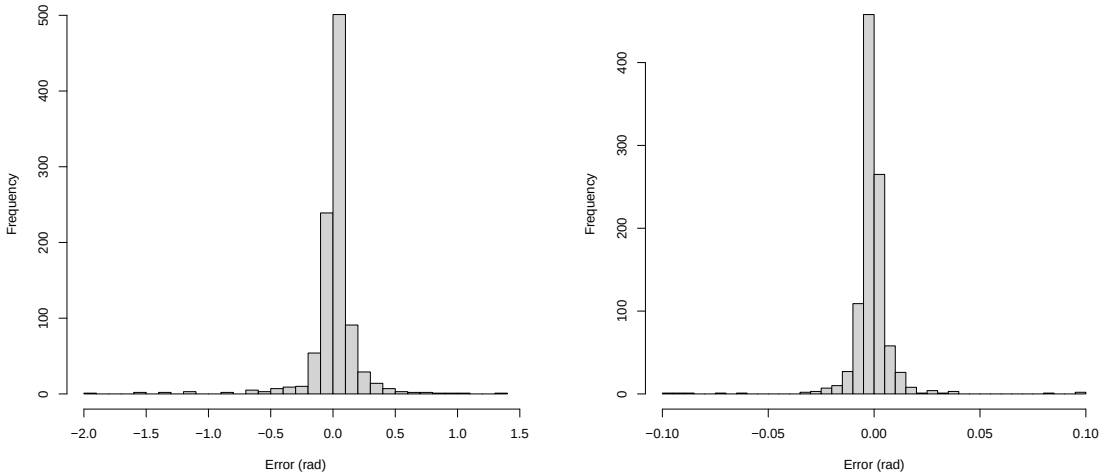
(a) Point r_3 (b) Point r_4

Figure 11: Historical error histogram of joint 2.

$(0.0126, 0.2007)$ and $(\bar{e}_6, \sigma_6) = (-0.0011, 0.0102)$ for the r_5 and r_6 reference points, respectively. The histogram is shown in Figure 12.

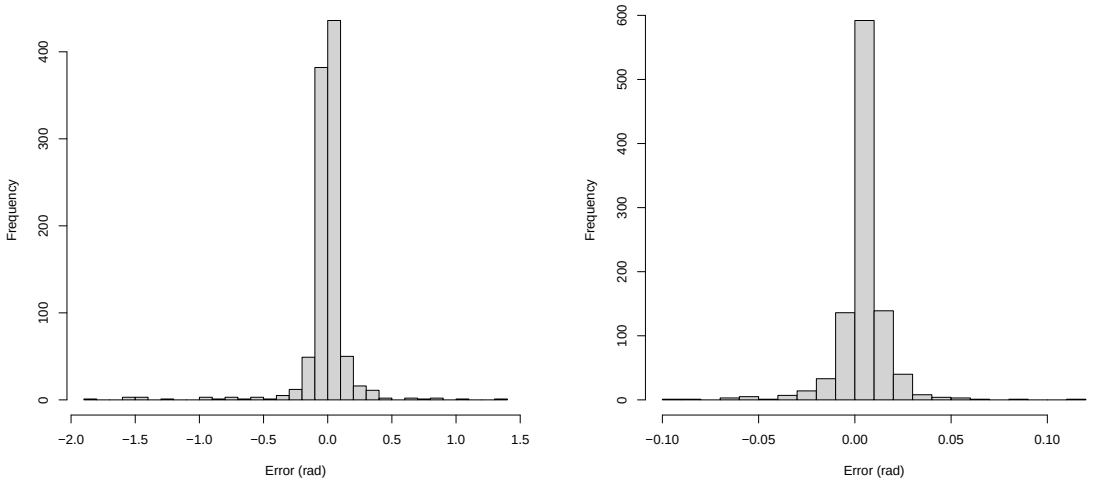
Finally, the analysis of the fourth joint resulted in a mean value and standard deviation of $(\bar{e}_7, \sigma_7) = (-0.0106, 0.1944)$ and $(\bar{e}_8, \sigma_8) = (0.0046, 0.0132)$ for the reference points r_7 and r_8, respectively. The histogram is shown in Figure 13.

The standard deviations at the end times $\sigma_2, \sigma_4, \sigma_6$ and σ_8 are smaller than their intermediate time counterparts ($\sigma_1, \sigma_3, \sigma_5$ and σ_7). It means that the calculation of the reference point at the final time (r_2, r_4, r_6 and r_8) is more accurate than the estimated value at the



(a) Point r_5 (b) Point r_6

Figure 12: Historical error histogram of joint 3.



(a) Point r_7 (b) Point r_8

Figure 13: Historical error histogram of joint 4.

intermediate time. It is important from a practical point of view because the joint angle at the end time is critical to the accuracy of the end-effector towards the target.

A comparison of the error dispersion between all reference points is presented in Figure 14. Note that the median and quartiles follow a similar trend as discussed in the previous paragraph.

The accumulated error of the final time reference points (r_2, r_4, r_6 and r_8) are shown in Figure 15. The biggest error was found in the last reference point r_8. A valid approach to mitigating this problem is to investigate the existence of a custom neural network archi-

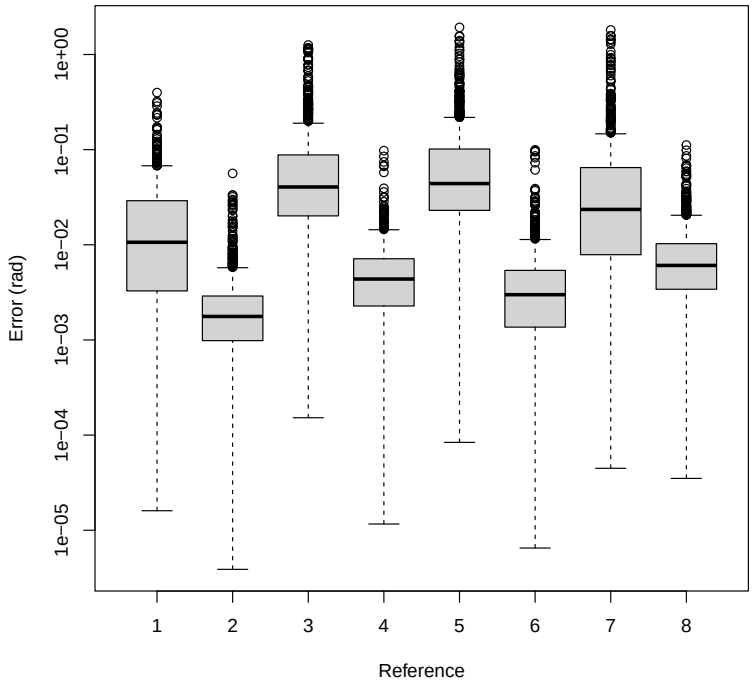


Figure 14: Comparison of error dispersion along the reference points.

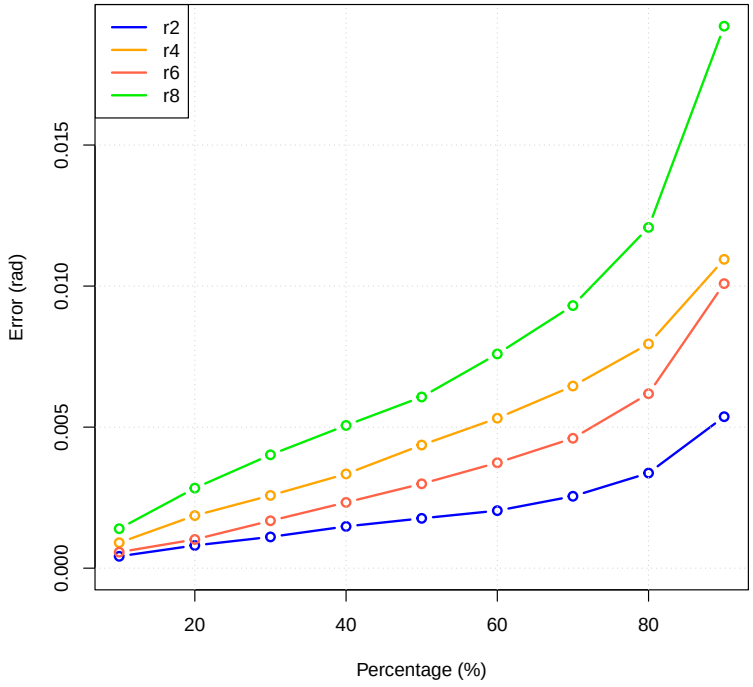


Figure 15: Cumulative error in multiple reference points.

tecture that improves the quality of results. In the present study, the same neural network architecture was used as a meta-model for all reference points. Although an independent training procedure was performed to estimate the weights of the network used in the calculations of each reference r_i, it was found that the quality of the model is different in each case.

As a final observation, the results of a neural network proved to be useful to estimate the value of reference points that will meet the inverse kinematics criteria. Despite the fact that in specific circumstances a large error can occur, this methodology is really efficient in providing a good estimate for the optimization procedure. As a result, the combined use of a Neural Network (for a robust estimation) and the technique of Nonlinear Programming (for an accurate solution) is recommended as a powerful tool in the path planning of space robots.

8 Conclusion

The present study analyzed the uncertainty that arises in the movement of a space robot composed of a satellite and a serial manipulator. The first model described the robot kinematics using rotation and translation matrices. No force effects are considered.

The second model used a multibody formulation written in OpenModelica software. In this case, forces and torques are transmitted between the robot and the satellite. The end-effector positioning error in both cases was compared.

A Neural Network was trained on the PyTorch platform to identify disparities between models and learn from historical data. The trained model was found to be a robust predictor of joint angle values. In most cases, it gives a result with sufficient accuracy.

The Nonlinear Programming methodology benefited from this estimate as initial guess for the optimization process. The procedure is efficient in providing accurate calculations for inverse kinematics computation.

It was confirmed that the combined use of Neural Network and Nonlinear Programming strategies is an efficient approach for the calculations of inverse kinematics, which increases robustness and speeds up the process.

Acknowledgments

D.A. Rade acknowledges the Brazilian research agencies Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) (grant #2018 / 15894-0) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (grant #312068 / 2020-4) for the financial support to his research work. Ijar M. da Fonseca acknowledges the Brazilian Space Agency (AEB-UNIESPACO, grant #104006129).

References

G. S. Aglietti, B. Taylor, S. Fellowes, T. Salmon, I. Retat, A. Hall, T. Chabot, A. Pisseloup, C. Cox, A. Mafficini, et al. The active space debris removal mission removedebris. part 2: in orbit operations. *Acta Astronautica*, 168:310–322, 2020.

L. E. Briese, A. Klöckner, and M. Reiner. The dlr environment library for multi-disciplinary aerospace applications. In *Proceedings of the 12th International Modelica Conference, Prague, Czech Republic, May 15-17, 2017*, number 132, pages 929–938. Linköping University Electronic Press, 2017.

F. Chen, G. Zhao, Y. Yu, J. Wang, and P. Cao. Modelica-based modeling and simulation of satellite on-orbit deployment and attitude control. In *4th International Conference on Machinery, Materials and Information Technology Applications*, pages 196–204. Atlantis Press, 2017.

S. B. Choze, R. R. Santos, and G. F. Gomes. Overview of traditional and recent heuristic optimization methods. In *Model-Based and Signal-Based Inverse Methods*, volume 1 of *Discrete Models, Inverse Methods & Uncertainty Modeling in Structural Integrity*, chapter 4, pages 107–142. University of Brasilia, 2022a. URL `https://doi.org/10.4322/978-65-86503-71-5.c04`.

S. B. Choze, R. R. Santos, A. B. Jorge, and G. F. Gomes. An overview of linear and non-linear programming methods for structural optimization. In *Model-Based and Signal-Based Inverse Methods*, volume 1 of *Discrete Models, Inverse Methods & Uncertainty Modeling in Structural Integrity*, chapter 3, pages 65–106. University of Brasilia, 2022b. URL `https://doi.org/10.4322/978-65-86503-71-5.c03`.

J. J. Craig. *Introduction to robotics: Mechanics and control, 3rd Edition*. Pearson Education, 2005.

I. M. da Fonseca. Space robotics and associated space applications. In *Vibration Engineering and Technology of Machinery*, pages 151–170. Springer, 2021.

I. M. da Fonseca, L. M. Unfried, G. L. B. Lima, M. R. da Silva, O. Satome, and E. J. de Oliveira. Space robot dynamic analysis of the relative orbital and attitude motion in the close range rendezvous phase and grasping of a target space vehicle. In *69th International Astronautical Congress (IAC)*. 2018.

I. M. da Fonseca, M. N. Pontuschka, and G. L. B. Lima. Kinematics for spacecraft-type robotic manipulators. In *Kinematics-Analysis and Applications*. IntechOpen, 2019.

R. R. dos Santos, V. Steffen, Jr., and S. de Fatima Pereira Saramago. Solving the inverse kinematics problem through performance index optimization. In *Proceedings of the XXVI Iberian Latin-American Congress on Computational Methods in Engineering CILAMCE 2005*. Brazilian Assoc. For Comp. Mechanics & Latin American Assoc. of Comp. Methods in Engineering, 2005.

R. R. dos Santos, V. Steffen Jr., and S. d. F. Saramago. Robot path planning in a constrained workspace by using optimal control techniques. *Multibody System Dynamics*, 19(1):159–177, 2008a.

R. R. dos Santos, V. Steffen Jr., and S. d. F. P. Saramago. Optimal path planning and task adjustment for cooperative flexible manipulators. In *ABCM Symposium Series in Mechatronics*, volume 3, pages 236–245. 2008b.

R. R. dos Santos, V. Steffen, and S. d. F. P. Saramago. Optimal task placement of a serial robot manipulator for manipulability and mechanical power optimization. *Intelligent Information Management*, 2:512–525, 2010. doi: 10.4236/iim.2010.29061.

H. Elmqvist, S. E. Mattsson, and M. Otter. Modelica: The new object-oriented modeling language. In *12th European Simulation Multiconference, Manchester, UK*, volume 5, 1998.

ESPI. Espi report 71 - towards a european approach to space traffic management - full report, 2020a.

ESPI. Espi report 72 - europe, space and defence - full report, 2020b.

W. Fehse. *Automated rendezvous and docking of spacecraft*, volume 16. Cambridge university press, 2003.

I. M. D. Fonseca, P. Gasbarri, E. F. Moura, and R. R. Santos. Trajectory generation for an on-orbit robot manipulator. In *Proceedings of the 72nd International Astronautical Congress (IAC)*. IAC, 2021.

P. Fritzson, P. Aronsson, H. Lundvall, K. NystrÃ¶m, A. Pop, L. Saldamli, and D. Broman. The OpenModelica Modeling, Simulation, and Software Development Environment. *Simulation News Europe*, 15(44/45):8–16, Dec. 2005.

P. Fritzson, P. Aronsson, A. Pop, H. Lundvall, K. Nystrom, L. Saldamli, D. Broman, and A. Sandholm. Openmodelica - a free open-source environment for system modeling, simulation, and teaching. In *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control*, pages 1588–1595, 2006. doi: 10.1109/CACSD-CCA-ISIC.2006.4776878.

S. G. Johnson. The NLopt nonlinear-optimization package. 2020. URL `https://cran.r-project.org/package=nloptr`.

D. Kuang, S. Desai, and A. Sibois. Observed features of gps block iif satellite yaw maneuvers and corresponding modeling. *GPS solutions*, 21(2):739–745, 2017.

S. M. Le Grand and K. M. Merz Jr. The application of the genetic algorithm to the minimization of potential energy functions. *Journal of Global Optimization*, 3(1):49 – 66, 1993.

G. Looye. The new dlr flight dynamics library. In *Proceedings of the 6th International Modelica Conference*, volume 1, pages 193–202, 2008.

M. Lovera and T. Pulecchi. Object-oriented modelling for spacecraft dynamics: A case study. In *2006 IEEE Conference on Computer Aided Control System Design,*

2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, pages 1898–1903, 2006. doi: 10.1109/CACSD-CCA-ISIC.2006.4776930.

D. G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. Springer, 3rd edition, 2008.

D. Moorman and G. Looye. The modelica flight dynamics library. In *Proceedings of the 2nd International Modelica Conference, Oberpfaffenhofen, Germany*, 2002.

P. Mróz, A. Otarola, T. A. Prince, R. Dekany, D. A. Duev, M. J. Graham, S. L. Groom, F. J. Masci, and M. S. Medford. Impact of the spacex starlink satellites on the zwicky transient facility survey observations. *The Astrophysical Journal Letters*, 924(2):L30, 2022.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

L. Petzold. Dassl. a differential/algebraic system solver. Technical report, Lawrence Livermore National Lab., CA (United States), 1982.

T. Pulecchi and M. Lovera. A modelica library for space flight dynamics. In *In Proceedings of the 5th International Modelica Conference*. Citeseer, 2006.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020. URL `https://www.R-project.org/`.

H. E. Romeijn and R. L. Smith. Simulated annealing for constrained global optimization. *Journal of Global Optimization*, 5(2):101 – 126, 1994.

R. R. Santos, D. A. Rade, and I. M. da Fonseca. A machine learning strategy for optimal path planning of space robotic manipulator in on-orbit servicing. *Acta Astronautica*, 191:41–54, 2022.

S. F. P. Saramago and V. Steffen Jr. Optimization of the trajectory planning of robot manipulators taking into-account the dynamics of the system. *Mechanism and Machine Theory*, 33(7):883–894, 1998.

M. A. Skinner, T. M. Kelecy, S. A. Gregory, J. P. Toth, D. Liang, D. Yamanaka, S. Kent, R. Tjoelker, D. Margineantu, A. L. Allison, et al. Commercial space situational awareness: an investigation of ground-based ssa concepts to support commercial geo satellite operators. *Proc. AMOS*, 2013.

M. Smith, D. Craig, N. Herrmann, E. Mahoney, J. Krezel, N. McIntyre, and K. Goodliff. The artemis program: an overview of nasa's activities to return humans to the moon. In *2020 IEEE Aerospace Conference*, pages 1–10. IEEE, 2020.

M. W. Spong and M. Vidyasagar. *Robot Dynamics and Control*. John Wiley and Sons, 1989.

V. Steffen Jr. and S. F. P. Saramago. Optimization techniques for off-line trajectories planning of robot manipulators. *Non-linear Dynamics, Chaos, Control and their Applications in Engineering Sciences*, 1:363–368, 1997.

R. Storn and K. Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341 – 359, 1997.

M. Uchiyama. A study of computer control of a mechanical arm. *Bulletin of the Japan Society of Mechanical Engineers*, 22:1640–1647, 1979.