Politecnico Di Torino

Master's Degree in Mechatronic engineering

Challenges and Innovations in OSAM through a novel Space Robot proposal

Candidate
Daniela Belvedere

Tutore
prof. Stefano Mauro
Cotutore
ing. Pierpaolo Palmieri

Year
2022/2023

So che ti ho già dedicato la prima tesi,
ma ci sarà un tuo aforisma anche in questa.
Perché anche se non ci sei più,
sono sicura che hai continuato a credere in me.
Ti voglio bene Nonno.

"Ricorda, meglio avere il sedere gelato, che un gelato nel sedere "

Non aveva detto sedere.

Contents

List of Figures

List of Tables

Introduction

By definition, a *robot* is an automaton which, following its programming, simplifies the tasks that a human being should perform in some sectors of industry or research. Depending on the sector that is considered, robots must have certain characteristics that are different from each other. This is because they have to replace human beings in their task and they cannot have the same accuracy in building a robot that works in the medical field or that works in the domestic sector.

Figure 1: NASA Canadarm2

The sector that is most under development is the space robot sector. Humans have always tried to go beyond their limits and challenge the boundaries and space exploration has made great progress in recent years. The substantial problem is that, in addition to being the field that most fascinates, it is also the field with the most limits as far as the work of a human being is concerned. Just think of the problems related to the earth's atmosphere: the working conditions of man in a field of this type make the actual performance very difficult if not impossible, lack of oxygen and absence of gravity are just some of the main problems that you can have. For this reason, using a robot that performs the tasks instead of the human being can greatly simplify the task that needs to be performed.

Two of the most current robots are the *Perseverance* rover and the *Ingenuity* helicopter drone, which are doing a huge job with regards to the exploration of Mars. NASA launched on July 30, 2020 and they landed on February 18, 2021 following what NASA calls "seven minutes of terror" in which there was no radio communication.

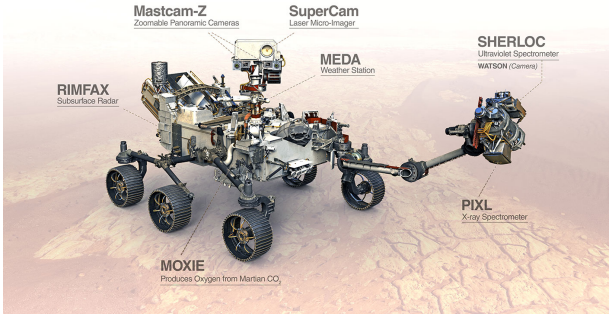

Figure 2: NASA Perseverance Rover

The space mission connected to the two is an exploration mission, consequently the main objectives are to study the surface of Mars, primarily if there are biological traces. Adding other objectives such as understanding climatic conditions or preparing for possible subsequent human exploration. The robots have been on Mars for 780 suns and Perseverance sends about 7000 photos to the earth in a month, it goes without saying that it is clear that a human would not have been able to perform the same task so efficiently.

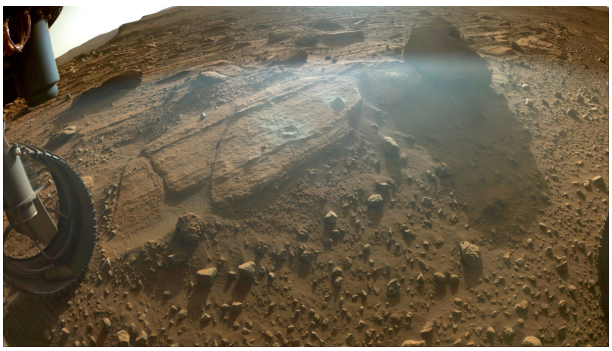

Figure 3: Perseverance's photo for NASA dated March 31, 2023

Another NASA project is the *Robonauts*, a robot belonging to the category of humanoids and no longer rovers. Considering that the latest version of the robot was completed on February 24, 2011, the similarity of this prototype to astronauts was something impressive. The main purpose was that of

"assistant": to carry out the tasks that would have been heavy and complex for man. The latest version created could be controlled both by astronauts and from the ground and did not need constant supervision. Even Robonauts, such as Perseverance and Ingenuity has greatly simplified what were the tasks if they had been performed only by humans.

Figure 4: NASA Robonauts 2

Working in conditions other than terrestrial ones, the most fundamental part in their construction is rigidity, which contrasts with the need to be light to ensure their transport on a spacecraft. One of the problems that can arise when creating structures operating in an environment that is clearly different from the Earth's atmosphere is the size of the object to be used. Having to ship these robots in spaceships, we try to have the structure with a weight and size that are as small as possible. For example, a solution to the most current problem is that of inflatable robots, which would greatly limit the weight of the manipulator, allowing it to be transported in a compact manner, even occupying a relatively small space. The study that is carried out for these manipulators is mainly related to its rigidity since, having a partially elastic structure to allow inflation, the robot has a low structural rigidity making its task problematic if subjected to strong pressures, oscillations, load lifting or limitations due to gravitational force. But this is only

one of the many difficulties encountered in this specific field of study.
The purpose of this thesis is to offer a comprehensive and detailed overview the control linked to a space mission. First considering the journey to bring the manipulator into orbit and then for the robot itself, dividing the research journey into three key sections. Initially, the exploration focuses on the structure of aerospace robots, with a particular emphasis on innovative concepts such as ISAM and Popup, aiming to provide a thorough understanding of these elements. The second section delves into orbital control strategies, introducing the reader to the intricate dynamics of space maneuvers. Finally, the third section addresses specific robot control, completing the overall framework. In summary, the thesis aims to provide in-depth knowledge of key elements in aerospace robotics, followed by a clear understanding of orbital control strategies and specific robot control.

Chapter 1

Robotics and Aerospace: Structures, ISAM, and Popup

This first paragraph will deal with the structure of robots, underlining the already mentioned difference between those who are manipulators forced to work in a hostile environment, such as space, and classic terrestrial robots. It should be underlined that robotics opens the doors to a world that studies the interaction between intelligent machines and human beings. The main aim is always to make work easier. In fact, robots are used in many advanced sectors, such as surgery and industrial production. Space exploration is only a small part of this huge study. The thing they have in common is an improvement in efficiency and precision.

The way to operate a manipulator is the combination of specific hardware

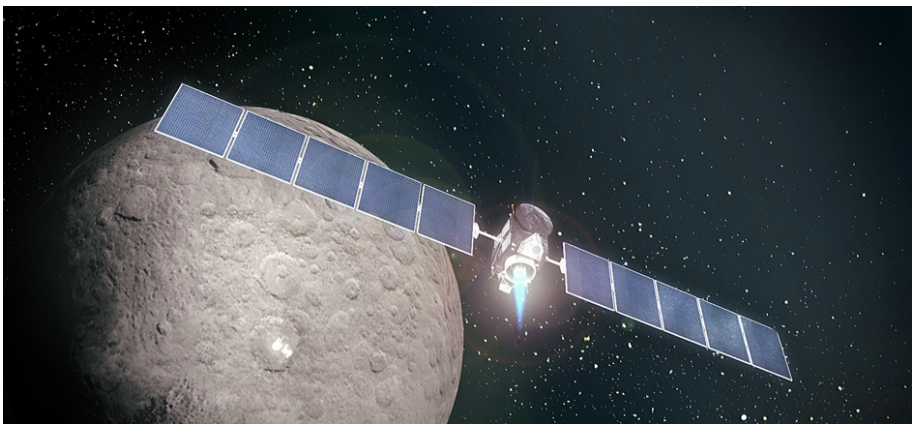

Figure 1.1: Space probes Dawn

and software. In this discussion the software used is the coupling of Matlab

and Simulink, mainly for simplicity. In this chapter we will discuss the hardware part, the structure of a manipulator. However, it should be underlined that the heart of a robot is the control system, since it guarantees the reception of information from the sensors and transfers the actions that the robot must carry out via actuators and motors. Without the controls part, robotics does not exist. The entire visual servoing part will also be covered. In this branch, the use of cameras that allow the robot to collect key information is fundamental.
Robots are diversified into vast categories depending on their use. Space robots are a category of robotic systems designed to operate in a very rigorous and challenging environment of outer space. These devices are definitely important in space exploration since the environment is hostile even for humans, so they are able to collect information and scientific data much more easily. A striking example are space rovers, such as the aforementioned Perseverance, they are vehicles equipped with wheels or legs that can explore surfaces. Clearly they are equipped with sensors and are able to analyze soil samples and detect chemical compositions very easily. Another example is space probes. Which are designed to carry out exploration and observation missions. Let us try to create a list with the major examples that we will discuss during this thesis.

- *Rover*, as already mentioned, mobile vehicles that move on the surface of planets in order to explore and conduct scientific experiments. Examples are the aforementioned Perseverance and Curiosity, which use projects to explore the surface on Mars.

- *Space probes*, unmanned vehicles. Also in this case they are used to explore space and collect scientific data but they move by orbiting. Some examples are the Dawn probe and the Voyager probe.

- *Artificial satellites* orbiting plants and moons. Their main purpose in this case is to communicate and observe the earth. The most famous is the Hubble satellite, the cylindrical space telescope that orbits the Earth.

- *Robotic arms*, i.e. mechanical extensions used to manipulate objects or perform specific tasks. Such as the Canadarm on the Space Shuttle, which carries out assembly activities.

- *Landers and elevators* for landing on the surface of a celestial body.

- *Social Robots*, such as the CIMON robot which was sent to the International Space Station to assist astronauts in daily activities and

also to provide emotional support. They are designed to interact with humans.

- *In-orbit assembly systems*, used to assemble and repair space components directly in orbit, such as the Robotic Refueling Mission to replenish fuel.

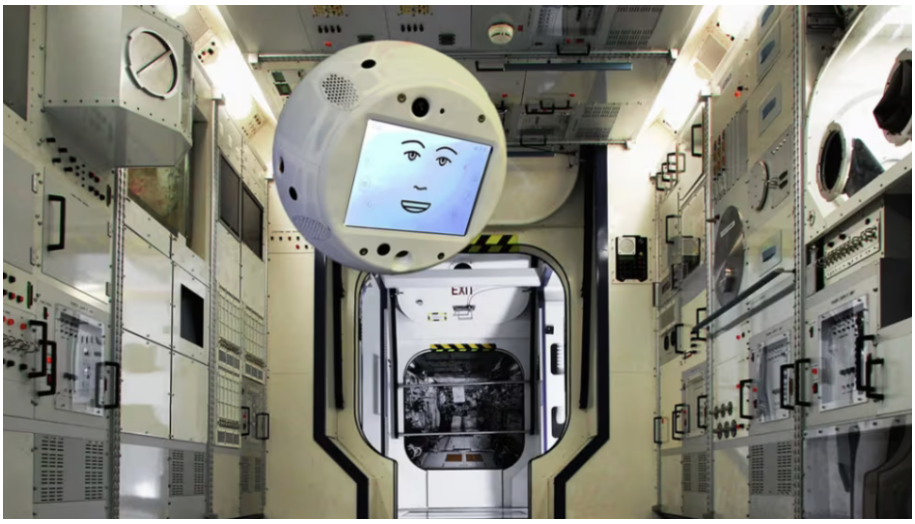

Figure 1.2: Social robot CIMON

1.1 Robot structure

Let's get into the specifics and deal with the structure of a robot. First of all, let's analyze what the different parts could be. As regards the purely mechanical part of the manipulator, the presence of arms, joints, end effectors and the base must be highlighted. The former are mainly responsible for movement, they connect the joints which are nothing other than the joints of the robots. Joints are distinguished based on the movement they provide: rotoidal if they guarantee rotation, prismatic if they guarantee translation and spherical if they allow movement in all directions. The last two represent the final and initial part of the manipulator respectively. The end effector is the end of the robot that interacts with the external environment; it normally features a tool suitable for a specific task (such as a socket or pliers). The base is the bottom part of the robot and serves as a support. Tracks and wheels must be installed at the base when the robot has the possibility of moving. Other elements to highlight are the sensors, which detect information about

the surrounding environment (such as the camera or position sensors). The latter are the eyes of the software system managed by the controller that coordinates the movement of the joints and control the end effector.

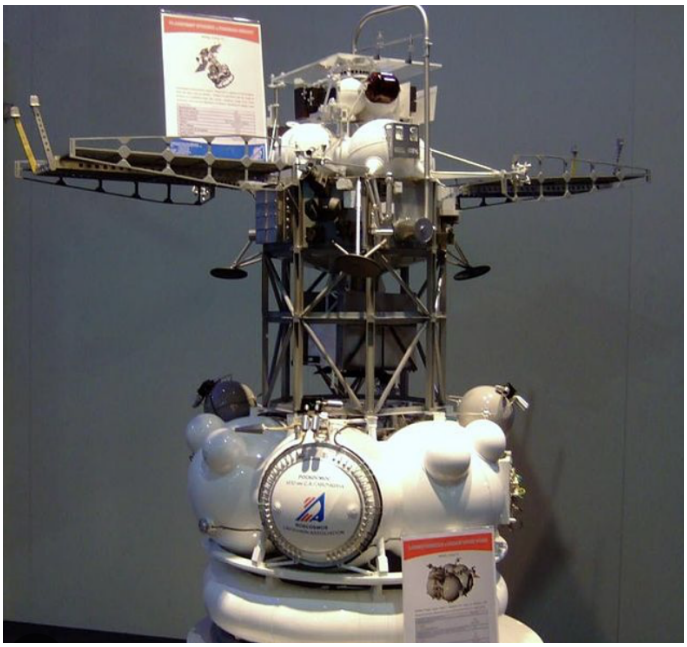

Figure 1.3: Phobos-Grunt

Let's now go into detail and see the typical robot configurations.

1. **SCARA**, characterized by two parallel rotoidal joints to allow movement in the plane and a prismatic joint for vertical movement. An example is the ISS robotic arm which uses this configuration for assembly and maintenance.

2. **Anthropomorphic** which sees a design very similar to the human one. The joints provide a wide range of motion. The aforementioned Robonaut is the most famous example.

3. **Cylindrical**, like the arm of the Curiosity rover. Its structure features rotoidal joints that allow circular movements along the vertical axis and a prismatic joint for vertical movement.

4. **Delta**, with parallel arms towards a common point. Normally carried out picking and packing application. The Phobos-Grunt mission used Phobos, a delta-type robot to explore the Mars satellite.

5. **Six-axis**, they feature six independent rotary joints that allow a very wide range of movements. The Canadarm2 is the prime example.

6. **Polar**. The name indicates that it uses a polar coordinate system to specify location. Like the robotic arm of the Phoenix Mars Lander mission that landed on Mars in 2008.

Figure 1.4: Phoenix Mars Lander

1.1.1 Space vs Terrestrial robots

Although the types of structures may be the same, the robots that operate in space have completely different software and hardware characteristics. The changes are due to the different environment and conditions in which they operate. The main differences are:

- *Cosmic radiation.* In space, robots are exposed to solar radiation which damages electronics and all sensitive components. This requires, as a first example, a design of the manipulator with more resistant materials so as to be able to shield the electronic components and protect them from cosmic radiation. An addition is made by highly efficient cooling systems.

- *Temperature.* Related to the previous problem, temperatures create high temperature changes. You can go from a decidedly hot area to a decidedly cold one in a short time.

- *Empty.* The lack of atmosphere clearly changes the movement methodology: there is no air support either for movement-related calculations or for heat dissipation.
- *Microgravity.* To avoid navigation and control problems, specific control systems are installed to compensate for the effects of microgravity and guarantee precise maneuvers.
- *Autonomy.* For delayed communication with the earth it is essential to have great control of the autonomy. Not to mention that they are robots that operate at a great distance from planet earth and therefore often have to solve problems without the supervision of ground operators.
- *Energy supply.* The lack of power outlets requires large power systems.
- *Spatial Debris.* Efficient trajectory planning to avoid the presence of debris and unwanted collisions.
- *Mission duration.* Guaranteed component longevity and reliability to meet long-term space challenges.

1.2 In-space Servicing, Assembly and Manufacturing

From the in-depth analysis of the intricate structures of robots, the foundational pillars emerge that underpin the crucial role of robotics in the realm of **In-space Servicing, Assembly, and Manufacturing (ISAM)**, a field of study that will revolutionize the way space activities are carried out. The robust characteristics and precise design of robotic structures prove to be not only fundamental requirements but also key elements that enable the overcoming of the intricate challenges posed by ISAM. This new frontier of space is fundamental when we go into a comprehensive study of how to create a manipulator that operates in space. It is not enough to consider its structure or its autonomous control, it is also necessary to optimize its way of working. ISAM aims to overcome conventional limitations by adding flexibility and sustainability to the aerospace context.
To go into specifics, we can begin the discussion by underlining that ISAM is an advanced discipline that deals with the design, development and implementation of technologies inherent to, as the name suggests, service, assembly and production of space components. Specifically, the three fundamental components of this sector are:

- *In-Space Servicing (ISS)* refers to the use of robotic spacecraft to perform specific operations, such as repair, maintenance to correct damage or malfunction, refueling.
- *In-Space Assembly (ISA)* refers to the ability to assemble pieces directly in space, very useful for building large structures.
- *In-Space Manufactoring (ISM)* refers to the actual production of components directly in space. Such as spare parts.

It is clear that this new frontier is being developed and studied for the simple reason that it has various advantages. Costs can be reduced, for example, since it is not necessary to send new satellites commissioned for a short time just to replace faulty ones. Or to build large structures, sending them into space by transporting them from the earth would be complicated and sometimes impossible. The main purpose, however, is operational sustainability which minimizes the use of the earth for carrying out multiple operations, directly exploiting the space resources we possess.

NASA has been trying to carry out this particular type of application for 55 years.

Figure 1.5: 55 years of ISAM

The main reason is linked to *microgravity*. Basically a point is used which should be a negative in favor of spatial operations. They are offered a better working environment that allows them to do jobs that would be impossible to do on earth. Such as the mixing of superalloys which is much more precise. Then the speed is greatly improved if the pieces are created directly in space and as needed, also avoiding having many useless stocks created previously. Even the use of very thin or light structures can be carried out without gravity while they would be impossible to use if we were in space (such as space telescopes). [16] [34]

A very important NASA project related to ISAM is *On-orbit Servicing, Assembly and Manufactoring 2* (OSAM-2), generally known as Archinaut. This project does exactly what we talked about above, it tries to develop manufacturing technology directly in space. First of all, ground tests were carried out between 2016 and 2019. The structural beams were 3D printed by simulating the pressure and temperature conditions of space at the NASA Ames Research Center in Silicon Valley, demonstrating that it is possible to work even in these adverse conditions. In 2018, further simulations were carried out to verify whether the assembly and production capacities were effective. In 2020 and 2022 respectively, Redwire first demonstrated the hardware's ability to print a seven-meter beam under expected gravitational conditions and then passed the Critical Mission Design Review (CDR), marking the end of the design and giving the off to construction.
Archinaut 1 will be launched into space in 2024. [35] [36]

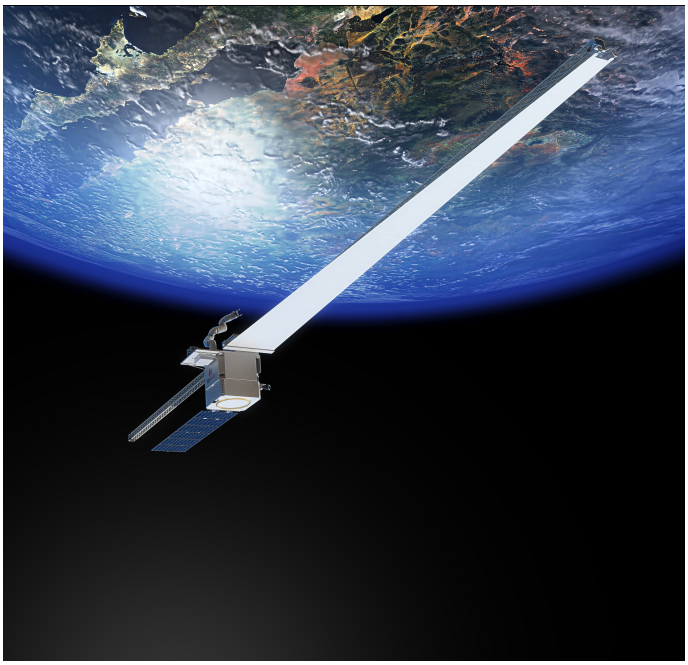

Figure 1.6: Archinaut 1

1.2.1 Space Lab

An honorable mention regarding ISAM related missions must be made to the mission that started all the other projects: the *Space Lab*, mission which began on November 28, 1983 on the Space Shuttle Columbia (STS-9) .

The main purpose of the mission was to conduct certain scientific experiments directly in space, exploiting and using extra-terrestrial atmospheric conditions, such as the microgravity we have already mentioned. The idea of the mission was to collect data and information after arriving in orbit and, only after landing, reconsider all the data and integrate what had already been studied in orbit.

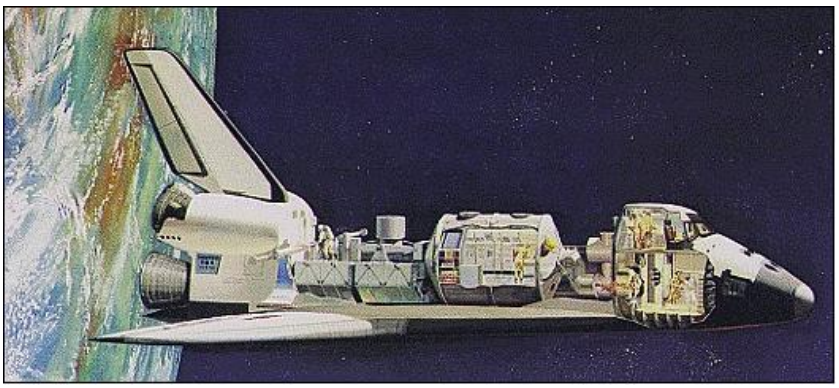

Figure 1.7: Space Lab

Specifically, there was a pressurized module installed in the shuttle that allowed a team, made up of six astronauts, to conduct scientific experiments in laboratories that possessed all the necessary scientific instruments and equipment. Communications were via the Orbiter system and the Tracking and Data Relay Satellite (TDRSS).

It is clear that this mission is too old and cannot be included within the actual ISAM missions, also because there were not the usual operations carried out in this particular group. In any case, it was among the first missions to carry out a particular task directly in orbit. For this reason it needs an honorable mention. [9] [10]

1.3 Popup Robot: description and model

One of the main problem to the robots is connected to the structure, in particular is related to the weight and to the size. There's an innovative approach allows the robot to efficiently utilize available space and resources in space missions: the **Popup** inflatable robot is designed to address the challenges of microgravity and the limited space inside spacecraft. Here's how the inflatable design helps solve these problems:

- Volume Minimization during Launch: In space, volume is a critical factor in costs and resource management. The inflatable design allows

the robot to be compressed into a much smaller space during launch. Once in space, it can be inflated to full operational size, maximizing the use of available space.

- Adaptability to Space Constraints: The inflation process allows the robot to adapt to space constraints during launch. In microgravity environments, where there is no defined "up" or "down" direction, the inflatable design allows the robot to expand flexibly, regardless of a specific orientation.

- Reduced Weight: Materials used for constructing inflatable robots are often lightweight, helping to keep the overall payload weight to a minimum. This is particularly crucial for reducing launch costs and maximizing the efficiency of space missions.

- Structural Robustness: Inflatable materials can be designed to be robust and resilient, despite their lightweight nature. This allows the robot to perform its intended functions once fully inflated and operational in space.

- Ease of Implementation: The inflation process can be automated, simplifying the implementation of the robot in space. It can be activated once in space and given the necessary time to reach full operational dimensions.

Before talking in particular, is necessary to give some definition which will be the "legend" for the whole discussion. The parts of a generic robot that will be considered during this work are:

- *Arms/links*, that are the idealized geometrical bars that connecting two or more joints;

- *Joints*, that are idealized physical components allowing a relative motion between the attached links. The type of joints can be "prismatic" when they allow a translation between the connected links; or "revolute" when they allow a rotation between the connected links. In the discussion of this particular soft robot is considered only the revolute joints.

- *End effector* is the structure at the end of the last link that is able to perform the required task or can hold a tool.

In this way, the integration of Popup Robots emerges as a key element in the complex ecosystem of In-space Servicing, Assembly, and Manufacturing

(ISAM). These dynamic and flexible devices, embedded within the broader context of space robotics, not only significantly expand operational capabilities but also offer versatile and adaptable solutions for specific challenges in the field of space. Their implementation, synergistic with the vision of ISAM, opens new perspectives for advanced services, precise assembly, and production processes in extraterrestrial space, outlining an innovative outlook in the landscape of space exploration and orbital operations.
The particular modeling structure is characterized by the two arms, which are the pieces of the robot capable of inflating and bending, and by the two revolute joints, specifically a wrist joint and a shoulder joint, that are actuated with 3 DC motors.

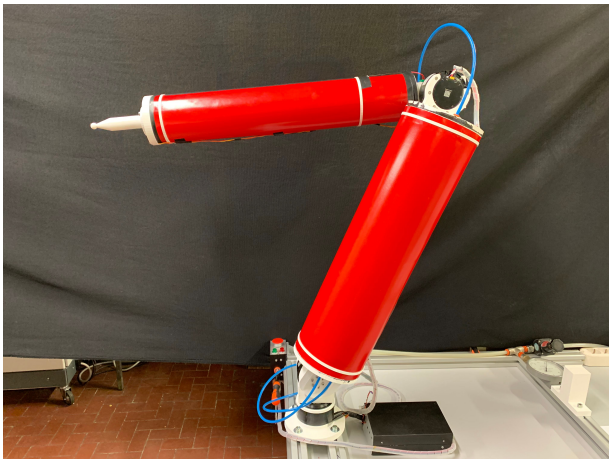

Figure 1.8: Popup Robot

The main advantages of the Popup Robot are adaptability, lightness and manageability, that are important characteristic when is talk about the space application where, like is said before, requires structures with decidedly reduced weight and volume. However, the structure, in addition to guaranteeing all these advantages, is also the cause of the main problems: considering the design of the soft manipulator, it is not always possible control it like a traditional rigid manipulator.

1.3.1 Kinematic consideration

The problem with this type of robot is dictated by the fact that in reality, there is not a rigid structure but a completely flexible one, which complicates the calculations and considerations to be made. However, we approach the model of the flexible robot step by step, starting by assuming an approximation of the robot link to a rigid link. The considerations about the kinematic

of the robot are done with reference to the *Denavit-Hartenberg convention* (D-H), that is a systematic procedure toward kinematics study of the manipulator. The solution is a table in which are reported, for all joints considered, some parameters: a_i, that represent the link length; α_i, that represent the link twist; d_i that represent the link offset; and θ_i that represent the joint angle. So, considering the fact that the Popup Robot have three joints and, for hypothesis, three rigid links, the scheme that is possible to do with the structure is designed in Fig. 1.9 and the parameters are reported in Table 1.1. this can be used for compute the homogeneous matrix that gives an estimate for the robot pose.

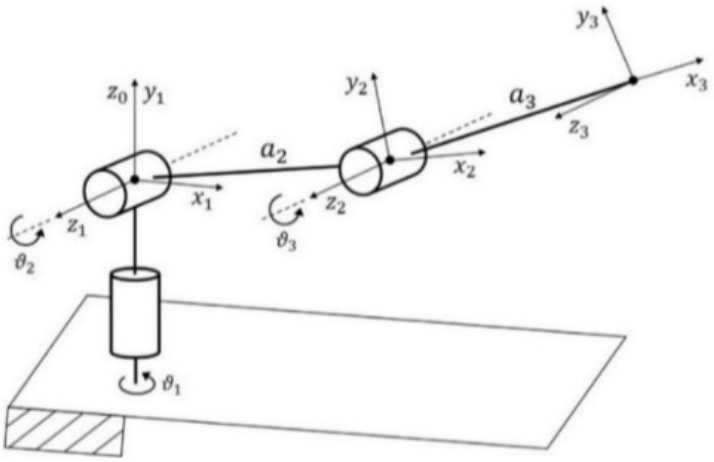

Figure 1.9: Kinematic scheme

Links	a_i	α_i	d_i	θ_i
1	0	$\pi/2$	0	θ_1
2	L	0	0	θ_2
3	L	0	0	θ_3

Table 1.1: Rigid Robot David-Hartenberg parameters

[20], [21], [22], [26]

Now let's consider the equations we just discussed without the approximation of rigidity. This allows us to readjust the study to the flexible model of the robot. However deformation during the movement of the robot cannot be considered negligible because the material of the structure is mouldable. Therefore, it is necessary to find a law capable of defining the dynamic behavior of the robot to simulate reality in the best possible way.

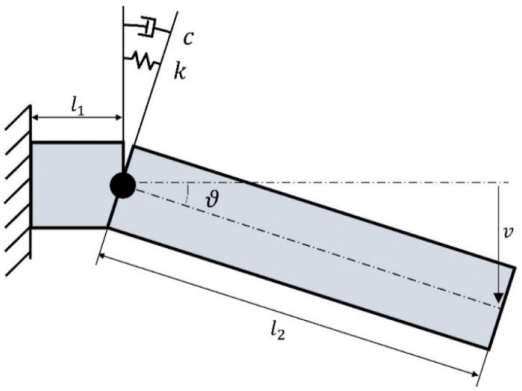

Figure 1.10: Pseudo-rigid body model, 2D scheme

In Fig. 1.10 is described the model[1] that highlight the consideration that the the links are considered like rigid structure with length, respectively l_1 and l_2. In the system are added a spring (k) and a dumper (c), for considering the flexibility. At least, θ represent the angular deflection and v is the extremity displacement.

The model is described by this formula

$$I\ddot{\theta} + c(p)\dot{\theta} + k(p)\theta = \tau$$

Where I is the moment of inertia, p the integral relative pressure and τ the external torque. Is underline the fact that stiffness and damping coefficient are depending on the internal pressure. This model is an approximation of a link of the Popup Robot, that consider the variable θ in the equation like the representative of the degree of freedom of the virtual joint.

The last aspect that is necessary to consider is the *wrinkling*[2] collapse following an external couple. The wrinkling moment M_w equation is given by

$$M_w = \frac{\pi^4}{4} p r^3 \tag{1.1}$$

where p is the pressure and r is the radius of the link.

To take these two equations into account, it is necessary to modify the scheme of the kinematic model considering the deflection around the two orthogonal axes.

In this way, the Popup Robot is compared to a manipulator with seven degrees of freedom, where three DOFs had already been introduced considering only the shoulder and the elbow, while the other four are obtained by

considering the virtual joints as links with a spring system. damper lying on an orthogonal axis.

The model is shown in Fig. 1.11 With the respective parameters of the D-H convention in the Table. 1.2

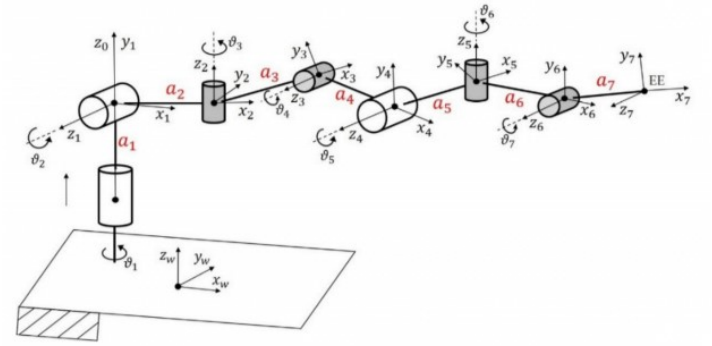

Figure 1.11: Kinematic scheme with additional degrees of freedom

Links	a_i	α_i	d_i	θ_i
1	0	$\pi/2$	0	θ_1
2	l_1	$-\pi/2$	0	θ_2
3	0	$\pi/2$	0	θ_3
4	l_2	0	0	θ_4
5	l_1	$-\pi/2$	0	θ_5
6	0	$\pi/2$	0	θ_6
7	l_2	0	0	θ_7

Table 1.2: Flexible Robot David-Hartenberg parameters

Let's underline that the model of the flexible robot have the joint variable that are $q = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7]^T$ that is divided in motor joint variable $q_m = [\theta_1, \theta_2, \theta_5]^T$ and virtual spring joint variables $q_k = [\theta_3, \theta_4, \theta_6, \theta_7]^T$.

[27] [15] [11]

Next step is to delve into orbital control and refine the management of inflatable robots, paving the way for a new era of advanced and flexible automation.

Chapter 2

Control strategies for spacecraft operation

In the context of sophisticated ISAM (In-space Servicing, Assembly, and Manufacturing) applications, the full operability of Popup Robots unfolds through an in-depth study dedicated to the hosting satellite. This analysis phase, paramount in the realm of orbital dynamics, is crucial to orchestrate the precise positioning of Popup Robots in orbit. Within this chapter, we will delve into how orbital trajectory control integrates into this intricate process. Through a comprehensive understanding of the satellite's orbital characteristics, we aim to outline advanced strategies and methodologies to optimize the integration of Popup Robots and ensure precise management of their activities during complex servicing, assembly, and production operations in extraterrestrial space. Remember that Orbital control refers to the precise management of the trajectory and position of an object in space. In robotics, orbital control is crucial to ensure that a satellite, spacecraft, or any other orbital device maintains its intended path. This may involve adjusting speed, orientation, and other parameters to ensure stability and achieve predefined objectives.

2.1 Laws of Physics

This class of physics is essential to be able to design, position and maintain the orbits in which artificial satellites operate. To date, manipulators capable of grasping objects that can move in an orbital trajectory have not yet been created, but for the discussion related to the trajectory lines associated with the cosmic companions, it is essential to deal with the orbital dynamics attributed to the satellites.

All these types of manipulators are considered as *rigid bodies* subjected to two types of movements: a translation of its center of mass (CoM) and a rotation of an axis passing through its center of mass. Substantially, orbital dynamics requires studying the ro-translation of the rigid body by exploiting in combination the three laws of Kepler and the three laws of Newton, which are reported below. The first three laws that are discussed were formulated

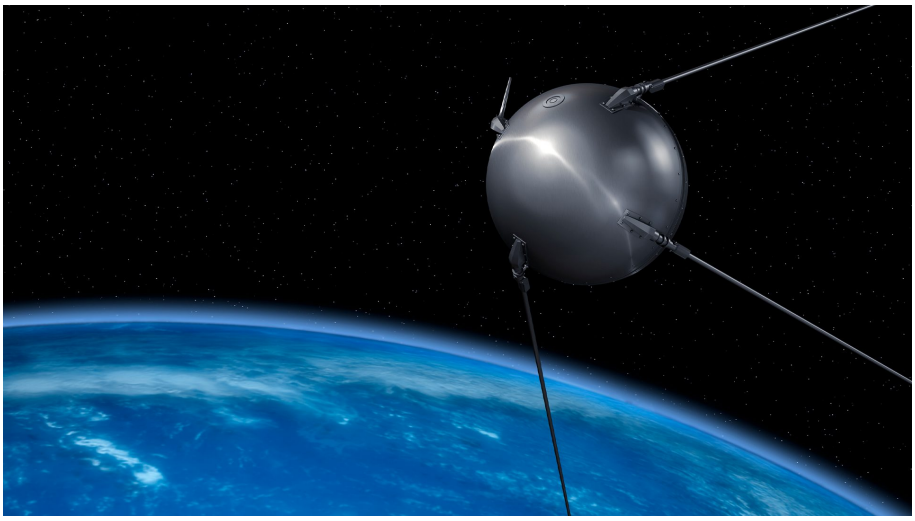

Figure 2.1: Sputnik 1: first artificial satellite

by Johannes Kepler in the first decade of the 17th century [18], these laws constitute a set of three principles that describe the orbital motion of celestial bodies. He certainly didn't expect that they would be used to create artificial satellites trying to match the motion of a natural satellite such as the moon. However, the three laws state that:

- The orbit of a planet is an ellipse with the sun located at one focus.
- The radius vector drawn from the sun to a planet sweeps out equal areas in equal time intervals (the areal velocity is constant).
- The square of the orbital period of a planet (or a satellite) is directly proportional to the cube of the semi-major axis of the orbit. In mathematical form, this law can be expressed as follows: $T^2 = k * a^3$

These laws provide important insights for designing the orbits of artificial satellites and for understanding orbital dynamics.
They were subsequently supported and taken up by Newton, who formulated three theories for the study of bodies:

- A particle remains at rest or continues to move at a constant velocity, unless acted upon by an external force.
- The rate of change of the linear momentum mv of a particle is given by
$$\frac{d}{dt}(mv) = F \tag{2.1}$$
where m is the particle mass, v is the particle velocity and F is the force acting on the particle.
- For any F_{12} exerted by a particle 1 on a particle 2, there exists a force
$$F_{12} = -F_{21} \tag{2.2}$$
exerted by particle 2 on particle 1.

In addition to these 6 laws in total, the law of universal gravitation must also be considered:

$$F = \frac{Gm_1m_2r}{r^3} \tag{2.3}$$

where m_1, m_2 are the particle masses, r is the vector of magnitude that connect the two particles and $G = 6,67x10^{-11}m^2/kg^2$ is the universal constant of gravitation.

These laws must be applied to the *two-body problem* which allows us to consider the problem related to the interaction between two bodies. This is because, when we deal with intra-terrestrial manipulators, the gravitational force is not the subject of the discussion for calculating the trajectory. But if we consider robots that must operate outside the atmosphere, perhaps even on other planets with a gravitational constant different from that of Earth, these laws must necessarily be considered. In particular the force of attraction between two bodies, since, if we consider a robot with a circular trajectory that interacts with another body inside planet earth, the masses will be very small and we will be able to ignore the force of attraction between them. But when we consider an object, such as a satellite, which rotates around a planet, the force of attraction between the two will be very large and must necessarily be considered. Having said this premise, in Fig. 2.3 we find a schematization of the aforementioned problem.

We will not go into the details of the calculations, what interests us is that in this way two equations can be obtained. The first describing relative motion

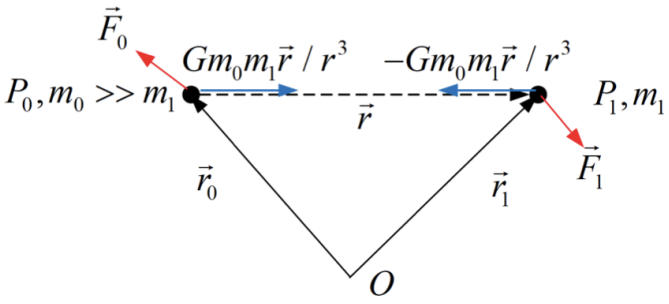

Figure 2.2: Two-body problem scheme

$$\dot{v} = -\frac{G(m_0 + m_1)}{r^3} r + \frac{1}{m_1}\left(F_1 - \frac{m_1}{m_0} F_0\right) \tag{2.4}$$

and the second describing the movement of the CoM

$$\dot{v_c} = \frac{1}{m_1} \frac{F_1 + F_0}{1 + m_0/m_1} \tag{2.5}$$

By considering one of the two masses significantly larger than the second, we transform the relative motion equation into the *restricted two-body equation*

$$\dot{v} + \mu \frac{r}{r^3} = \frac{1}{m_1} F_1 \tag{2.6}$$

where $\mu = G_m 0$ and is called *graivitational parameter*.

This model simplifies all calculations and procedures since it takes into account all the gravitational forces exerted by the two bodies on each other and does not take other bodies into account. The solution of this equation, in addition to allowing us to consider and calculate the energy of the system and the angular momentum, allows us to calculate the trajectory of the less massive body (such as a satellite) considering the movement of the more massive body almost unchanged.

Considering a parameter $p = h^2/\mu$ where h is the angular momentum we can write this equation starting from the previous one

$$r = \frac{p}{1 + e\cos\theta} \tag{2.7}$$

where e is the eccentricity. This equation, commonly called *Kepler's equation*, provides us with the shape of the orbital trajectory as a solution.

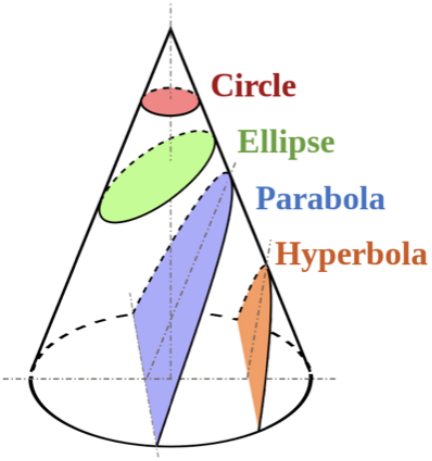

Figure 2.3: Conic section

Basically, depending on how the eccentricity parameter varies, the trajectory of the orbit varies.

The conisc section considered are:

1. **ellipse**, when $0 \leq e < 1$. An ellipse is a closed curve defined as the geometric locus of points whose ratio of distances from two fixed points, called foci, is constant and less than 1. The major and minor axes are the axes of the elliptic curve, and the sum of the distances from each point on the ellipse to the two foci is constant.

2. **parabola**, when $e = 1$. A parabola is an open curve defined as the geometric locus of points whose ratio of distances from a fixed point, called the focus, and a straight line, called the directrix, is constant. The parabola is symmetrical with respect to the axis perpendicular to the director passing through the focus. In this case we can compute the *escape velocity* $v_e = \sqrt{2\mu/r}$

3. **hyperbola**, when $e > 1$. A hyperbola is a curve defined as the geometric locus of points whose ratio of distances from two fixed points, called foci, is constant and greater than 1. The transverse and conjugate axes are the principal axes of the hyperbola, and the difference of distances from each point on the hyperbola to the two foci is constant.

Let's now see a concrete example covered on Matlab based on real values of NOAA-19 Satellite. [18]

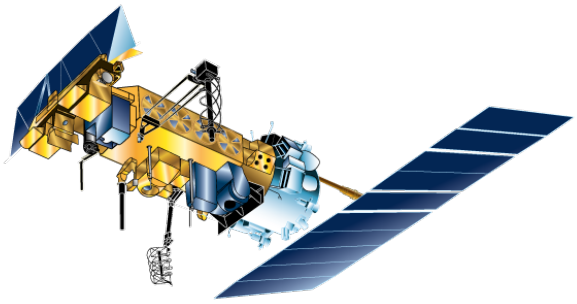

Figure 2.4: NOAA-19

```
clear all
close all
clc

%% Integration of the Orbital Equation for NOAA-19 Satellite

MU=3.986e14; % Gravitational parameter of Earth (m^3/s^2)
a=8.5e6; % Semimajor axis of satellite orbit (m)
rp=7.35e6; % Perigee radius of satellite orbit (m)
vp=sqrt(MU*(2/rp-1/a)); % Initial velocity of the satellite (m/s)

x0=[rp 0 0 0 vp 0]'; % Initial state vector [x y z vx vy vz]
tfin=2*pi*sqrt(a^3/MU); % Orbital period

t=linspace(0,tfin,1000);
[~,x]=ode23tb(@(t,x)fr2b(t,x,MU),t,x0);

%% Figure

lw1=1.2;
fs1=20;

figure('color','w')
hold on, grid
daspect([1 1 1])
xlabel('$x$␣[m]','interpreter','LaTex','fontsize',fs1)
ylabel('$y$␣[m]','interpreter','LaTex','fontsize',fs1)
zlabel('$z$␣[m]','interpreter','LaTex','fontsize',fs1)
xlim([-2 2]*9e6)
```

```
ylim([-2 2]*9e6)

plot3(0, 0, 0, '*r','markersize',14) % Earth
plot3(x(200,1), x(200,2), x(200,3), '.b','markersize',20) % NOAA-19
    Satellite
plot3(x(:,1), x(:,2), x(:,3), 'b','linewidth',lw1)
text(0, 0, 0, 'Earth','interpreter','LaTex','fontsize',fs1)
text(x(200,1), x(200,2), x(200,3), 'NOAA-19␣Satellite','interpreter
    ','LaTex','fontsize',fs1)
text(rp, 0, 0, '$\mathbf{x}_0$','interpreter','LaTex','fontsize',
    fs1)



%% 3D figure


figure('color','w')
plot3(x(:,1), x(:,2), x(:,3), 'b','linewidth',1.5);
hold on, grid on
daspect([1 1 1])
xlabel('$x$␣[m]','interpreter','LaTex','fontsize',14)
ylabel('$y$␣[m]','interpreter','LaTex','fontsize',14)
zlabel('$z$␣[m]','interpreter','LaTex','fontsize',14)
title('Orbit␣of␣NOAA-19␣Satellite','interpreter','LaTex','fontsize'
    ,16)
xlim([-1.5e7 1.5e7])
ylim([-1.5e7 1.5e7])

% Plot Earth
earth_radius = 6.371e6;
[x_earth, y_earth, z_earth] = sphere(100);
earth_surface = surf(x_earth * earth_radius, y_earth * earth_radius
    , z_earth * earth_radius);
set(earth_surface,'FaceColor',[0.2 0.2 1],'EdgeColor','none','
    FaceAlpha',0.5);

% Plot Satellite
plot3(x(1,1), x(1,2), x(1,3), '.g','markersize',25,'markerfacecolor
    ','g') % Start point
plot3(x(end,1), x(end,2), x(end,3), '.b','markersize',25,'
    markerfacecolor','b') % End point
```

```
% Adjust view angle for better visualization
view(30,30);

legend('Satellite␣Orbit','Start␣Point','End␣Point');
%% Orbit equation

% Parameters for the NOAA-19 satellite orbit
p=(rp*vp)^2/MU; % Semilatus rectum (m)
e=0.001; % Eccentricity

a=p/(1-e^2); % Semimajor axis (m)
P=2*pi*sqrt(a^3/MU); % Orbital period (s)

th=linspace(0,2*pi,1000);
r=p./(1+e*cos(th));

x1=r.*cos(th);
y1=r.*sin(th);

plot3(x1,y1,zeros(size(x1)), 'g--','linewidth',lw1)


return
```

[19]

This Matlab code simulates the orbital movement of a real existing artificial satellite. In the first part of the code you will find the various parameters used, alongside their identification. The call to the `ode` and `fr2b` functions will be looked at specifically later.
The next two parts represent the writing to print the two images on video, one in 2D and the second more outlined in 3D.
The fourth sector allows me to calculate the orbit equation. Orbital parameters such as semilatus rectum, eccentricity, semimajor axis and orbital period are calculated. The orbit is plotted as a green curve.

In addition to the Matlab code, external functions must be considered, the `fr2b` which represents the system of ordinary differential equations (ODE) for the two-body problem in an inertial reference frame.

```
    function xd=fr2b(t,x,MU)

xd=zeros(6,1);
r3=norm(x(1:3))^3;

```

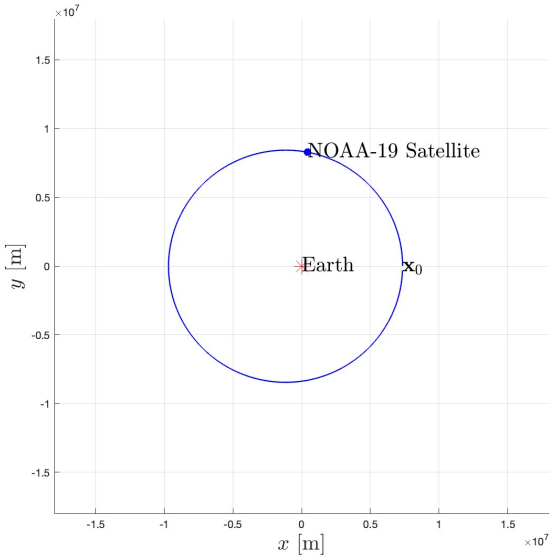

Figure 2.5: Code figure 2D

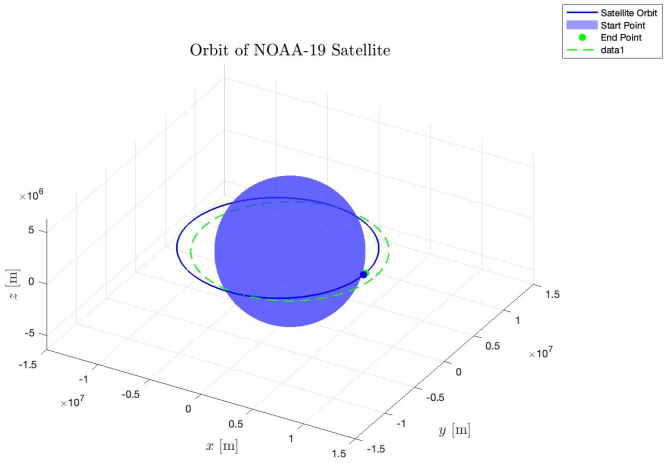

Figure 2.6: Code figure 3D

```
xd(1:3)=x(4:6);
xd(4:6)=-MU*x(1:3)/r3;
```

and the `oe2rv`, which converts orbital elements into state vectors relative to

a geogentric reference system.

```
    function [r_ge,v_ge,p,T]=oe2rv(y,th,mup)

a=y(1);
e=y(2);
i=y(3);
Omega=y(4);
omega=y(5);

p=a*(1-e^2);
rs=p./(1+e*cos(th));
r=[rs.*cos(th)
        rs.*sin(th)
        zeros(size(th))];

T=rot_mat([3 1 3],[Omega i omega]);
r_ge=T*r;

if nargin<3
    v_ge=[];
else
    smp=sqrt(mup/p);
    v=[-smp*sin(th);smp*(e+cos(th));zeros(size(th))];
    v_ge=T*v;
end
```

We will leave out the MATLAB calculation of escape velocity.

To conclude the chapter on orbital systems we need to discuss the reference systems used in orbital analysis. The main ones are three:

1. **LVLH** (Local Vertical, Local Horizontal).

$$R_l = P_1, l_1, l_2, l_3 \tag{2.8}$$

   It is a non-inertial spatial coordinate system anchored to the orbiting object. Generally the origin is placed in conjunction with the center of the object and the system will move following its orbital trajectory. The three values correspond to: l_1, Local Radial Outward, aligned with the radial direction that exits the surface of the body considered; l_2, Local Orbital Velocity, opposed to motion; l_3, Local Cross-Track, orthogonal to the other two.

2. **LORF** (Local Orbital Frame).

$$R_l = P_1, o_1, o_2, o_3 \tag{2.9}$$

Also in this case we are considering a non-inertial coordinate system anchored to the warp object. The axes are exchanged, the main direction of the orbital motion is different: in fact if l_2 is in the direction opposite to the motion, in this case o_1 is aligned with the direction of the speed. The substantial difference with the reference system described before is that the first focuses on the direction while this system focuses on the orbital speed.

3. **PF** (Perifocal)

$$R_l = P_0, p_1, p_2, p_3 \tag{2.10}$$

This last coordinate system differs from the others because it is inertial and is anchored to the main body. Consequently the axes also change: p_1 (Perifocal Axis), aligned in the perigee direction. p_2(In-Plane Axis) orthogonal to p_1 p_3(Out-of-Plane Axis),orthogonal to the other two axes. In this specific case, rotation and translation operations are taken into greater consideration.

[24]

2.2 Orbital control

The key element of automation is the repetitiveness of a manipulator's movements. What we are going to see in this chapter is how a robot can move autonomously, with precision and safety. The *control of a manipulator* refers to the management and regulation of movements which, considering everything as a set of joints and actuators, are nothing more than the way of carrying out specific tasks.

Let's look specifically at the schematization of a control system.

A control system is a dynamical system which nehaves in a certain prescribed way, in absence of human action. Let's see in a very schematic way what is present in this diagram:

- The **plant** is the main part, it is what needs to be checked. It specifically represents the manipulator, the physical system that must be regulated. From it comes $y(t)$ which is the output that must be controlled. Taking a trivial example like a cruise control, the output is the speed of the car.

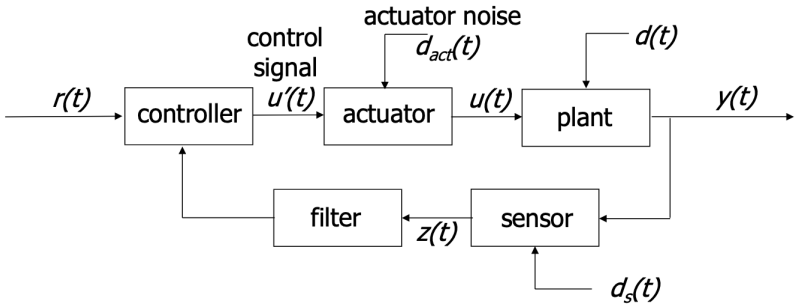

Figure 2.7: Control scheme

- The *controller* is the component of the control system that generates signals in response to the information received. Enter $r(t)$, i.e. the reference signal that is received before doing any type of computation.
- The *actuator* is used to convert the generated signals. The controller output cannot always enter the plant. For example, we could have an electrical signal that needs to be transformed into a mechanical one, or from digital to analogue, etc.
- The organ that has the task of capturing the output information is the *sensor*. In this way it is possible to recover the results obtained from the physical system and re-evaluate them to see if the control is done in the best way.
- The *filter* has a very similar function to the actuator: it modifies the signal. In this case, however, we are not talking about conversions but about reducing unwanted noise.

Everything indicated with d in the diagram represents the disturbance entering the corresponding block. [**26**] [7]
Another way to differentiate controllers is to see whether they exhibit linear or nonlinear behavior. In the case of linear controllers, the behavior of the controller can be described by linear differential equations with a mathematical structure that presents the connection between linear input and output. This concept greatly simplifies the analysis and design of control systems.
In the case of nonlinear controllers this is not possible, since inside there are equations with nonlinear terms to describe the behavior of the system. This leads to a marked increase in the level of complexity of the system and analysis. Generally the dynamic behaviors of this system are not easily understood and the methodology for dealing with these control systems changes

depending on the topic in question.[5] [29]
Also in this case a best example of motion control is the control of orbital trajectory. Generally the best way to control is considering three important task: guidance, for planning an optimal trajectory to track; navigation, for measuring and estimating the spacecraft state and control it, that means bringing the spacecraft state close to the trajectory planning. The examples are multiple, launch of a satellite or a spacecraft, an orbit keeping/orbit period change or a "randevouz" a manouver where a vehicle is required to approach another vehicle. The idea is always the same for all examples: chose an orbital dynamical model with all the formulas and then applied a controller for the problem. In this case we are going to see Nonlinear Model Predictive Control (NMPC) that we treat in the following examples

2.2.1 Trajectory Correction Maneuver

As we have already said previously, a satellite can be considered a robot only if it has peculiarities that allow it to correct itself automatically. The peculiarity that is gaining the upper hand in recent years is the ability to autonomously correct its orbit. A clear example is Ingenuity, the drone helicopter launched on 30 July 2020 inside the Perseverance rover which landed on Mars on 18 February 2021. The peculiarity of this drone was that it autonomously faced the **TCM (Trajectory Correction Maneuver)**, i.e. the correction of the trajectory so that it arrives at the desired point. [3] [28]

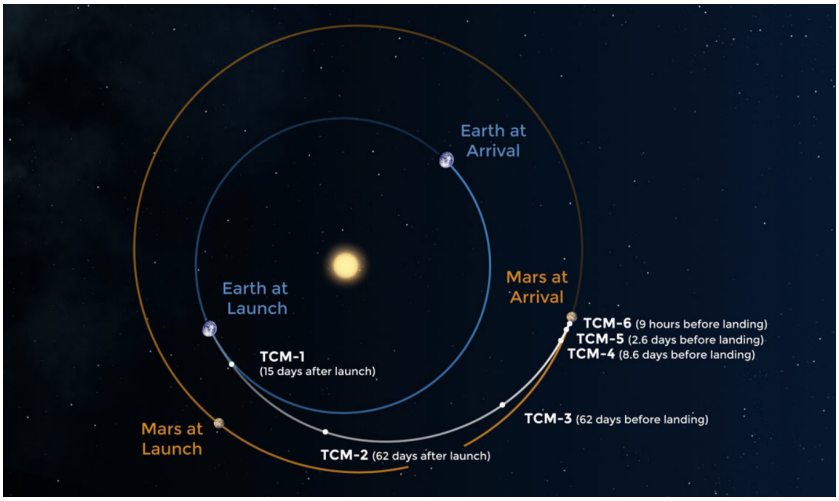

Figure 2.8: TCM illustration of Ingenuity

The trajectory correction maneuver is crucial for In-Service Assembly and

Manufacturing (ISAM) missions for several pivotal reasons.

Firstly, trajectory correction maneuvers are essential to ensure that spacecraft involved in ISAM missions maintain precise and safe trajectories during approach and rendezvous. This is particularly critical when maneuvering near other space structures or vehicles, where precision is paramount to avoid collisions and ensure a safe encounter.

Moreover, trajectory correction maneuvers are crucial to optimize the path of the spacecraft and ensure they reach desired points with precision. This is especially important in ISAM missions, where in-orbit assembly requires precise positioning of vehicles and their components.

Furthermore, the trajectory correction maneuver can be used to adjust the speed and direction of the spacecraft, enabling a controlled approach during rendezvous and facilitating assembly operations. This is crucial when handling large components and complex structures in space.

Finally, trajectory correction maneuvers contribute to the overall safety of the mission by ensuring that spacecraft are always in an optimal position relative to mission objectives, minimizing the risk of collisions or other incidents.

Basically, the manipulator taken into consideration, which can be a satellite but is often a rocket or a rover that must move from a planet A to a planet B, is given a terminal position and in addition the ability to verify the position in which is found. In this way, by calculating the error, it is possible to estimate the movement to be made to return to the correct trajectory. Below is a very simple MATLAB code that simulates this process. [17]

```
    clear all
close all
clc

% Initial parameters (approximate values for Earth to Mars mission)
current_velocity = 25000; % Current spacecraft velocity in m/s (
    approximate)
current_angle = 10; % Current trajectory angle in degrees (
    approximate)

% Calculate the necessary correction
target_velocity = 25500; % Desired velocity after correction (
    approximate)
target_angle = 15; % Desired angle after correction (approximate)

delta_velocity = target_velocity - current_velocity;
delta_angle = target_angle - current_angle;
```

```
% Display results before TCM
fprintf('Current velocity: %.2f m/s\n', current_velocity);
fprintf('Current angle: %.2f degrees\n', current_angle);

% Display the necessary correction
fprintf('Delta velocity for TCM: %.2f m/s\n', delta_velocity);
fprintf('Delta angle for TCM: %.2f degrees\n', delta_angle);

% Calculate the new velocity and angle after TCM
new_velocity = current_velocity + delta_velocity;
new_angle = current_angle + delta_angle;

% Display results after TCM
fprintf('New velocity after TCM: %.2f m/s\n', new_velocity);
fprintf('New angle after TCM: %.2f degrees\n', new_angle);

% Plot trajectory before TCM
figure;
subplot(1, 2, 1);
polarplot(deg2rad(current_angle), current_velocity, 'ro', '
    MarkerSize', 10);
title('Trajectory Before TCM');

% Plot trajectory after TCM
subplot(1, 2, 2);
polarplot(deg2rad(new_angle), new_velocity, 'go', 'MarkerSize', 10)
    ;
title('Trajectory After TCM');

%% Trajectory
% Approximate data for Earth-Mars trajectory
average_distance_earth_mars = 225e9; % Average distance (meters)
travel_duration = 9 * 24 * 60 * 60; % Travel duration (seconds)

% Calculate the average velocity needed for the journey
average_velocity_earth_mars = average_distance_earth_mars /
    travel_duration;

% Generate points on the theoretical trajectory
theta_theoretical = linspace(0, 2*pi, 1000); % Angles to create the
     trajectory
r_theoretical = ones(1, 1000) * average_velocity_earth_mars; %
```

```
    Constant velocity for the trajectory

% Plot the theoretical trajectory
figure;
polarplot(theta_theoretical, r_theoretical, 'b', 'LineWidth', 2);
title('Theoretical Earth-Mars Trajectory');

%% 3D
% Approximate data for Earth-Mars trajectory
average_distance_earth_mars = 225e9; % Average distance between
    Earth and Mars (meters)
travel_duration = 9 * 24 * 60 * 60; % Travel duration (seconds)

% Calculate the average velocity needed for the journey
average_velocity_earth_mars = average_distance_earth_mars /
    travel_duration;

% Generate points on the theoretical trajectory
time_theoretical = linspace(0, travel_duration, 1000); % Time to
    create the trajectory
position_theoretical_x = time_theoretical *
    average_velocity_earth_mars; % Position along the x-axis

% Positions of Earth and Mars
earth_position = [0, 0, 0];
mars_position = [average_distance_earth_mars, 0, 0];

% Fixed position of the satellite on the trajectory
satellite_position = [position_theoretical_x(500), 0, 0]; % For
    example, position halfway through the path

% Plot the theoretical trajectory in 3D with Earth, Mars, and the
    satellite
figure;
scatter3(earth_position(1), earth_position(2), earth_position(3),
    500, 'g', 'filled', 'Marker', 'o', 'MarkerEdgeColor', 'k', '
    LineWidth', 1.5);
hold on;
scatter3(mars_position(1), mars_position(2), mars_position(3), 500,
     'r', 'filled', 'Marker', 'o', 'MarkerEdgeColor', 'k', '
    LineWidth', 1.5);
scatter3(satellite_position(1), satellite_position(2),
    satellite_position(3), 100, [0.7, 0.7, 0.7], 'filled', 'Marker',
```

```
    'o', 'MarkerEdgeColor', 'k', 'LineWidth', 1.5);
plot3(position_theoretical_x, zeros(size(position_theoretical_x)),
    zeros(size(position_theoretical_x)), 'b', 'LineWidth', 2);
xlabel('X (meters)');
ylabel('Y (meters)');
zlabel('Z (meters)');
title('3D Earth-Mars Trajectory with Satellite');

% Grid settings
grid on;
set(gca, 'Color', [0.8, 0.8, 1]); % Blue background

return
```

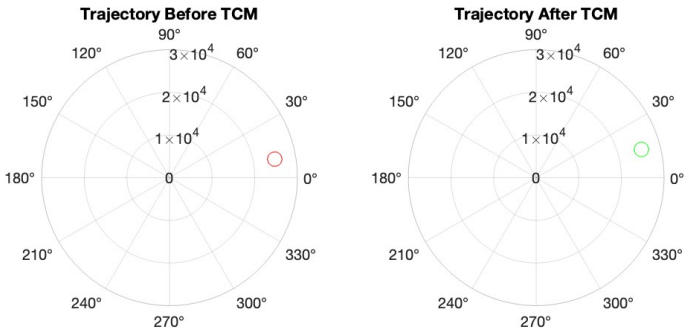

Figure 2.9: Correction simulation

In the first part of the code, parameters are defined for a non-real mission that sees a manipulator go from Earth to Mars. After verifying the position and speed it possesses, the robot calculates the correction necessary to re-establish the TCM. The program prints the parameters and results on the screen before and after making the corrections. The last two parts print two images featuring angle correction and a representation of the satellite moving from Earth to Mars.

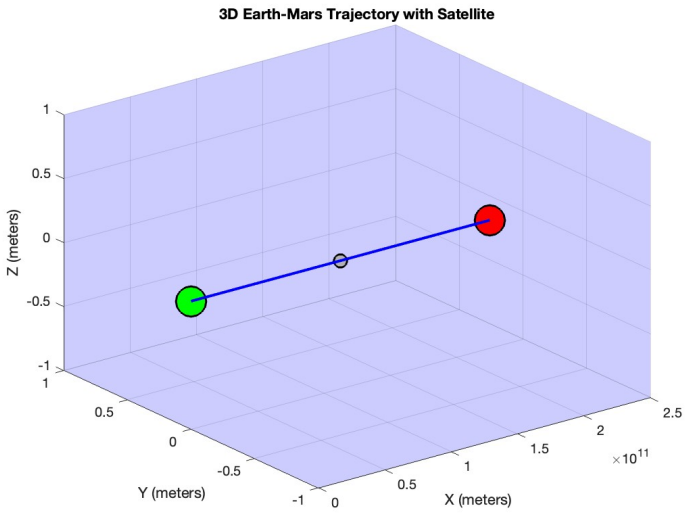

Figure 2.10: 3D image of the Earth, Mars and satellite

2.2.2 Change the shape orbit

Let's get to the heart of control theory, adding the work carried out on Simulink to our study. Let's imagine we need to create a controller finalized to change the shape of a spacecraft orbit from circular to ellipsoidal. We can estimate the following data:

- Spacecraft:
  - spacecraft body mass: 4000 kg;
  - total initial mass (body + fuel): 10000 kg;
  - engine exhaust velocity is ve = 4.4 km/s;
  - input saturation: ui=[-132, 132] kg · km/s2 , i = 1, 2, 3;
- Initial orbit:
  - emi-major axis: a0 = 6871 km;
  - eccentricity vector: e0 = (0, 0, 0);
  - inclination: ci0 = 1.
- Target orbit:
  - semi-major axis: ar = 8932 km;

– eccentricity vector: er = (0.25, 0, 0);

– inclination: cir = 1.

For simplicity we use a NMPC, designed using the `nmpc_design_st` See the main code:

```
clear all
close all
clc

%% Parameters
global MU RE ve
MU=4e5;
RE=6371;
ve=4.4;
m0=10e3;

%% NMPC Design

par.model=@model_orbit;
par.n=7;
par.Ts=30;
par.Tp=90;
par.R=0.1*eye(3);
par.P=diag([1,1e5*ones(1,4)]);
par.ub=132*ones(3,1);
par.lb=-132*ones(3,1);
par.tol=[1;0.005*ones(4,1)];
par.Tctrl=500;

K=nmpc_design_st(par);

% Reference
ar=(RE+500)*1.3;
er=[0.25;0;0];
cir=1;
yr=[ar;er;cir];

%return
%% Simulation

% Initial conditions
r0=(RE+500);
v0=sqrt(MU/r0);
```

```
x0=[r0 0 0 0 v0 0 m0]';
u0=[0;0;0];

% Simulation parameters
Tfin=30000;
st_size=2;

open('sim_orb_ctrl_1')
%return

%% simulation

sim('sim_orb_ctrl_1')

return
%% plot
close all
% clc

i1=580;
i2=1200;
Tctrl=K.Tctrl;
orbit_plot
```

The first part in this code is the assignation of the parameters. Let's remember that MU is the gravitational parameter and change with the coordinate system that we use and RE is the earth radius.

The second block is is the set up for the parameters of the NMPC. The model orbit function is implemented by this lines of code:

```
    function [f,y]=model_orbit(t,x,u)

global MU

r3=sqrt(sum(x(1:3,:).^2,1)).^3;

f(1:3,:)=x(4:6,:);
f(4:6,:)=-MU*x(1:3,:)./r3+u./x(7,:);
f(7,:)=0;

y=rv2oe(x);
```

After that we find reference and simulation setup.

The most important part is the simulink control. The block used are:

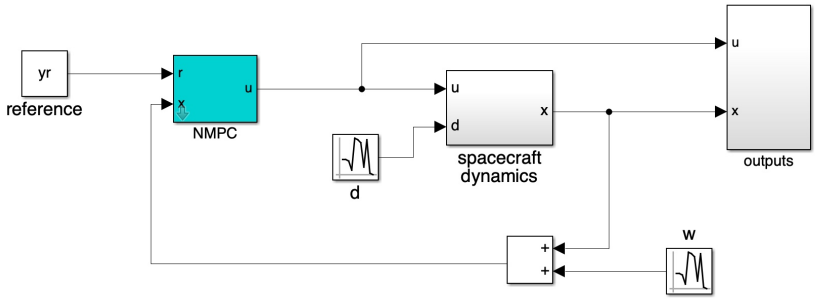

Figure 2.11: Simulation Orbital Control

- Constant reference where is implemented the desired condition.
- NPMC block, in which is implemented the K that we had already compute.
- Spacecraft dynamics block, in this we consider as input u (the output of the NPMC) and d that is the disturbance (in this case is setted zero). Inside the block of "spacecraft dynamics" we have a subsystem represented in 2.12.

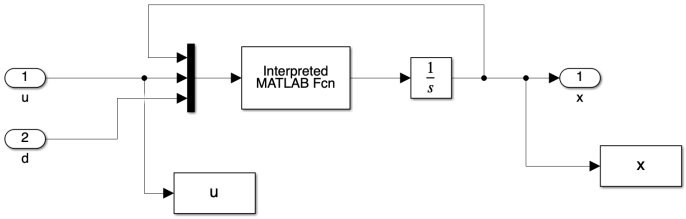

Figure 2.12: Spacecraft dynamics subsystem

In this subsystem is written the Matlab function that follow.

```
function xd=spacecraft_dynamics(t,x,u,d)

% parameters
global MU RE ve
mb=4e3;

% variables
```

```
8  if size(x,2)>1 && size(u,2)==1
       u=u*ones(1,size(x,2));
10     d=d*ones(1,size(x,2));
   end
12 r=vecnorm(x(1:3,:),2,1);
   v=vecnorm(x(4:6,:),2,1);
14 m_dot=-vecnorm(u,2,1)/ve;

16 % end of propellant
   ii=find(x(7,:)<=mb);
18 x(7,ii)=mb;
   u(:,ii)=0;
20 m_dot(ii)=0;

22 % drag force
   cd=1;
24 S=12; % m^2
   rho0=1.22; % kg/m^3
26 H=8; % km
   rho=rho0*exp(-(r-RE)/H);
28 cf=1e3; % conversion factor m -> km
   Fdrag=-1/2*cd*S*rho.*v.*x(4:6,:)*cf;
30
   % state equations
32 xd(1:3,:)=x(4:6,:);
   xd(4:6,:)=-MU*x(1:3,:)./r.^3+(Fdrag+d+u)./x(7,:);
34 xd(7,:)=m_dot;
```

This function plays a fundamental role in modeling and describing the dynamics of the spacecraft under the influence of gravitational forces and control inputs. A crucial aspect that this representation carefully considers is the variation of mass, which dynamically decreases due to propellant consumption. This aspect is critical for accurate modeling, as propellant consumption directly impacts the spacecraft's mass and, consequently, the forces acting on it.

A key element in space dynamics is the drag force, representing the resistance the spacecraft encounters while traversing the Earth's atmosphere at high speeds. The formulation of this force takes into account various factors, including atmospheric density, spacecraft velocity, and drag coefficient. This step is crucial for understanding the atmosphere's effect on the spacecraft's trajectory, especially during reentry.

Finally, the state equations defined in this context represent the deriva-

tives of position (velocity), velocity (gravitational forces and control contributions), and mass (propellant flow rate). This comprehensive modeling allows for a detailed and accurate capture of the spacecraft's dynamic behavior under various flight conditions, providing a solid foundation for spacecraft analysis and control.

- Output Block for simulate the behavior of controller.

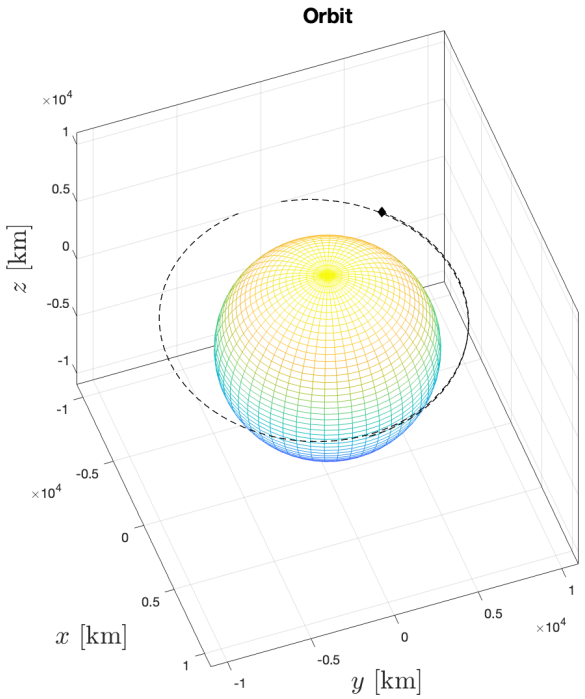

Figure 2.13: Orbit plot

The last part of the code is for the plot of the orbit in fig. 2.13.

2.2.3 Randevouz

The last example is the space randevouz. In space exploration, space rendezvous refers to the process where two spacecraft are brought together and meet in orbit around Earth or another celestial body. This maneuver is critical for various space missions, including spacecraft docking with space stations such as the International Space Station (ISS), satellite rendezvous

for orbital positioning or inter-satellite connections, and docking maneuvers in space cargo missions.

Space rendezvous involves complex orbital calculations and control algorithms to determine the optimal trajectory and timing for approach maneuvers. It requires precise planning to ensure a safe and controlled approach, avoiding collisions and hazardous situations in orbit. The process relies on sophisticated orbital calculations and control algorithms to determine the optimal trajectory and timing for approach maneuvers. Rendezvous is crucial for In-Service Assembly and Manufacturing (ISAM) missions for several reasons.

Firstly, it facilitates in-orbit assembly by allowing various components or spacecraft to approach and connect safely, enabling the construction of complex structures in space. This is particularly relevant for missions involving the building or assembly of large-scale structures, such as space stations.

Moreover, rendezvous is essential for transferring components or modules between spacecraft. Whether transporting vital parts for space station assembly or exchanging materials, controlled approach and connection are necessary for the success of ISAM missions.

Additionally, ISAM missions may involve refueling or maintenance operations. Rendezvous enables spacecraft to approach each other for these tasks, ensuring the efficient replenishment of resources and the necessary upkeep of equipment.

The flexibility provided by rendezvous is critical for ISAM missions. It allows spacecraft to adapt to evolving mission requirements, addressing unforeseen challenges or changes in plans during the mission.

Furthermore, conducting rendezvous in orbit enhances resource efficiency. Rather than launching fully assembled large structures from Earth, ISAM missions can send smaller components, optimizing resource utilization and reducing the costs associated with space launches.

Let's see a Matlab code and Simulation based on the same functions and Simulink block of the previous example.

```
clear all
close all
clc

%% HCW equations

MU = 0.3986e15;
RE = 6.38e6;
a = RE + 400e3;
w = sqrt(MU / a^3);
```

```
11
A = [0 0 0 1 0 0
13     0 0 0 0 1 0
       0 0 0 0 0 1
15     3*w^2 0 0 0 2*w 0
       0 0 0 -2*w 0 0
17     0 0 -w^2 0 0 0];
eig(A)
19
B = [zeros(3); eye(3)];
21
H = ss(A, B, eye(6), zeros(6,3));
23
%% NMPC design
25
par.model = @pred_model;
27 par.nlc = @nlcon;
par.n = 6;
29 par.Ts = 3;
par.Tp = 100;
31 par.Q = 0.1 * eye(6);
par.P = 10 * eye(6);
33 par.R = 1 * eye(3);
par.lb = -1 * ones(3,1);
35 par.ub = 1 * ones(3,1);

37 K = nmpc_design_st(par);

39 %% Simulation initialization

41 % Data for a real rendezvous
r_target = [7000e3; 0; 0]; % Desired final position
43 v_target = [0; sqrt(MU/7000e3); 0]; % Desired final velocity
x0 = [-500; 0; 0; 0; 0; 0]; % Initial conditions
45
open('sim_hcw_nmpc.slx')
```

We will not delve much into the code portion since it utilizes the same previous controller. Regarding the equations used in the controller, they are the Hill-Clohessy-Wiltshire equations, derived under the assumption that the two bodies are in circular orbits and close to each other, allowing the relative motion to be approximated by a relative coordinate system. This relative coordinate system simplifies the analysis of relative motion.

Here the Simulink system

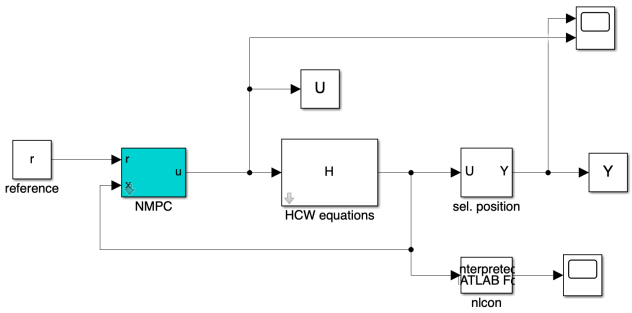

Figure 2.14: Simulink randezvous

Chapter 3

Integrated Robotic Control: Trajectory Planning and Visual Control

After a thorough exploration of satellite control and orbital trajectory planning in the previous chapters, it is now time to delve into the detailed study of specific robot control, with a particular focus on the application of these advanced strategies to the previously discussed Popup Robots. In this section of our study, we will delve into the dynamics involved in integrating robotic control, examining in detail considerations related to trajectory planning and visual control. These aspects play a crucial role in optimizing operational performance, contributing to the success of complex advanced space operations, especially in the context of Popup Robots.

3.1 Trajectory planning

Another fundamental aspect in the discussion of automation is linked to the trajectory to follow. We underline the fact that a trajectory is important to consider even when dealing with specific machinery that moves without the automaticity of the movement. When we consider robots that operate at very large distances, or in any case that are not controllable and have automatic learning of their tasks and movements, programming their trajectories is not only important but also fundamental. In this way, a process is created that defines and optimizes the paths that the robot must take to achieve specific objectives.
Clearly the field of movement of a manipulator is vast, the study behind the revolution movement of a satellite will not be the same as the movement of

an exploratory rover. In any case, however, the trajectories must be studied and planned in order to make the movement as simple as possible. First of all for everything related to operational efficiency, also saving time to complete the tasks assigned to the robot. There are also other reasons related to planning, for example being aware of the route that a manipulator must take brings a certain safety in the operations carried out since it is possible to avoid obstacles on the route. In any case, especially in space operations and scientific missions, this aspect cannot be ignored since it is essential to have the precision of movements as high as possible.

3.1.1 Joint Space vs Operational Space

Let's briefly mention the theory by considering non-mobile robots, differentiating two different cases. The first to be treated is **the joint space**, which represents the set of possible configurations in the robot's joints where each point corresponds to a specific position and orientation of the joints themselves. The requirement is to consider an initial position and a final position considering the laws that describe the motion. For this reason, algorithms are created to create trajectories. In the space of the joints it is necessary to consider the initialization of the positions and orientations of the terminal member. In this category we differentiate two methods:

- *Point-to-point motion*, in which the manipulator must move from an initial configuration of the variables to a final configuration in a specific time interval. This basic form has the particularity of not considering intermediate points. This allows us to have a simple trajectory model that allows us to directly involve the initial and final position. The speed will also be maximum given that the system has no intermediate slowdowns and the control will, consequently, be much more precise.

- *Motion through a sequence of points* which is the exact opposite: it interpolates the trajectories and considers the series of intermediate points that the robot must pass through. It is a much broader and broader method and is better to use especially when the robot moves in trajectories going from one point to another in a fluid manner.

The second model considers **the operational space** for the study, which differs from the first as it no longer considers the various configurations of the robot but defines the trajectories directly in the space in which it operates. Basically, the former studies and controls the joints to obtain a certain movement, while the latter focuses directly on defining the desired positions.[30]

3.1.2 Obstacle-Cluttered Environments

Let's briefly discuss the issues related to mobile robots. Fundamentally, the discussion made above is valid, what must be considered is the issue of obstacle avoidance which makes it crucial to generate a movement that allows the robot to avoid obstacles in order not to risk collisions during the execution of an assigned task. Motion planning techniques are different:

- *RRT - Rapidly-Exploring Random Trees.* This technique creates an exploratory tree, the algorithm focuses on randomly generating new nodes. After that, new random nodes are sampled in space. In this way there will be efficient coverage of obstacles even in areas not yet seen.
- *Roadmap* which exploits the network of feasible paths to create the free space connectivity of the configurations. Create a structured representation of the journey and simplify the search for solutions.
- *Cell decomposition*, divides the space into cells and searches for safe channels for planning. It is very valid within large complexes because it separates the environment and resolves more easily.
- *Probabilistic* adds the random sampling of the configuration space to the methods already seen.
- *Artificial potentials.*

[5] [12] [32] [4]

Let's see a simple example to understand.

```
% Definition of the environment map
x = linspace(-10, 10, 100); % x-coordinate
y = linspace(-10, 10, 100); % y-coordinate
[X, Y] = meshgrid(x, y); % Coordinate grid
obstacle = zeros(size(X)); % Initialization of the obstacle map
obstacle(X.^2 + Y.^2 < 4) = 1; % Definition of a circle as an
    obstacle

% Definition of the initial and final position of the rover
start = [-8, -8]; % Initial position
goal = [8, 8]; % Final position

% Controller parameters
K_att = 1; % Attraction constant
K_rep = 100; % Repulsion constant
```

```
% Simulation of the rover's motion
dt = 0.1; % Time step
max_iter = 1000; % Maximum number of iterations
pos = start; % Initial position of the rover
trajectory = pos; % Trajectory trace of the rover

for i = 1:max_iter
    % Computation of the force field
    F_att = K_att * (goal - pos); % Attraction force towards the
        goal
    F_rep = K_rep * sum((pos - [X(:), Y(:)]) ./ ((pos(1) - X(:)).^2
        + (pos(2) - Y(:)).^2).^3); % Repulsion force from obstacles
    F_total = F_att + F_rep; % Total force

    % Update of the rover's position using Newton's law
    pos = pos + dt * F_total / norm(F_total);
    trajectory = [trajectory; pos]; % Addition of the position to
        the trajectory

    % Check if the rover has reached the goal
    if norm(pos - goal) < 0.1
        disp('The␣rover␣has␣reached␣the␣goal!');
        break;
    end
end

% Visualization of the rover's trajectory and obstacles
figure;
contourf(X, Y, obstacle, 'LineStyle', 'none'); % Visualization of
    obstacles
hold on;
plot(start(1), start(2), 'go', 'MarkerSize', 10, 'MarkerFaceColor',
     'g'); % Initial position of the rover
plot(goal(1), goal(2), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', '
    r'); % Final position of the rover
plot(trajectory(:, 1), trajectory(:, 2), 'b', 'LineWidth', 2); %
    Plotting the rover's trajectory
scatter(trajectory(:, 1), trajectory(:, 2), 50, 'k', 'filled'); %
    Addition of trajectory points
xlabel('X');
ylabel('Y');
title('Rover␣Trajectory␣with␣Obstacles');
```

```
50 axis equal;
   grid on;
```

This short and simple code uses the force model derived from the controller parameters to see a rover moving iteratively over time and simulates a map in which there are obstacles and the initial and final position of the rover.

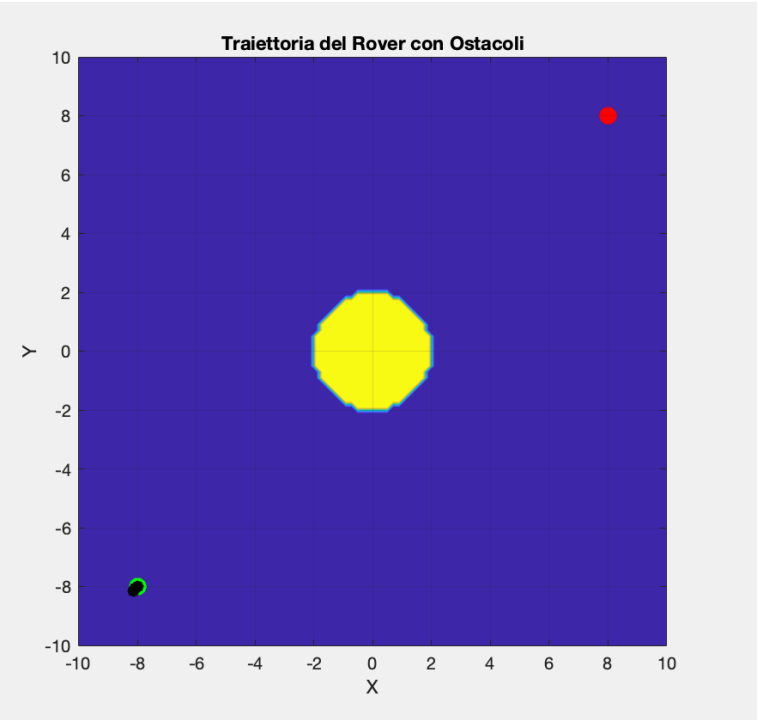

Figure 3.1: Rover position

3.1.3 Obstacle Detection and Avoidance

Clearly here we have very briefly discussed the theory linked to the control of trajectories. A piece that is missing is linked to the fact that it is not enough to randomly simulate the movement of a manipulator because if you rely on chance, collision is inevitable. For this reason, sensors and cameras are necessary to be mounted and positioned above the robot, in order to detect the obstacle with absolute certainty and clarity. This application is also used in everyday objects, a very common example being parking sensors placed on cars.

Sensors dedicated to obstacle detection play a crucial role in robotics and automation, playing fundamental importance in the perception and safe navigation of autonomous devices in various contexts. There are various solutions that lead to satisfying all even the most specific needs to ensure that the systems work correctly. Generally the simplest and most easily available are ultrasonic sensors. They are characterized by emitting a chime and sound waves and measuring the time it takes for them to return. This provides detailed information about distance and therefore location. Infrared, alternatively, uses infrared rays to identify the presence of nearby objects, while laser sensors use beams of laser light to obtain precise measurements and allow accurate detection of obstacles. The aim is however always the same, that is to calculate the distance and transform it into useful space before the collision.

Cameras are also able to give the same help. They can provide visibility by adding sensory perception. It is certainly a more sophisticated method that is also able to distinguish shapes and interpret specific details of the object in question. The problem is that the calculations and software also become more complex.

As an example, let's consider an autonomous vehicle that uses a comprehensive sensor system to navigate an urban environment. Ultrasonic sensors are engaged in detecting nearby obstacles, while laser sensors provide precise information about the distance and shape of surrounding objects. Cameras integrate this information, identifying traffic signs, pedestrians and vehicles. The Electronic Control Unit (ECU), the beating heart of the system, continuously processes data from all the sensors, making instant decisions to drive the vehicle safely. The integration of all this information is entrusted to the ECU, the electronic control unit. This component processes the data coming from the different sensors in real time, creating a detailed map of the surrounding environment. In obstacle detection situations, the ECU triggers warnings, implements avoidance strategies or initiates corrective actions, ensuring safe and smooth interaction with the environment.

Incessant technological progress has led to a new era of advanced sensors, enhanced by computer vision and artificial intelligence technologies. These innovations enable a deeper understanding of the environment, significantly improving the perception and decision-making capabilities of autonomous devices. In this way, obstacle detection sensors play an increasingly central role in shaping the future of automation and robotics. [6]

3.2 Visual Control

Making a robot as autonomous as possible also means providing a method for verifying the position it is in to allow it to move autonomously. Basically, a visual system is provided that allows you to acquire information on the geometry of the environment in which the robot is located. This is because, in this way, it is possible to plan both the motion and the control and, by programming it through the software, make this procedure autonomous. The highly versatile visual control strategy can be successfully implemented in Popup Robots as well, providing the unique capability to effectively compensate for link deformations. This specific application enables precise addressing of dynamic variations in flexible structures, thereby enhancing the accuracy and reliability of space operations, especially during complex servicing, assembly, and production activities.
The mechanism to control the movement of a robot must be based on the image, a bit like the human body. The idea is that, by placing some "eyes" on the end-effector, like a camera, it is possible to acquire the image and, after, compute the various distances of the object in front of the manipulator and, by simulating a man's celebrating function, go and calculate the distance of the object from the manipulator.
This method which is based on the image is called *Visual Servoing* and exploits the type of feedback control, which will be explained later. Visual

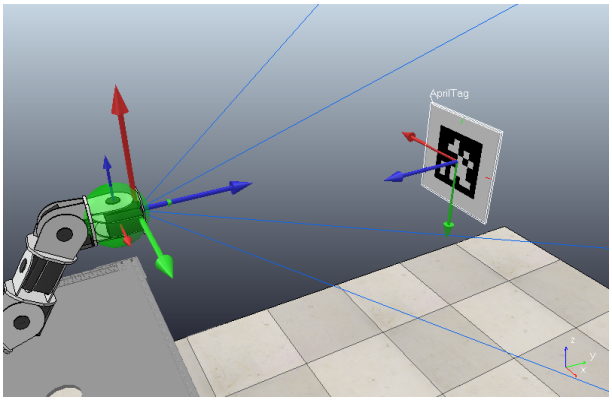

Figure 3.2: Schematic Visual Servoing

control, in the context of inflatable robots, can be implemented to compensate for deformations in their links in various ways. The use of vision systems allows the robot to perceive the surrounding environment, detecting any deformations or changes in shape. These visual cues are dynamically utilized to adjust the robot's control, enabling real-time responses to variations in its

structure. Through the use of visual feedback, obtained through cameras or visual sensors, it is possible to monitor the current shape of the robot and adjust the commands sent to motors or actuators to compensate for deformations. Control algorithms can be designed adaptively, leveraging visual information to recognize and respond to link deformations. Additionally, the implementation of predictive models based on past visual data allows the system to anticipate and mitigate future deformations. Overall, integrating visual control into inflatable robots enhances the system's ability to adapt in real-time, enabling them to maintain the desired shape and improve overall performance.
The general idea of this method, after acquiring the image, is to synthesize the control in the image plane. First of all, an analysis must be made that allows calculating the error between the appearance of the image itself and what the image really is. Only then go on to create the relative motion that relates the camera to the object. Basically what the software sets out to do is plan in the image plane by programming a gradual evolution of the desired feature towards the final destination. It goes without saying that various features such as points, lines and curves are needed for object tracking. By translating this part into homogeneous coordinates, a map can be created between the various points in space that belong to the rigid object. In this way, knowing the movement of the object a priori, such as that of a satellite, it is possible to predict what its position will be a few moments before reaching it. [14] As has been highlighted, the substantial phases for

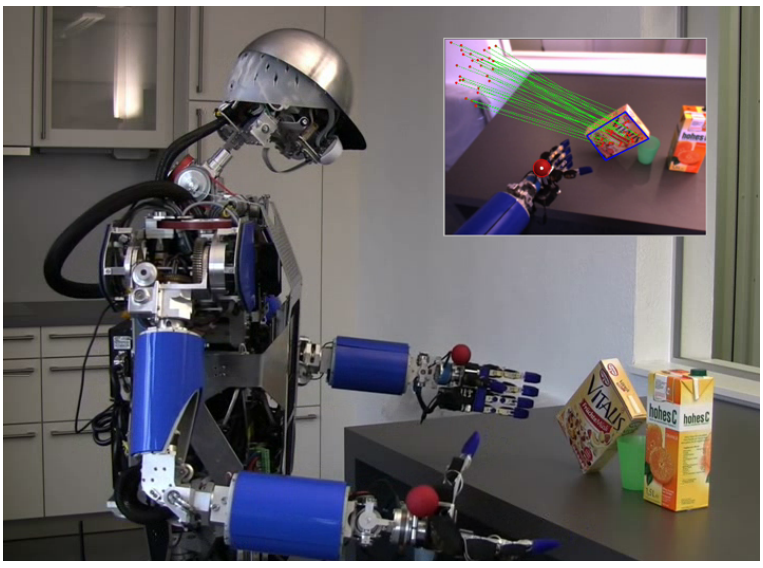

Figure 3.3: Visually guided trajectory execution

image vision systems can be distinguished in a first part of classification of the object, i.e. of recognition of it and then, subsequently, of its localization. This is where machine learning comes into play, since unlike a human being, a machine is able to recognize an object, but if it differs a little in the criteria from the image it had as a basis, it becomes a problem. For this reason, increasingly precise algorithms came into play, eventually using *deep learning*, which repeated the process several times by scanning the image more precisely. [23]

Whether we consider a manipulator that must grasp an object, or a robot that must travel a certain path, in both cases it is necessary to plan a trajectory that takes the manipulator to a certain position. In the case of the object to be grasped, the step of calculating the trajectory must be added to grasp that object considering its shape. The algorithm in question is called *look-and-move* and uses a control algorithm to evaluate the space and make the manipulator move.

To make the manipulator work properly, there will be more than one step in programming the manipulator. First of these is the **configuration of the visual system** which tries to associate, as much as possible, the human eye with a telemera, taking into account all the problems related to depth calculation. Normally to correctly simulate this point you use more than one camera, so that you can have a *3D vision* that perfectly simulates the distance between the object and the manipulator. It is not possible to get this view with just one camera, unless there is more than one image of the object you are considering. The choice of the number of cameras to use lies in considering a matrix that relates the economic side and the accuracy of the manipulator.

Clearly it is not enough to choose the number of cameras to use but it is also necessary to consider the positioning of the cameras. For simplicity, only single-camera configurations will be discussed, which are divided into two groups.

- **eye-to-hand**, in this configuration the camera is positioned on a fixed base integral with the manipulator triad. This mode allows for more simplicity in the calculation of the algorithms since, by not moving, the camera's field of view does not undergo variations. The pro is that the accuracy is very high and the calculations are simpler. The downside is that when the manipulator moves there is the risk of it coming between the object and the camera.

- **eye-in-hand**, in this case the camera is fixed on the manipulator, therefore it moves integrally with the end effector of the manipulator.

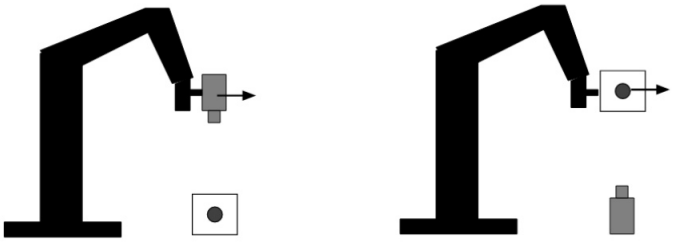

Figure 3.4: Eye-in-hand on left, eye-to-hand on right

In the following discussion we will consider the eye-in-hand configuration with a camera, but we underline that there can also be mixed configurations with multiple cameras which are called **hybrid**.

3.2.1 Processing

The next step is to process the visual information obtained. From a computational point of view, it is the most complex step: it is necessary to extrapolate numerical information from the image obtained that can be converted into actions that the robotic system must carry out. There are two steps to deal with: **segmentation** and **interpretation** of the image which we will see in the next two subsections.
The algorithm consists of taking the displayed image and treating it as a two-dimensional array that decodes the space that is displayed by the manipulator. This allows you to define the vector *image function* which has as components the physical quantities associated with the pixels of the image itself.

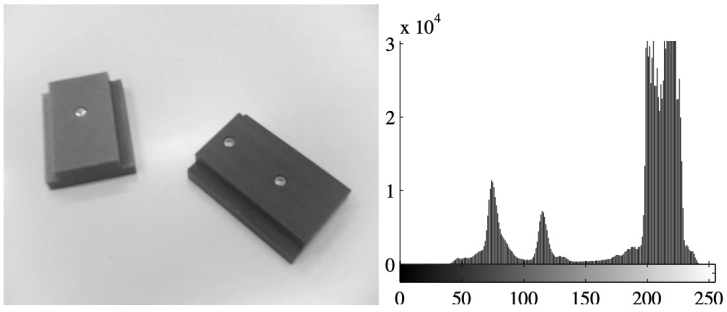

Figure 3.5: Grayscale histogram

Depending on the type of image, the number of components changes, in the color image there are three components $I_r(X_I, Y_I)$, $I_g(X_I, Y_I)$ and

$I_b(X_I, Y_I)$ which identify the shades of red, green and blue; while, in the case of a black and white image, the function has only one component $I(X_I, Y_I)$ which identifies the gray tones. Also in this case, for simplicity, only black and white images will be treated. Generally the device scale is made up of 256 gray levels and each of these levels corresponds to 1 byte of memory and, at each level, a value on the histogram corresponds. In this way you can create a graph in which a bar corresponds to the pixel of the image associated with its specific level which goes from 0 to 255.

3.2.2 Segmentation

At the very moment we begin to divide the images into parts to analyze them piece by piece, we are performing a *segmentation*. Normally, it is segmented based on the objects present within the image, aiming to obtain the clearest and most specific geometric indication of space. Generally, two specific approaches can be followed: the first involves identifying the *regions* within the image, while the second involves identifying the *edges* of the image. In any case, the two methods are complementary since edges can be obtained by isolating the contours of a region, and vice versa.

Region-based segmentation

The fundamental concept of this method is pixel uniformity. In fact, the regions connected by growth are obtained, that is, by means of small groups of pixels.
Generally, *binary segmentation* is used which consists in referring to a light intensity scale with only two values, 0 and 1. It is easy to understand this mechanism by imagining an object of low brightness on a background with very high brightness, where a level is attributed to low brightness and all the pixels that belong to the same gray level will be part of the S_o group (i.e. of the object), while all the other pixels will be part of the S_b group (i.e. of the background). Clearly if we modify everything, for example thinking of a bright object to grasp in a very dark room, we can reverse everything. The tricky part is choosing a brightness threshold that makes the pixel part of the "object" or "background", otherwise you risk creating confusion. Generally we take the above-mentioned histograms and, depending on whether we have light or dark objects, we choose the maximum or minimum of the histogram as the threshold value. This is difficult when there is a lot of noise in the histograms. But you can implement everything through the use of filters.

Figure 3.6: Region-based segmentation

Image-based segmentation

In this typology there must be discontinuities on the gray level in the image, in this way the edges are formed. Basically when a high variation is found at a precise moment between the analyzed pixel and the previous one, the pixel in question is called "edge". These pixels are then joined together allowing the formation of lines to consider real objects. On a practical level, understanding the concept becomes trivial, just check where the pixel is "significantly darker" than the neighboring one. On a mathematical level the situation is slightly more complex because gradients are used.
There are many techniques used but most consist of calculating the gradient of the aforementioned vector component ($I(X_I, Y_I)$). Since the boundary is the transition of two regions, by calculating the gradient, which by definition measures the rate of change, we can see where there is maximum modulus. At these points there will be contours. To calculate the gradient it will be sufficient to evaluate the directional derivatives along the two orthogonal directions.

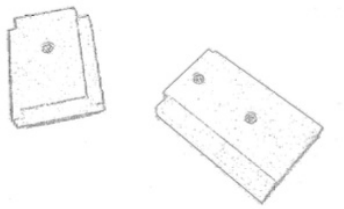

Figure 3.7: Image-based segmentation

3.3 Advanced Visual Control Strategies for Robotic Operations

In space, the use of visual servoing is crucial to enable precise and autonomous robotic operations. Two notable methodologies in this context are ArUco marker recognition and the utilization of YOLO (You Only Look Once).

ArUco marker recognition, based on two-dimensional patterns, proves particularly effective for the precise localization of objects in space environments. ArUco's ability to accurately determine the position and orientation of objects makes it an ideal ally for spacecraft navigation and orientation, contributing to ensuring safe and accurate space operations.

On the other hand, YOLO offers an advanced approach to real-time object detection. Its capability to quickly recognize and locate elements of interest in images and videos is highly valuable in space, enabling an immediate response to dynamic and unexpected situations. This swift identification is crucial for the safety and success of complex space operations.

The integration of ArUco and YOLO in space addresses the need to develop autonomous and intelligent robotic systems. ArUco provides detailed information about object positions, while YOLO contributes to real-time detection of changes in the space environment. This combination of technologies facilitates the realization of increasingly sophisticated space operations, supporting navigation, object monitoring, and the manipulation of tools and payloads in challenging space environments.

3.3.1 Marker ARUCO

One method to facilitate image capture from the camera is to use a matrix that allows for binary coding. This matrix is called *marker* and, in this specific case **marker ArUco**, that play a fundamental role as tools for object recognition and localization in three-dimensional space.
These markers are two-dimensional codes that serve as visual reference points for visual servoing systems. The word "ArUco" comes from "Augmented Reality" and "University of Cordoba," where they were first developed. The peculiarity of these tools is due to their ease of decoding due to the structure. Indeed, ArUco is nothing more than a library, which finds the correspondence between the points of an environment and projects them into a 2D image, our binary matrix. Generally the ArUco marker is composed of a very thick external black border, in order to make its identification easier, and of the identification matrix within this perimeter. The peculiarity of this type of

coding is that it also allows to identify the rotation angle of the object itself. The set of markers for a particular application is called a *dictionary*. The

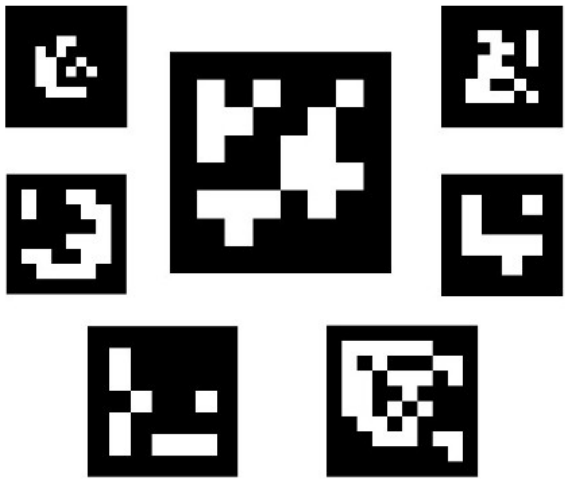

Figure 3.8: Example of marker

procedure that should be carried out starts from the creation of the marker which, subsequently, should be printed and positioned in the environment, in our case in the object to be docked. Parameters such as the "marker id", the size of the output marker image and the output image itself are essential to create the marker. Generally the marker can be created using code, but also from the website `https://chev.me/arucogen/` which prints an image of the marker. The second step is to allow recognition of the marker by the

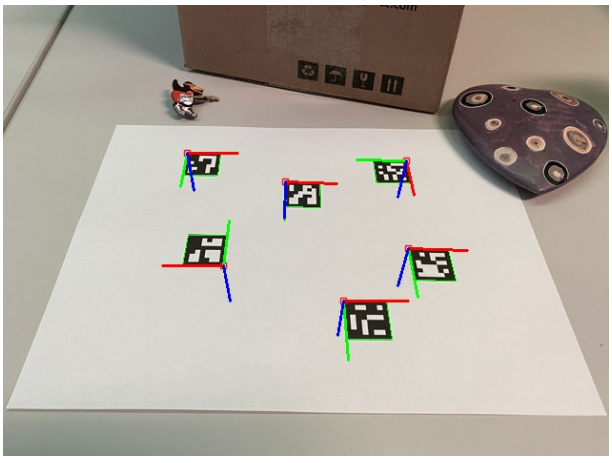

Figure 3.9: Marker with trajectory

video camera. First markers must be detected, roughly speaking it means that the camera must identify "black squares" in its image. Subsequently, the image inside the black frame must be decoded, also to understand if they are actually markers. The fundamental parts for this point are the different functions because depending on the one used I can identify the position or rotation of the point with respect to the camera.

Therefore, a summary of the steps in using markers consists of carrying out these functions in order:

1. *Detection*: first, markers are detected using the camera modes mentioned above. It is clear that in this case we will have grayscale image processing. In this way the markers are detected by the algorithm in the working environment of the manipulator.

2. *Localization*: the markers then give information related to position and angle.

3. *Tracking*: ArUco markers allow the tracking of objects in time and space.

This method is widely used in the aerospace sector thanks to the ease of detecting the markers. In a working environment with conditions that are extremely different from those of planet earth, it is essential to have a system that allows you to approach targets in a clear and distinguishable way. There are two operating modes in this case:

- *Acquisition*, where the ArUco marker is detected thanks to the image and subsequently identified by reading the embedded code even without it being known beforehand

- *Tracking*, which uses the location information found in the first phase to process the information.

From tests carried out in Low Earth Orbit using the Planet and Asteroid Natural scene Generation Utility tool for realistic synthetic image generation, it was calculated that the success rate in satellite detection through ArUco markers is 99.76% within 5m, of 97.04% in the 7m and 78.06% in the 10m. This is because, in any case, the markers stand out against the background around them. [25] [33] [8]
Visual markers, exemplified by ArUco markers, play a crucial role in both In-Service Assembly and Manufacturing (ISAM) and soft robotic applications. Their significance lies in facilitating precise localization during assembly processes, ensuring accurate alignment and component assembly. Acting as visual reference points, these markers guide robots or human operators through

assembly sequences, enhancing overall operational efficiency. Additionally, their contribution extends to real-time quality control in manufacturing, allowing for the visual verification of product specifications. In contexts with diverse product configurations, visual markers offer adaptability, aiding in the identification and management of variations. In soft robotics, such markers, like ArUco, prove instrumental in compensating for deformations in flexible links or structures, providing essential reference points for shape adaptation. Furthermore, within automation and robotics, visual markers serve navigation and control purposes, offering crucial information about the robot's position and orientation in its environment. Overall, the utilization of visual markers enhances precision, operational efficiency, and flexibility, fostering increased automation and adaptability in dynamic scenarios.

3.3.2 YOLO

In the same way as AruCo markers, **YOLO** (You Only Look Once) is a system that sets itself the goal of solving all those problems related to predicting the movement of an object, such as the need for real-time analysis. This advanced algorithm, based on the use of color cameras, allows an image to be divided into a grid and, subsequently, to identify the objects within each cell into which the image has been divided. For each being YOLo is able to predict the coordinates and objects present. This step is done more than once and, thanks to deep learning, the algorithm is able to keep only the most accurate predictions.
The development of YOLO has made it possible to have a much better forecast than the image seen by the camera. The creators have developed three successive versions of YOLO, going to greatly increase the system's performance, even if the basic concept does not change: it bases its entire operation on deep learning and makes it possible to split the image into smaller parts , going to obtain a grid of squares. Furthermore, the prediction can be done on different scales, so as the grid size increases, the cells that divide the image become smaller and more precise. YOLO can also interface with software such as ROS, essential for robotics applications

Due to the lack of lights or colors this model is not so easy to reproduce in environments such as space, but it is convenient if you want to detect specific shapes within an environment. [37]

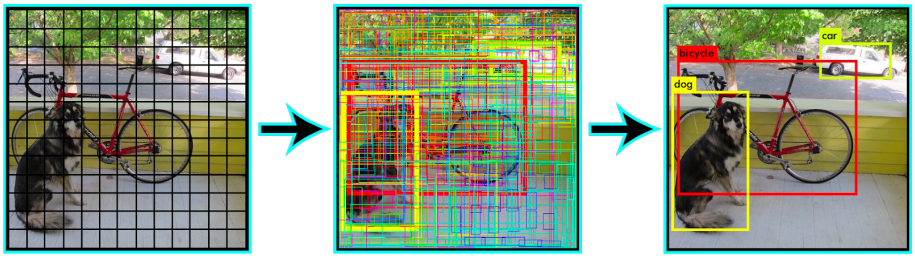

Figure 3.10: YOLO system

3.4 Scheme

All the calculations and simulations carried out previously are not precise because the treatment of all that part of robotics that allows us to identify our object to dock is missing. Basically, the previous calculations have been made in a simplified way to make everything more linear, but without a sensor or a camera it is almost impossible to identify an object to be grasped, mostly moving.
Through *Visual Servoing* it is possible to emulate the robot movement. In this specific case, it is necessary to fix the camera on the manipulator (on the end-effector to be precise), so that through the movement of the robot the target is kept in view of the camera (Eye in Hand system). Clearly the points that are highlighted by the view are then passed to the software which processes them and activates the joints so as to assume the configuration necessary to reach those points.
In fig. 3.14 it is possible to see a schematic idea of our visual servoing. The idea is to have visual feedback. Basically, images are acquired from the

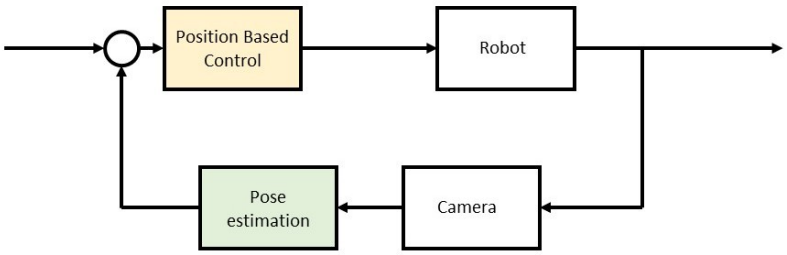

Figure 3.11: Model of Visual servoing

camera which, added to the initial position of the system, can calculate the relative position.

3.5 Perspectives of Visual Servoing: Atroscale and GITAI

In the landscape of cutting-edge space exploration, the convergence of Astroscale and GITAI signifies a captivating fusion of specialized knowledge and technological prowess. Astroscale, heralded as a trailblazer in the field of space debris removal, and GITAI, renowned for its expertise in humanoid robot development, come together in a shared quest for groundbreaking solutions. At the core of their collaborative endeavors lies the pivotal adoption of visual servoing, an advanced real-time visual guidance technology. This unique and sophisticated element emerges as the linchpin, fostering an environment conducive to the augmentation of perception, precision, and operational efficiency within the ambit of both organizations' ambitious initiatives.

Astroscale, standing at the forefront of addressing the escalating challenge posed by space debris, distinguishes itself through the engineering and deployment of advanced spacecraft meticulously designed for the capture and removal of debris within Earth's orbit. Armed with robotic arms and an arsenal of sophisticated technologies, these spacecraft navigate the intricacies of the space environment. Within the framework of visual servoing, Astroscale integrates cutting-edge algorithms and dynamic control systems, orchestrating the nuanced phases of approach, capture, and debris removal with unparalleled finesse. The infusion of visual servoing serves not only to refine the precision of maneuvers but also to ensure the safety and optimization of interactions with space debris, setting new benchmarks in space debris mitigation strategies.

GITAI, on the other hand, stands as a pioneering force in the realm of humanoid robotics tailored for space missions. Their avant-garde systems, seamlessly integrating artificial intelligence and advanced robotics, have redefined the scope of tasks achievable in space. From the maintenance of space infrastructure to the manipulation of intricate tools, GITAI's humanoid robots exhibit versatility and adaptability. Within GITAI's overarching philosophy, visual servoing stands as a linchpin technology. By harnessing visual information, GITAI's humanoid robots can adeptly recognize objects, orient themselves spatially, and execute precise operations with a level of accuracy unparalleled in space robotics. The dynamic adaptability afforded by visual servoing establishes GITAI as a formidable contender in addressing complex tasks, epitomizing flexibility and precision in the space exploration domain.

3.5.1 Astroscale

Astroscale, at the forefront of addressing the escalating issue of space debris, distinguishes itself through the development of advanced spacecraft designed to capture and remove debris in Earth's orbit. Equipped with robotic arms and sophisticated technologies, these spacecraft navigate the complex space environment. Within the framework of visual servoing, Astroscale integrates this technology to orchestrate the intricate phases of approach, capture, and debris removal. Visual servoing proves to be a fundamental pillar, enriching operations with dynamic and detailed control, improving maneuver precision, and ensuring safe and optimized interactions with space debris.
Active Debris Removal (ADR), also known as debris removal, encompasses a set of technologies and strategies developed to address the growing issue of space debris in orbit around Earth. Space debris comprises non-functional objects such as decommissioned satellites or launch vehicle components that remain in orbit and pose a threat to operational satellites and future space missions. The key features and objectives of ADR include the identification and monitoring of space debris using advanced technologies like radar and optical telescopes, the characterization of their physical properties, and the acceleration of their deorbit. This is achieved through the use of specially designed spacecraft equipped with robotic arms or other devices to grasp and push debris out of orbit. Astroscale, one of the leading companies in this field, has developed projects such as *ADRAS-J* (Active Debris Removal by Astroscale-Japan), aiming to demonstrate the feasibility of removing space debris like decommissioned rocket stages. ADR represents a crucial aspect of responsible space management and the prevention of space debris proliferation.

This process not only contributes to increasing the sustainability of our planet, but also represents an intricate and well-coordinated mission. It begins with the rendezvous of the upper stage rocket, such as the JAXA H2-A, proceeding until the crucial moment of the approach. During this delicate phase, the spacecraft follows an elliptical trajectory to ensure maximum safety of the satellite involved. The entire process is made possible thanks to the advanced use of technologies such as VISCam navigation, IRcam navigation and LiDAR Navigation.

- VISCam navigation refers to visual navigation, which uses cameras on board the spacecraft to recognize and track the target, providing crucial information for orientation and control during the approach.

- IRcam navigation is based on using infrared cameras to detect infrared radiation emitted by objects in space. This allows the spacecraft to

Figure 3.12: ADRAS-J

accurately identify the target even in low light conditions or when the target is not visible in visible light.

- LiDAR Navigation uses LiDAR (Light Detection and Ranging) technologies to measure distance and create high-resolution three-dimensional maps of the surrounding environment. This allows the spacecraft to navigate precisely and safely through the space environment, avoiding obstacles and ensuring an accurate approach to the target.

The combination of these advanced technologies allows the spacecraft to precisely navigate around the satellite or space debris to be removed. During this phase, detailed images of the target are captured, and the spacecraft performs a full rotation to obtain a 360-degree view of the object to be removed. The culmination of this process occurs during the final approach, when capture techniques are implemented to ensure safe and efficient removal. In particular, in the context of missions such as ADRAS-J (Active Debris Removal by Astroscale-Japan), these advanced technologies play a crucial role in ensuring mission success and contributing to the goal of reducing the impact of space debris on our Earth orbit.

[2] [1]

3.5.2 GITAI

GITAI, a pioneer in the development of humanoid robots for space missions, designs cutting-edge systems that integrate artificial intelligence and

advanced robotics. These humanoid robots are crafted to tackle a wide range of tasks, from maintaining space infrastructure to manipulating sophisticated tools. Visual servoing technology stands out as a cornerstone in GITAI's philosophy. By employing visual servoing, GITAI's humanoid robots leverage visual information to recognize objects, orient themselves in space, and conduct precise operations. The ability to dynamically adapt to the space environment through visual servoing allows GITAI to address complex tasks with remarkable flexibility and precision.
Let's now get into the specifics. Within the context of advanced robotics, visual servoing emerges as a fundamental element for companies such as GITAI, specialized in the production of humanoid robots and, in particular, mechanical arms. This technology plays a crucial role in facilitating complex tasks such as locating objects to be moved or modified, using easily recognizable reference points, such as markers. In the case of GITAI, which stands out as a large company in the sector, the connection with visual servoing becomes essential to guarantee the precision and efficiency of the operations carried out by its humanoid robots and mechanical arms.

The synergy between visual servoing and the production of humanoid robots becomes evident in the ability to recognize objects through markers, which act as key viewpoints. This ability is crucial for tasks of manipulation and interaction with the surrounding environment. The use of markers not only simplifies the localization of objects, but also allows real-time visual control, improving the precision of the actions performed by the robots. Visual servoing technology, therefore, becomes a strategic component in the design and programming of the robots produced by GITAI.

The presence of a mechanical arm in GITAI's portfolio further highlights the importance of visual servoing. This type of component is particularly useful for performing a wide range of tasks, from docking spacecraft to manipulating payloads. The ability of the mechanical arm to dynamically adapt to the environment, using visual information to recognize markers and reference points, amplifies the effectiveness of operations in space.

Focusing on GITAI's latest standout innovation, we examine the so-called "Inchworm Robot", a creation designed specifically to operate in space stations and satellites. This robot, characterized by a shape similar to that of a caterpillar, presents itself as the ideal solution to face the challenges of adverse environments and the peculiar problems of extra-terrestrial space. Its targeted design makes it particularly suitable for a wide range of applications within maintenance and repair operations, positioning itself as a key component for ISAM activities. From a structural point of view, the Inchworm Robot stands out for its 2 meter long arm with 7 degrees of freedom, giving

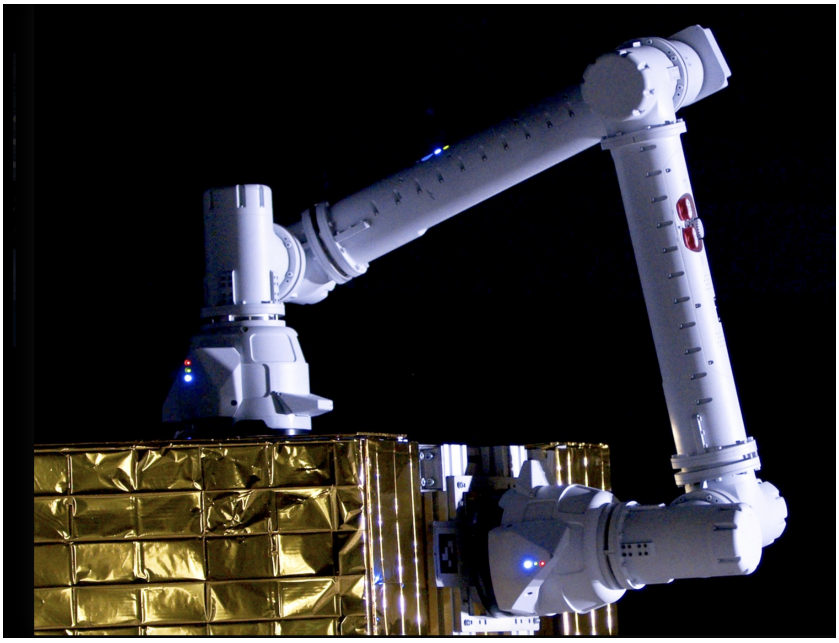

Figure 3.13: Inchworm GITAI

it a wide reach and exceptional flexibility to carry out a variety of operations. Despite the complexity of its potential, the robot is mainly controlled by advanced sensors, operating almost autonomously. Its autonomy, fundamental for operating in dynamic spatial environments, is optimized thanks to the strategic use of AruCo markers. These are strategically positioned in places of interest, making it easier to recognize the objects to be repaired or ISAM applications in general.

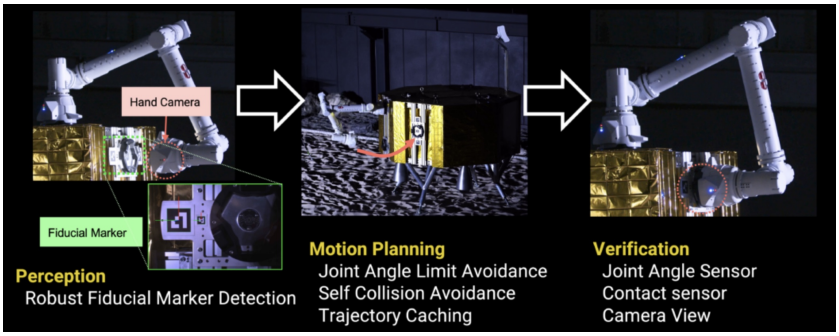

Figure 3.14: Approach to Marker AruCo

The combination of an advanced mechanical structure, considerable autonomy and the integration of Aruco markers makes the Inchworm Robot an

indispensable element for maintenance and repair activities in space. Its ability to operate relatively autonomously, guided by the visual data provided by the Aruco markers, underlines the versatility and effectiveness of this latest creation from GITAI. In a context in which ISAM plays an increasingly relevant role, the Inchworm Robot presents itself as a key resource for the management and maintenance of space infrastructures.

[31] [13]

Chapter 4

Conclusion

As we draw the curtain on this comprehensive thesis, our exploration has unfolded across three intricately woven chapters, each contributing a nuanced layer to our understanding of the complex landscape of aerospace robotics. In the inaugural section, we meticulously navigated through the state of the art, dissecting the multifaceted structures of aerospace robots. Our focus honed in on the pivotal role played by In-space Servicing, Assembly, and Manufacturing (ISAM), with a spotlight on the revolutionary concept of Popup Robots. This initial analysis served as a robust foundation, unveiling the unique challenges embedded in the aerospace environment and introducing innovative technologies poised to overcome these hurdles.

Transitioning seamlessly into the second chapter, our inquiry pivoted towards the celestial intricacies of orbital trajectory planning, an indispensable facet in orchestrating the precise spatial ballet of Popup Robots. The exploration of fundamental laws of physics, coupled with orbital control strategies, shed light on the intricate maneuvers such as trajectory corrections and rendezvous – maneuvers indispensable for the seamless execution of sophisticated space operations.

In the final chapter, we plunged into the heart of robotic control, casting a specialized lens on trajectory planning and visual control. Through avant-garde approaches like ArUco marker recognition and the strategic implementation of YOLO, we not only outlined advanced strategies but tailored them specifically to enhance the operational performance of robots, with a laser focus on their application in Popup Robots.

As we conclude this exhaustive journey, our insights transcend the mere theoretical framework, envisioning a trajectory towards practical and impactful applications in aerospace robotics. This research, undoubtedly, stands as a testament to the dynamic interplay between theory and practice, offering a roadmap for continued innovation and progress in the ever-evolving realm

of aerospace robotics.
In concluding this exploration, the trajectory of space robotics unfolds before us like an unwritten manuscript, teeming with promise and ripe with opportunities for technological advancement. The relentless march of innovation, marked by breakthroughs in artificial intelligence, materials science, and autonomous systems, is steering us toward a future where the frontiers of space exploration are both expansive and transformative.

Technological evolution stands as the cornerstone of our journey into the unknown, paving the way for scientific discoveries that transcend our current understanding of the cosmos. As we peer into the future, we are fueled by the exciting prospect of not only unraveling the mysteries of distant celestial bodies but also harnessing this knowledge for practical applications that benefit humanity. From satellite servicing missions to the assembly of structures in orbit, the practical implications of space robotics are poised to redefine how we interact with and utilize the space environment.

The assurance with which we gaze into the future is grounded not only in the technological prowess at our disposal but also in the resilience and adaptability of human ingenuity. The endeavors of the past and the ongoing missions have laid a robust foundation, and the experiences garnered from these missions provide invaluable insights for navigating the challenges that lie ahead.

In this evolving narrative, we anticipate that the nascent frontiers of space robotics will not only match but surpass the intrigue and significance of missions preceding them. As we embark on future explorations, we do so with the conviction that the symbiosis of human curiosity, scientific exploration, and technological innovation will propel us into a new era of unprecedented possibilities, reshaping our understanding of the cosmos and our place within it.

Acknowledgements

Solitamente, non ho l'abitudine di esprimere gratitudine quando si tratta di studio o università. Ho sempre affrontato tutto con le mie risorse, senza ricevere supporto economico o lezioni private da nessuno. Tuttavia, giunta al termine del percorso, ho compreso che l'università va oltre il pagamento delle tasse o il risultato di un esame. La componente psicologica dell'esperienza universitaria è fondamentale, uno stress continuo e prolungato che raramente ho sperimentato lungo il mio cammino. Per questo motivo, desidero rivolgere i miei ringraziamenti alle persone che mi hanno concesso un sollievo, anche se solo momentaneo o temporaneo, su alcuni frammenti o sull'intero percorso. È a loro che va il mio riconoscimento per avermi sostenuto durante questo viaggio, rendendo più leggero il peso dello stress universitario.

Un sentito grazie va a quegli amici che hanno reso il mio percorso universitario un'avventura indimenticabile. Nel lontano 2017, abbiamo iniziato il nostro viaggio paragonabile alle strade del sud: pieno di buche e senza autostrade. "RandomNameTillNextEvent" ha passato molte giornate in aula studio, con la paura degli esami che incombeva su di noi che ci ha unito come un club degli alcolisti anonimi per studenti. Tra viaggi, escursioni e chiacchiere senza senso, quegli 11 membri del gruppo hanno reso tutto più leggero. Anche se oggi molti hanno preso strade diverse, vorrei ringraziare in modo speciale Chiara e Giorgio. Chiara, compagna di studio e complice nei karaoke con Baby K. Le serate passate a casa sua tra lo studio e i suoi minestroni imbattibili sono state a dir poco indispensabili. Giorgio, sempre pronto a soccorrere chi era a pezzi dopo un esame andato male o a calmare l'ansia pre-esame. I cornetti mattutini la domenica hanno reso più sopportabili le lunghe sessioni di studio. Senza di loro, questa magistrale sarebbe stata molto più complicata.

Un sincero ringraziamento va a Nonna Giusy, la persona che chiamavo prima e dopo ogni esame. La sua straordinaria capacità di sdrammatizzare avrebbe potuto tranquillizzare chiunque, indipendentemente dal risultato dell'esame, perché, come diceva lei, "puoi sempre ridarlo". Grazie per tutti i ceri accesi in chiesa, credo che il 90 % dei miei successi accademici sia

dovuto a quelle candele "Gesù approved".

Un grazie speciale va anche a Nonno Beppe, purtroppo non glielo posso dire di persona ma so che più di tutti credeva che ce l'avrei fatta. Dedico a lui quel 22 in Disegno e Progettazione, i 6 CFU più terribili di tutto il mio percorso di studi. Sappiamo entrambi che senza la sua capacità nel disegnare e progettare, imparata nei suoi 40 anni in Fiat, non lo avrei mai superato. Nel vero senso della parola. Ma quella dell'esame è una storia che, per ragioni legali, non possiamo raccontare qui. Chi vuole intendere intenda.

Infine è necessario ringraziare una persona che ho conosciuto quasi due anni fa ormai, e che nell'ultimo anno è diventata la persona più importante della mia vita. Ho sempre ritenuto che scrivere tra i ringraziamenti la dedica al proprio fidanzato fosse la cosa più sdolcinata e nauseabonda sulla faccia della terra, ma Alessio è subentrato nella mia vita in un periodo non molto facile. In quel periodo, intravedevo la conclusione del mio percorso accademico, con soli due esami rimasti alla laurea. Purtroppo, la vita ha preso una piega complicata: un mese dopo aver cominciato un nuovo lavoro a tempo pieno che già mi rubava molto tempo ho trascorso diversi giorni in ospedale a causa di un problema di salute. Lo stress era alto, il nervosismo pure e io avevo deciso di mollare. "Grande, tu si che partorisci delle idee geniali a volte! Mollare a due esami dalla laurea, questa intelligenza l'hai imparata al Politecnico?". Alessio ha lottato al posto mio quando non riuscivo ad aprire un libro nemmeno per sbaglio. Ha dato 90 in una cosa in cui io riuscivo a dare 10, nonostante non interessasse la sua persona. Ha insistito e mi ha fatto ragionare ragionare quando di studiare, dopo 8 ore di riunioni, proprio non ne avevo voglia. Ed eccoci qua, a concludere questo lungo e travagliato percorso e a ringraziare la persona che più amo. A ringraziare quel ragazzo che mi ha insegnato che l'amore, quello vero, non è tossico e ti supporta e ti sprona a fare il meglio possibile con le forze che si hanno. A ringraziare quella persona che mi ha insegnato che l'età è davvero solo un numero, e che quei 6 anni in meno non ti rendono una persona meno matura, anzi. A ringraziare il mio copilota con la speranza che resterà tale per ancora molto tempo, il solo a cui farei guidare la mia Ferrari F40 appena comprata. Ricorderò per il resto della vita il giorno in cui ho visto il voto dell'ultimo esame, la soddisfazione di averlo passato, le lacrime e la gioia sul tuo volto accompagnata dalla frase "Amo però io una cosa te la devo dire, sono veramente felice che tu abbia finito perché sei davvero insopportabile in sessione esami." Grazie per tutto piccola tartaruga scaduta.

E infine, ringrazio tutti gli elfi. Ciao.

Bibliography

[1] Astroscale. *Active Debris Removal (ADR)*. URL: https://astroscale.com/services/active-debris-removal-adr/. (accessed: 13.03.2024).

[2] Astroscale. *ADRAS-J*. URL: https://astroscale.com/missions/adras-j/. (accessed: 13.03.2024).

[3] Astrospace. URL: https://www.astrospace.it/2020/08/19/ingenuity-e-perseverance-correggono-la-loro-traiettoria-per-la-prima-volta/. (accessed: 18.12.2023).

[4] Marco Bertaggia. "Confronto di algoritmi di pianificazione di traiettoria sampling-based per robot mobili". In: (2017-2018).

[5] Luigi Villani Bruno Siciliano Lorenzo Sciavicco and Giuseppe Oriolo. *Robotics. Modeling, Planning and Control*. Springer, 2010.

[6] Maria Teresa Calcagni. "Sistema di rilievo ostacoli per applicazione in robot collaborativi". In: (2019-2020).

[7] Massimo Canale. *Automatic Control*. followed in 2021/2023.

[8] Roberto Opromolla Claudio Vela Giancarmine Fasano. "Pose determination of passively cooperative spacecraft in close proximity using a monocular camera and AruCo markers". In: (2022).

[9] ESA. URL: https://www.eoportal.org/satellite-missions/spacelab#metric-camera. (accessed: 9.02.2024).

[10] ESA. *SpaceLab data book*. 1983.

[11] Francesco Gambino. "Development of the control system for an Inflatable Robot". In: (2021-2022).

[12] Matthew A. Garratt. "Obstacle Avoidance in Cluttered Environments using Optic Flow". In: (2009).

[13] GITAI. *GITAI Inchworm Robot*. URL: https://gitai.tech/inchworm-robot/. (accessed: 13.03.2024).

[14] H2T. URL: https://h2t.anthropomatik.kit.edu/english/378.php. (accessed: 5.01.2023).

[15] Christian Wielgosz e J-C Thomas. *Deflections of inflatable fabric panels at high pressure*. 2002, pp. 523–536.

[16] Justin R. Jackson. "In-space Servicing, Assembly and Manufacturing (ISAM)". In: ().

[17] ZHAO Yu-hui HOU Xi-yun LIU Lin. "YError Analysis and Trajectory Correction Maneuvers of Lunar Transfer Orbit". In: (2013).

[18] Rhonda Martens. *Kepler's Philosophy and the New Astronomy*. Princeton University Press, 2000, pp. 10–38.

[19] NASA. URL: https://gpm.nasa.gov/science-team/resources/noaa19-satellite. (accessed: 18.12.2023).

[20] Nasa. URL: https://mars.nasa.gov/mars2020/. (accessed: 23.03.2023).

[21] Nasa. URL: https://www.nasa.gov/pdf/469616main_Robonaut2_factsheet.pdf. (accessed: 3.04.2023).

[22] Nasa. URL: https://mars.nasa.gov/resources/27377/perseverance-samples-berea/. (accessed: 1.05.2023).

[23] Alessio Nizzoli. "Progettazione e realizzazione di un sistema di asservimento visivo per un robot manipolatore". In: (2005-2006).

[24] Carlo Novara. *Nonlinear control of aerospace system*. followed in 2022/2023.

[25] OpenCV. URL: https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html. (accessed: 5.01.2023).

[26] Matteo Melchiorre Pierpaolo Palmieri and Stefano Mauro. "Design of a Lightweight and Deployable Soft Robotic Arm". In: (2022).

[27] Mario Troise Pierpaolo Palmieri Matteo Gaidano Andrea Ruggeri Laura Salamina and Stefano Mauro. "An Inflatable Robotic Assistant for On-board Ap- plications". In: (2021).

[28] Proquest. URL: https://www.proquest.com/docview/2500912726?pq-origsite=primo&sourcetype=Wire%20Feeds. (accessed: 18.12.2023).

[29] Alessandro Rizzo. *Robotics*. followed in 2021/2023.

[30] Paolo Rocco. URL: https://rocco.faculty.polimi.it/caut/Pianificazione%20del%20moto.pdf. (accessed: 6.02.2024).

[31] GITAI Hiring in the US. *Ground demonstration of GITAI autonomous robot for the tech-demo outside the ISS planned in 2024*. URL: https://www.youtube.com/watch?v=aEyb6iKgHPw. (accessed: 13.03.2024).

[32] Christos K. Verginis. "Reconfigurable Motion Planning and Control in Obstacle Cluttered Environments under Timed Temporal Tasks". In: (2013).

[33] P. Viola and M. Jones. *Rapid object detection using a boosted cas- cade of simple features*. 2001.

[34] Wikipedia. URL: `https://en.wikipedia.org/wiki/Space_manufacturing`. (accessed: 8.02.2024).

[35] Wikipedia. URL: `https://en.wikipedia.org/wiki/Archinaut`. (accessed: 8.02.2024).

[36] Wikipedia. URL: `https://www.nasa.gov/mission/on-orbit-servicing-assembly-and-manufacturing-2-osam-2/`. (accessed: 8.02.2024).

[37] Haibin Li Dengchao Wu Wenming Zhanga Cunjun Xiao. "YOLO- PL: Helmet wearing detection algorithm based on improved YOLOv4". In: (2024).