POLITECNICO DI TORINO

Master's Degree Course
in Mechatronic Engineering

Master Thesis

Inverse kinematics of hyper-redundant robots

Supervisors
Prof. Alessandro Rizzo
Dr. Carlo Canali

Candidate
Lorenzo Giustozzi

Accademic Year 2019-2020

Abstract

Redundancy represents one key towards design and synthesis of different types of versatile manipulators. Obstacle avoidance and limited joint range constitute two kinds of constraints which can be potentially met by a kinematically redundant robot. The natural scenario is the inverse kinematic problem which is certainly a crucial point for robotic manipulator analysis and control. Based on a recently proposed algorithmic solution techniques, the inverse kinematic problem for hyper-redundant manipulators is solved in this work of thesis. A formulation that allows to include the above mentioned constraints is presented; the result is an efficient, fast, closed-loop algorithm which only makes use of the direct kinematics of the manipulator. Extensive simulation results illustrate the tracking performance for a given trajectory in the task space, while guaranteeing a collision-free trajectory and not violating a mechanical joint limit. Experiments on a planar redundant robot show the effectiveness of the solution in a real case.

Acknowledgements

Finally the long awaited day has come, writing these acknowledgments is the finishing touch not only to my thesis work, but also to this university career. It has been a long and challenging period of my life, full of satisfactions and disappointments, which allows me to personally grow. I must not forget what I learned from this experience, the beauty of change and the modesty of the intelligence. This is the starting point of the rest of the life. Moreover it has been a period of growth from the scientific point of view. Above all, the internship allows me to understand the complexity behind the development and realization of a real case application, within a completely new scenario, much more dynamic with respect to an academic one.

I would like to spend few words to all the people who supported and helped me during this period. Firstly, I would like to thank all the colleagues met during this internship at Italian Institute of Technology. In particular, I turn to C. Canali, that gave me the chance to work to this project, providing me with everything I needed to understand the project goals to reach. A special thank goes to D. Ludovico, a kind of "mentor", for all the support to the technical development of the thesis work. I would not have been able to get the desired results without his help. It must be mentioned P. Guardiani for proposing me this thesis opportunity and for the support during the period in IIT and A. Pistone for the help in the last period of experiments. From the academic side, I would like to thanks my supervisior A. Rizzo, that despite all the problem faced during the work, has always been by my side, recommending the best for the realization of a good work. Obviously a special thanks goes to my family, that gave me the possibility to reach this important milestone, supporting me during the darkest periods of all this trip and always believing in me. An other special thanks goes to my sister Francesca, which has always supported me through the difficult moments and she has pushed me to reach the best version of myself. Last but not least, I would like to thank all my friends. We always supported each other, in moments of joy and sadness, spurring ourselves to reach the goal.

A heartfelt thanks to all!

Lorenzo Giustozzi

Contents

List of Tables 5

List of Figures 6

1 Introduction 9

2 State Of Art 16

3 Inverse Kinematics of Hyper-Redundant Robots 21
- 3.1 Proposed algorithm . . . 22
  - 3.1.1 Joint Limits as primary task . . . 22
  - 3.1.2 Obstacle avoidance as primary task . . . 23
  - 3.1.3 Three tasks solution . . . 28
  - 3.1.4 Exact Solution . . . 29

4 Simulations 32
- 4.1 Simulation . . . 32
  - 4.1.1 Robot Configuration . . . 32
  - 4.1.2 Toroidal Environment . . . 33
  - 4.1.3 Obstacle Modelling . . . 34
  - 4.1.4 Cubic Spline . . . 35
  - 4.1.5 Simulation Results . . . 37

5 Experiments 46
- 5.1 Simulation for experiments . . . 47
- 5.2 Experiments . . . 52

6 Conclusion 56

List of Tables

4.1 joint limit table 33
4.2 obstacle table 35

List of Figures

1.1 Kuka manipulator . . . 9
1.2 iCub,Spot . . . 10
1.3 The heelbarrow Mk8 plus used by Italian Army as bomb disposal unit . . . 11
1.4 Eddy Current Array Semiautomated Solution for Pipeline Surface Cracking Inspection by Olympus . . . 11
1.5 HoneyBee pipe inspection robot sliding through a small diameter pipe by means of its mecanum magnetic wheels. . . . 12
1.6 Series II X125 System and planar Snake-arm . . . 12
1.7 Hyper-Redundant prototype at IIT . . . 14
3.1 Distance from obstacle . . . 24
3.2 Point-line distance . . . 25
4.1 Example of Torus environment . . . 34
4.2 Inspection environment . . . 34
4.3 Cubic Spline . . . 36
4.4 Toroidal environment simulation 1 . . . 37
4.5 Toroidal environment simulation 2 . . . 38
4.6 Toroidal environment simulation 3 . . . 38
4.7 Toroidal environment simulation 4 . . . 39
4.8 Toroidal environment simulation 5 . . . 39
4.9 Toroidal environment simulation 6 . . . 40
4.10 Toroidal environment simulation 7 . . . 40
4.11 Toroidal environment simulation 8 . . . 41
4.12 End-Effector pose error . . . 42
4.13 Joint limits . . . 43
4.14 Obstacle 1 distance . . . 44
5.1 6 joints planar robot prototype . . . 47
5.2 6 joints planar robot simulation 1 . . . 48
5.3 6 joints planar robot simulation 2 . . . 48
5.4 6 joints planar robot simulation 3 . . . 49
5.5 6 joints planar robot simulation 4 . . . 49

5.6 End-effector error of 6 joints planar prototype simulation 50
5.7 Joint angles of 6 joints planar prototype simulation 51
5.8 Distance from obstacles of 6 joints planar prototype simulation . . . 52
5.9 Labview control panel 53
5.10 Engines behavior during experiment 54
5.11 6 joints robot prototype experiment 54

Chapter 1

Introduction

Nowadays, robotics is a field under continuous expansion. In recent years, the studies dealt with the development of robots for a series of different applications related to various domains. For example medical robots, where the major purpose is for surgery employment, or in military field, whose applications refer to transport, search rescue and attack, as well as agriculture robots, used for precision tasks and environmental monitoring and industrial field. The latter case is about the use of industrial manipulator, that are employed in tasks as assembly, disassembly, pick and place, welding, painting and manipulation. A typical example is the application in automotive industries, where robotic arms perform high precision operation of different complexities.

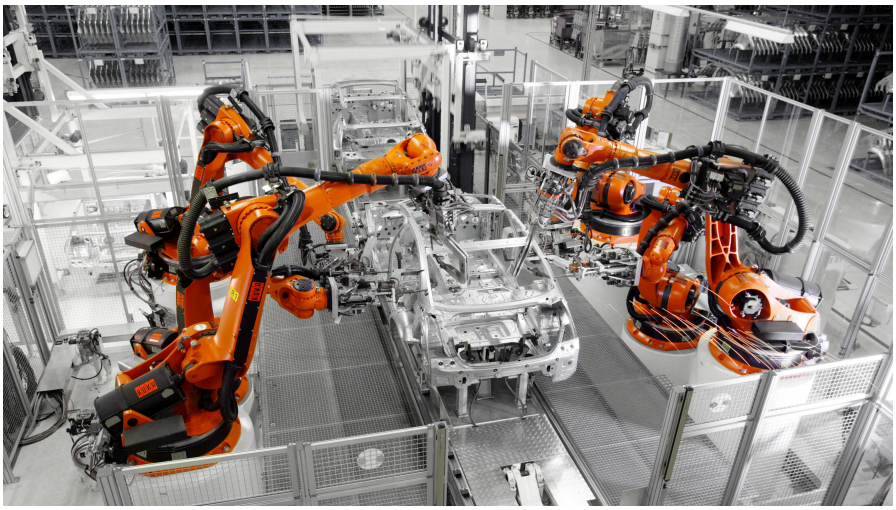

Figure 1.1: Kuka antropomorphic manipulators used in automotive applicatio

According to the different types of operations that the robot should perform,

the most recent studies focused on the development of a series of different family of robots, which are strictly related to the type of task to be executed and also to the working environment. Similarly to anthropomorphic robotic arm, that are usually composed by shoulder, elbow and wrist, humanoid robots or animal-like ones are bio-inspired as shown in Fig. 1.2. This family of robots are inspired by biological systems with the goal of learning concepts from nature and applying them to the design of real-world engineered systems. In this field applications are huge since, once developed to a proper technology readiness level, bio-inspired robot could benefit from both the flexibility and adaptability of biological systems and the reliability of a mechatronics system.

Figure 1.2: iCub from IIT and Spot from Boston Dynamics

As well as the above mentioned applications, it could be interesting to deeply analyze the industrial robot for inspection and maintenance purposes that are the main topic of this thesis work. Robots used in inspection have several advantage: above all they can substitute or can assist human operators in many different situations. Some examples are: dangerous environments like disaster scenarios where working conditions are not ideal for human safety, confined spaces difficult to be reached by humans such as vessels, pipes, combustion chambers or similar structures. A typical example are the robots used as bomb disposal as shown in Fig. 1.3: in this case the robot is teleoperated by a human situated into a safe zone and, thanks to the use of wheels, cameras and articulated arms, it can be used to inspect suspect explosive manufactures, minefields or other hazardous objects

Differently from robots used in assembly tasks (such as the already mentioned anthropomorphic robots), inspection robots are very often designed starting from the task that they have to perform: if the robot need to inspect pipeline surfaces it could have a magnetic adhesion system ad being equipped with ultrasound sensors (see Fig.1.4), on the contrary if the robot needs to inspect pipes from the inside it could be necessary to have a reduced size as shown in Fig. 1.5

Particularly challenging environments for robot inspections are all that areas

Figure 1.3: The heelbarrow Mk8 plus used by Italian Army as bomb disposal unit

Figure 1.4: Eddy Current Array Semiautomated Solution for Pipeline Surface Cracking Inspection by Olympus

where wide spaces are accessible only from a small aperture (for example a manhole) and, at the same time, surfaces are not suitable for magnetic adhesion. In this circumstances the use of drones can be a valid option , but if precise or contact measurements need to be performed the use of snake robots can lead to better results. The challenge of confined space environments is to reach a desired pose to perform the specific task, by taking into account the complexity of the surrounding environment. This could be difficult because of the presence of obstacles around the area to be reached. The above mentioned point is discussed in this thesis where the problem of navigating a snake robot into a complex environment is faced with particular attention to the kinematics of the robot. Different types of robots have been developed in recent years, which are characterized by an high

Figure 1.5: HoneyBee pipe inspection robot sliding through a small diameter pipe by means of its mecanum magnetic wheels.

number of joints in order to guarantee flexibility and dexterity during the execution of the inspection purposes. These type of robots are called hyper-redundant. Some examples are the Series II X125 System of OC Robotics and the Planar snake-arm robot from Petrobot project (see Fig.1.6).

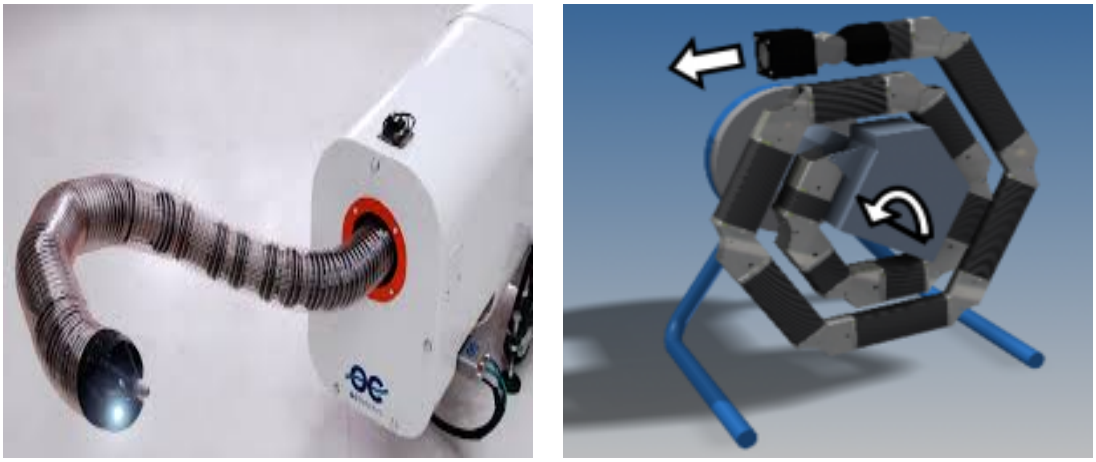

Figure 1.6: Series II X125 Sytem and planar Snake-arm

The configuration of these robots depend on the environment where the inspection task must be accomplished.

In general a robot should be able to perform a desired trajectory in order to reach a desired point in the working environment. The trajectory can be represented by a sequence of point to be followed by the robot. Reaching a specific point in the space for inspection application is the basic concept of this work of

thesis, as stated by the Inverse Kinematics (IK) theory. IK problem aims to determine the joint configuration corresponding to a given end-effector position and orientation. The end-effector is the final part of the robot where, usually, is fixed the tool that performs the task, e.g. a camera in inspection application. The solution to IK problem is of fundamental importance in order to transform the desired end-effector motion into the corresponding joints motion. This concepts pose an interesting challenge in the case of snake robots where the number of joints could be particularly high.

This is the case of the so called "hyper-redundant robots" that have an high number of joints and, as a consequence, they exhibit a certain number of redundant degrees of freedom (DoF). In this case, the inverse kinematics problem is more complex with respect to a non-redundant robot, because it could have infinite solutions. On the other hand, it is possible to exploit the available degrees of redundancy to take into account additional tasks, which depends on the configuration of the robot and the space to be inspected. While the robot is performing the operation, it is important to take into account the presence of obstacles. Thus, obstacle avoidance is one of the possible tasks to be included in the IK solution. Furthermore, the joint limit could be specified as another task, due to the physical limits of the actuators and the robot structure. Finally, it is important to point out that different methods can be applied in order to solve inverse kinematics problem for hyper redundant robots.

To sum up, the main objective of this work of thesis is the development of an Inverse Kinematics algorithm for hyper-redundant robots employed in inspection applications, which must include the obstacle avoidance and joint limits together with the end-effector position and orientation. A set of methods will be analyzed in order to choose the best solution for the algorithm.

After a simulation phase, the algorithm has been tested on a planar hyper-redundant robot prototype available at the Italian Institute of Technology, shown in figure 1.7.

Figure 1.7: An image of snake-arm robot prototype composed by 6 revolute joints

The thesis is organized as follows:

- in chapter 1 is given an overview of the different types of robots, introducing the relevance of inverse kinematics problem.
- in chapter 2 is presented the state of the art of the inverse kinematics solutions, focusing on the hyper-redundant robots case.
- in chapter 3 is described the contribution to find a solution to the inverse kinematics problem that satisfy multiple concurrent tasks.
- in chapter 4 are shown the results of a toroidal environment inspection with a 13 revolute joints planar hyper-redundant robot, using the solution found in chapter 3.
- in chapter 5 the experimental phase is presented regarding a robot prototype available in IIT.
- in chapter 6 the conclusion of the work of thesis are provided and future possible works are presented.

Chapter 2

State Of Art

In this chapter the well-known theory behind the Inverse Kinematics (IK) will be analyzed, in order to establish the basis for the successive development of the algorithmic solution.

Direct Kinematics (DK) is based on the definition of the joint configuration that allow a certain pose of the end-effector and viceversa for the IK.

The pose of the robot with respect to the origin frame is defined by means of homogeneous transformation matrix (HTM). The relative structure is a block matrix that include translation and rotation of the i-th joint, starting from the first one until the end-effector. Therefore, with the HTM, it is possible to obtain the current configuration of the robot.

Each joint allows a motion in one direction or a rotation around an axis. Each admissible motion corresponds to a degree of freedom (DoF). The manipulator has a number of DoF related to the number of joints and consequently a number of joint variables.

In order to compute the DK and thus the transformation coordinate of the manipulator, the Denavit-Hartenberg (DH) convention is generally used, that provides a systematic method to define the relative position and orientation between consecutive links.

Inverse Kinematics problem is based on finding the joint variables that correspond to a desired end-effector pose. Thus, finding the solution of the problem is crucial in order to transform the motion specification of the end-effector into the corresponding joints motion.

Whereas in DK problem, where the solution for determining end-effector pose is unique whether the joint variables are known, on the contrary, usually, the IK problem is more complex, because closed-form solution is not always obtainable or infinite solutions may exist.

The space where the end-effector tasks have to be defined is termed Operational Space (OSP), which dimension in given by m. Given the above, a portion of the

OSP is the Workspace, that is the part of the space reachable by any motion of all the joints.

The definition of the task to be executed by the end-effector in the OSP requires a certain number of variables, which dimension is r, that could be lower than m. On one side, the joint variables are defined in the Joint space of dimension n. Thus, the IK can be seen as a mapping between the operational space and the joint space. On the other hand, Differential Kinematics establishes the relationship between the joints velocities and the corresponding end-effector linear and angular velocities described by a matrix, named geometric Jacobian, which depends on the manipulator configuration

$$v_e = J(q)\,\dot{q} \tag{2.1}$$

where v_e is a $(r \times 1)$ vector of end-effector velocity for a specific task, J is the corresponding $(r \times n)$ Jacobian matrix and $\dot{q}$ is the $(n \times 1)$ vector of joint velocities.

Instead the analytical Jacobian could be computed via differentiation of the direct kinematics function, if the end-effector pose is expressed with a minimal representation in the operational space.

A robot is redundant when it has the number of DoF greater than the number of variables needed to define a task in the operational space.

Therefore, the relation between the number n of DOFs of the structure, the number m of OS variables, and the number r of OS variables necessary to define the desired task are taken into account through the Differential Kinematics.

If $r < n$ the robot is kinematically redundant and there exist $n - r$ redundant DoF. In equation 2.1, the Jacobian has to be considered as a constant matrix, as the instantaneous velocity mapping is of interest for a given pose. The mentioned mapping is defined with respect to two spaces.

The range of J, $R(J)$, is the subspace of the end-effector velocities that can be generated by the joint velocities while the second one is the Null space of J, that is the subspace $N(J)$ in R^n of joint velocities that do not produce any end-effector velocity. The existence of the Null space is important in the redundant manipulator case, as allows to specify different strategies to handle multiple tasks.

Introducing inverse differential kinematics, equations represent a linear mapping between the joints velocities space and the operational space velocities. Whereas $n = r$, the joint velocities can be obtained by the simple inversion of the Jacobian matrix:

$$\dot{q} = J^{-1}(q)\,v_e \tag{2.2}$$

This technique requires that the Jacobian is square and full rank. If the condition is verified, the correspondent joint position could be computed using Euler integration method [16]:

$$q(t_{k+1}) = q(t_k) + \dot{q}(t_k)\,\Delta t \tag{2.3}$$

If the manipulator is redundant $n < r$ and the Jacobian matrix has more columns than rows and infinite solution exist to (2.2). A solution could be to formulate the problem as a constrained linear optimization problem.

Following the definition of the IK problem as a constrained linear optimization problem, the solution found in [16][15] is given as

$$\dot{q} = J^{\dagger} v_e + \left(I_n - J^{\dagger} J \right) \dot{q}_0 \tag{2.4}$$

The obtained formulation is important as allows, at the same time, to minimize the norm joint velocities and to try to satisfy additional constraints. The matrix $(I_n - J^{\dagger}J)$ is Null Space projector and is one of the key concept in redundant case, as it allows to generate motion of the manipulator's body without changing position or orientation of the end-effector. It is possible to choose $\dot{q}_0$ to define a secondary objective function.

To solve the IK problem as an iterative algorithm, the equations have to be solved making use of numerical method [16] . Thus, the joint variables at subsequent time is computed starting from the value at previous time instant, leading into the following equation

$$q\left(t_{k+1}\right) = q\left(_{t_k}\right) + J^{-1}\left(q\left(_{t_k}\right)\right) v_e\left(t_k\right) \triangle t \tag{2.5}$$

Using numerical method, problems could arise due to numerical integration. Thus, the numerical solution does not coincide with the continuous time one and the difference between the two solutions originate an operational space error.

The relation between $\dot{q}$ and e gives a differential equation that allows to describe error evolution over time.

Different methods has been developed to face the IK problem. The Jacobian transpose method is based on the use the transpose of J instead of the inverse of J. The algorithm has been studied by various authors as [2] and [16].

From what concern the cited studies, the algorithm results to be computationally efficient as it need to compute the transpose of a matrix with respect to an inversion, but is usually preferred with the Pseudo-Inverse method for the velocity to the convergence to the solution and the possibility to include secondary tasks together with the desired pose of the end-effector with the Null-Space projection.

The Jacobian pseudo-inverse method allows to include other tasks together with the end-effector pose. In redundant case, it is possible to define the solution as

$$\dot{q} = J_A^{\dagger}(q)\left(\dot{x}_d + Ke\right) + \left(I_n - J_A^{\dagger} J_A \right) \dot{q}_0 \tag{2.6}$$

where $J_A^{\dagger}$ is the pseudo-inverse of the Jacobian matrix and $(I_n - J_A^{\dagger} J_A)$ is the null-space operator that allows to project a secondary objective function into the null-Space of the primary solution.

The damped least squares method prevents many of the pseudo-inverse method's problems with singularities and can give a numerically stable method of selecting $\dot{q}$ [3]. the damped least squares solution is

$$\dot{q} = \left(J^T J + \lambda^2 I\right)^{-1} J^T e \tag{2.7}$$

where $J^T J$ is a $n \times n$ matrix, where n is the number of degrees of freedom of the manipulator and $\lambda \in \mathbb{R}$ is a non-zero damping constant. Moreover, the selection of the damping coefficient has been the basis of various studies as in [7][5], in such a way to be dynamically determined, depending on the configuration of the Robot.

In order to perform certain tasks for hyper-redundant robot, it could be relevant to set an higher priority to joint limits or obstacle avoidance while accepting a certain tolerance on the end-effector pose, or try to solve all the desired tasks concurrently.

Augmented Jacobian method is based on the extension of the Jacobian with the inclusion of additional tasks in such a way that redundancy can be conveniently exploited to solve the inverse kinematics problem together with obstacle avoidance and/or limited joint range.

CLICK algorithm based on the extended Jacobian has been developed in [4] that deal with obstacle avoidance taking into account of one joint at the time.

A different approach is to invert the priority of the secondary and primary task. In [6] the author set the joint limit as primary task and set pose of the end-effector as secondary task. Moreover, including an activation matrix in the solution, as shown in [12], it is possible to activate the primary task only when a joint is near to its limit, taking into account of the pose when the joints are distant from their own limits.

The same approach can be used in order to set obstacle avoidance as primary task as in [13] [18]. In this way the algorithm try to satisfy the desired pose of the end-effector while gives priority to avoid the obstacle that are in the surrounding environment.

An alternative approach that can be used is named follow-the-leader [17] [1], that is an ideal path planning method for hyper-redundant manipulator based on the concept that as the manipulator moves forward, all the sections follow the path that the end-effector has gone trough, which simplifies the obstacle avoidance.

In order to find a solution to the problem to IK for redundant robots that satisfy primary and secondary task concurrently, an approach is to find an algorithm for a multi-objective optimization, that try to optimize a group of conflicting objective functions simultaneously and obtain a group of Parallel Pareto optimal solution. The approach has been studied by various authors as in [9] based on genetic Algorithm, named Differential Evolution (DE), that provides a solution scheme for the complex optimization problems.

Chapter 3

Inverse Kinematics of Hyper-Redundant Robots

In robotics field, a crucial point for robotic manipulator analysis, is the transformation of task space coordinates into joint space coordinates, that is the solving of the IK problem.

Differential inverse kinematics has been presented in the state of art. As shown in (2.2), the Jacobian matrix represent a linear mapping between joint velocity space and task velocity space.

In case of redundant manipulators the Jacobian matrix is non-square and infinite solutions to the IK problem exist.

Nevertheless, it is possible to use the degrees of redundancy to deal with additional tasks, such as obstacle avoidance and joint limits. To include additional secondary tasks together with the end-effector desired pose, the null-space operator has been taking into account. Nevertheless the null-space projector does not guarantee to satisfy the second objective functions as it operates only taking advantage of the degrees of redundancy, since the solution for the pose has higher priority. Thus, the secondary tasks are not always satisfied.

Recalling that the aim of this work of thesis is the development of an algorithm for solving the inverse kinematics problem for hyper-redundant robot acting into confined spaces, it is important to fulfill with multiple tasks at the same time.

Consequently, the approach has been the research of different methods in order to satisfy the secondary tasks concurrently with the primary task, i.e. the pose of the end-effector. Moreover, in our case, obstacle avoidance and joint limits assume more relevance with respect to the pose objective.

As a matter of fact, in some applications it could be relevant to set an higher priority to joint limits or obstacle avoidance while accepting a certain tolerance on the end-effector pose. This is due to the the physical limits of the joints or

the obstacle to be avoided, thus it is significant to find a solution of the Inverse Kinematics problem taking into account those considerations.

3.1 Proposed algorithm

The proposed algorithm has to provide a solution of the IK problem considering the joint limit as primary task, the obstacle avoidance as secondary task and the end-effector pose as third task. The reasons behind the mentioned choice is that the considered hyper-redundant robot has physical joint limits that must be satisfied and the solution to IK has to be found managing, first of all, this task.

In order to accomplish with this requirement, the joint limits as primary task solution is considered.

The second priority task is the obstacle avoidance. Usually the inspection purposes are defined in restricted area and obstacles could be present in the environment. The obstacle avoidance as primary task solution is used to manage this task.

In this section will be explained the two methods taken into account, e.g. joint limit as primary task and obstacle avoidance as primary task. At the end will be showed how the methods has been fused together in order to obtain the final solution.

3.1.1 Joint Limits as primary task

In the state of art has been shown that joint limit task can be implemented as secondary objective function in the Null space, or included in the extended Jacobian. In the first approach such avoidance is not guaranteed while in the second problems can arise in a non feasible-solution or in algorithmic singularities. The only way to guarantee the satisfaction of joint limit avoidance is to set it as primary task and set the pose of the end-effector as secondary task [6]. To this purpose, it is defined $J_1 = H_m$ matrix, with m being the number of joints of the robots, and matrix H_m is expressed as follows

$$H_m = diag(h_\beta(q_i)) \tag{3.1}$$

where h_β is a continuous function, usually defined as piecewise function, that deactivates the joint when it reaches a specified threshold β from its limits. In this way, it is possible to define the main task error in the form of

$$e_1 = -\lambda_{JL} q \tag{3.2}$$

with λ_{JL} stand for a scalar weight factor to define the relevance of each joint limits. Usually, the range is $\lambda_{JL} \in [0.1\ 0.5]$.

The algorithmic increment $\dot{q}_1$ is zero when the joint angle is not near the correspondent limit and has a push-to-center value when is in the limit neighborhood. In order to find the version to be implemented as an algorithm, the null space method is applied here for the pose task, defining $J_2 = J$ that represent the Jacobian matrix of the robot and $e_2 = x_d - x$ as the positioning error in the operational space. So when the joints are far from their limits, the first task in null and a solution to the pose goal is to be founded. The algorithmic version of what has been discussed above is the following

$$\dot{q} = J_1 e_1 + (I_m - J_1^{\dagger} J_1) J_2 e_2 \tag{3.3}$$

Regarding the activation matrix to deal with the Joint limits, it is possible to define it as [12] $H = diag(h_1, ..., h_m)$, where h_i is

$$h_i = \begin{cases} 1 & \text{if } e_1 > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

In this approach the joint limits range is guaranteed with detriment to the precision of the pose goal achievement. In fact the convergence to the desired pose of the end-effector due to null-space operator is not guarantee or the solution could not exist due to singularities.

3.1.2 Obstacle avoidance as primary task

Although hyper-redundant robot can perform multiple tasks, it is not guaranteed the simultaneous fulfillement of all the desired tasks. In some case obstacle avoidance could assume an higher relevance with respect to the precision of the pose goal of the end-effector.

In order to manage the obstacle avoidance problem, it is important to understand if the manipulator is approaching an obstacle during the execution of the desired motion. Thus, it necessary to introduce a method to establish the distance of the robot and the obstacle.

Distance from obstacle

Introducing the distance from an obstacle, it is significant to take into account the shape that the obstacles could assume. Clearly it depends on the environment where the robot should perform the desired task or on the spaces where the robot should go through. Two assumptions are introduced: in this work of thesis the model of the obstacles will be considered as a sphere and that the position of each obstacles is a priori known.

Taking into account the distance between the robot and the obstacle, it is important to establish the point on the body of the robot that is at the minimum

distance from the obstacle. This point, from now, will be termed *critical point* p_0, represented in figure 3.1. The matrix A_0 is the corresponding Homogeneous Transformation matrix to the critical point.

Moreover, given the nearest point to the obstacle, which distance is d_0, it is needed to define a threshold, the critical distance d_m, to establish if the manipulator is too close to the obstacle and a velocity has to be assigned in order to move away from the obstacle. It is possible to define the quantity d_m depending on the distance that is desired to maintain with respect to the manipulator.

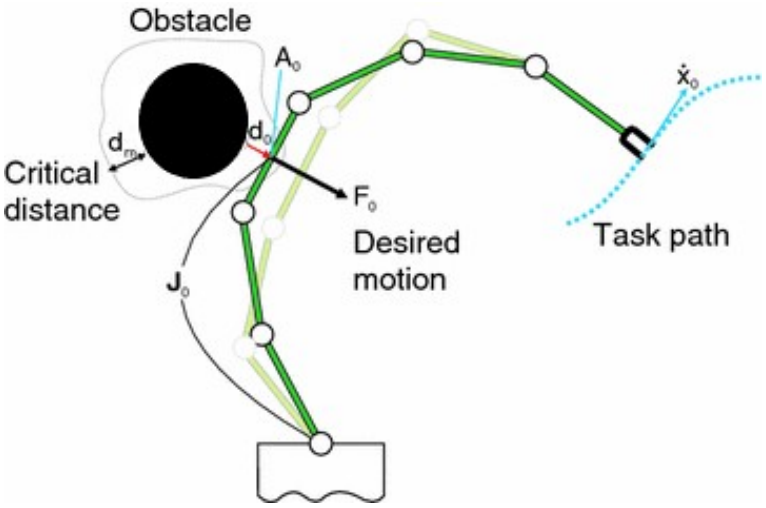

Figure 3.1: Distance of a robotic arm from an obstacle in a planar case

In order to find the critical point on the robot body, a geometric solution is considered.

Geometric solution

Before introduce the geometric solution that is taken into account, it is important to clarify the assumptions that are considered:

- The obstacles are modeled as a sphere in 3D case and as a circumference in the 2D case.
- The links of the robot are thought as a straight line as the length is predominant with respect to the other dimensions.

Given the above, the distance between the obstacle and p_0 can be considered as the distance between a point, i.e. the center of the obstacle, and the nearest point on a straight line, representing the links of the robot. Thus, the method is to compute the minimal distance between a point and a line.

Moreover, it must be mentioned that the position of the joints in the space is known during the computation of the inverse kinematics solution, included in the homogeneous transformation matrix. Therefore, those distances can be easily computed, but in order to find a more accurate solution, the introduced method is considered.

To compute the distance $d(P, L)$ from an arbitrary point P to a line L described by a parametric formulation, suppose that $P(b)$ is the base of the perpendicular dropped from P to L. Let the parametric line equation be defined as: $P(t) = P_0 + t(P_1–P_0)$. Then, the vector $P_0P(b)$ is the projection of the vector P_0P onto the segment P_0P_1, as shown in the figure

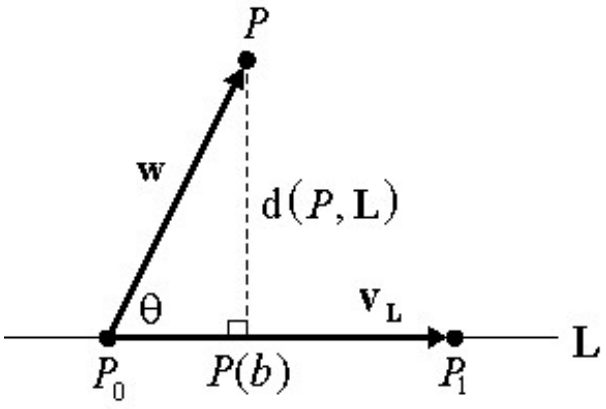

Figure 3.2: Distance between a point and a parametric infinite line

Given the above and with $v_L = (P_1–P_0)$ and $w = (P–P_0)$ it is possible to obtain

$$b = \frac{\mathrm{d}\,(P_0, P(b))}{\mathrm{d}\,(P_0, P_1)} = \frac{|w|\cos\theta}{|v_L|} = \frac{w \cdot v_L}{|v_L|^2} = \frac{w \cdot v_L}{v_L \cdot v_L} \tag{3.5}$$

and it is possible to formulate the distance as

$$\mathrm{d}(P, L) = |P - P(b)| = |w - bv_L| = |w - (w \cdot u_L)\, u_L| \tag{3.6}$$

where u_L is the unit direction vector of L.

Moreover, in our case we are facing a finite length link, so it is needed to understand if the minimal distance point is inside a finite segment or not. The two extremity are given by the position of the joints in the space, that are known. Given the previous formulation, the point is inside the links if $0 \leq t \leq 1$.

Reduced Operational Space

Following the computation of the critical point p_0, the relative distance d_0 from the center of the obstacle has to be compared to the critical distance d_m. In

the case that $d_0 < d_m$, the obstacle is "active" and a velocity has to be imposed to p_0 in order to move away the manipulator from the obstacle. The mentioned requirement is expressed in the following kinematic way:

$$J_0 \dot{q} = \dot{x}_0 \tag{3.7}$$

where J_0 is the Jacobian related to the critical point p_0. As p_0 is a point in the space, is defined as 3D vector. Thus, a Jacobian of dimension $3 \times n$ is needed in order to solve the equation (3.7).

Nevertheless, as the obstacle avoidance solution requires the motion in the direction of the line that connect the critical point to the closest point on the obstacle, i.e. d_0, it is possible to define the constraint in 1-dimensional space.

Introducing the versor n_0 as the unit vector in the direction of d_0, the Jacobian corresponding to the critical point can be written as

$$J_{d0} = n_0^T J_0 \tag{3.8}$$

where J_{d0} is a $1 \times n$ Jacobian matrix defined in the reduced operational space. In this way, from (3.7), $\dot{x}_0$ becomes a scalar and the computation becomes faster with respect to the 3D case.

Multiple obstacles

During the desired inspection task, the robot could deal with one or more obstacles. In the previous section, the formulation to deal with an obstacle has been introduced. Moreover, in the case of the presence of more than one obstacle, the previous formulation has to be extended.

Furthermore, in order to accomplish with the possible presence of multiple active obstacles, it is important to introduce the weighting factor w_i that take into account how much is active on obstacle with respect to another.

$$w_i = \frac{d_i - ||d_{o,i}||}{\sum_{i=1}^{n_0}(d_i - ||d_{o,i}||)} \tag{3.9}$$

where n_0 is the number of active obstacles, and d_i is the distance where the obstacle influences the motion of the manipulator. Thus the Inverse Kinematics equation taking into account the (3.7) definition is

$$\dot{q} = \sum_{i=1}^{n_0} w_i J_{d0i} \dot{x}_{0i} \tag{3.10}$$

Inverse Kinematics Solution

Proceeding in a similar way to the joint limit avoidance, it is possible to consider the obstacle-avoidance as the primary task T_1, while the end-effector pose becomes the secondary task T_2. The Jacobian matrices is defined for each of the desired task, respectively named J_1 and J_2. The correspondent solving equation is the following:

$$\dot{q} = J_1^{\dagger}\dot{x}_1 + N_1 J_2^{\dagger}\dot{x}_2 \tag{3.11}$$

With this formulation, if an obstacle disturbs the motion of the end-effector, the task execution has to be set to lower priority in order to deal with the first primary task. This method can be applied only in the case that end-effector trajectory accuracy is not essential or could accepts a certain error.

Furthermore, recalling that the solution depends on the active obstacles as well as on the transition between the primary and secondary tasks, it is important to guarantee a smooth transition between the two tasks. As the transition depends on the active obstacles, the term α_h is introduced depending on the distance between the critical point and the obstacle.

$$\alpha_h = \begin{cases} 1 & \text{for} \quad \|d_o\| \leq d_m \\ \frac{1}{2}\left(1 + \cos\left(\pi \frac{\|d_0\| - d_m}{d_i - d_m}\right)\right) & \text{for } d_m < \|d_o\| < d_i \\ 0 & \text{for } d_i \leq \|d_o\| \end{cases} \tag{3.12}$$

In this way, when the i-th obstacle is farther than d_i, the corresponding velocity become null. When the manipulator is approaching an obstacle, the α_h value smoothly increase until reaching the 1 value, when the critical point reaches the critical distance d_m. Thanks to this parameter, smooth velocities are obtained.

In view of the above, it is possible to define the solution including both α_h and w_i parameters. The resulting primary task T_1 is the obstacle avoidance, defined as the motion in the direction of d_0 and the motion of the end-effector the secondary task T_2. The following assumption can be made taking into account the reduced operational space:

$$J_1 = J_{d0} \tag{3.13}$$

$$J_2 = J \tag{3.14}$$

The solution of the Inverse Differential Kinematics can be written as

$$\dot{q} = \sum_{i=1}^{n_0} w_i \alpha_i J_{d0i} \dot{x}_{0i} + N_0' J^{\dagger} x_e \tag{3.15}$$

The formulation allows an unconstrained joint motion while α_h is close to zero, i.e. when all the obstacle are not active, while when the robot is close to one or

more obstacles, the null space $N_0' = N_0$ and allows the motion only in the null space of the primary task, i.e., the obstacle avoidance task.

Since the primary task is the sum of the active obstacles, the overall solution is the intersection of each individual obstacle avoidance solution. In order to define the corresponding null space, it is necessary to introduce how to determine the intersection of the null spaces. Recalling (2.4), where the null space operator is defined as $N = (I_n - J^\dagger J)$, the intersection of two null spaces N_1 and N_2 is defined as [14]

$$\ker(N_1 + N_2) = \ker(N_1) \cap \ker(N_2) \tag{3.16}$$

Thus, it is possible to describe the null space of the primary task as the sum of the null spaces and the solution of the Inverse Kinematics (3.15) can be rewritten as

$$\dot{q} = \sum_{i=1}^{n_0} w_i \alpha_i J_{d0i} \dot{x}_{0i} + \sum_{i=1}^{n_0} (w_i \alpha_i N_{0i}) J^\dagger x_e \tag{3.17}$$

where N_{0i} is the null space of the i-th active obstacle, defined as $N_0 = (I_n - J_{d0}^\dagger J_{d0})$.

3.1.3 Three tasks solution

Until now, two separate solutions for the inverse kinematics problem has been considered, one for the joint limit as primary task and the other for the obstacle avoidance as primary task.

The aim is now to find a solution imposing the joint limit as primary task, the obstacle avoidance as secondary task and the end-effector pose as third task. In order to do that, it is necessary to extend the solution given by equation (2.4) in the case of three tasks. It follows that also the formulation of the null space has to be extended into successive projection [8]. Considering $N_{prec} = (I_n - J^\dagger J)$, the successive projection is defined as

$$N_{succ} = N_{prec}(I - J_{i-1}^\dagger J_{i-1}) \tag{3.18}$$

The formulation can be extended in a systematic way for each successive projection.

Making use of the previous definition, it is now possible to express the solution of the inverse kinematics problem for our case study. Thus, exploiting the equations (3.15), (3.3), the joint velocities are computed as follow

$$\dot{q} = J_1 e_1 + (I_m - J_1^\dagger J_1) J_2 e_2 + (I - J_1^\dagger J_1)(I - J_2^\dagger J_2) J_3 e_3 \tag{3.19}$$

where the terms are given as:

- $J_1 = H = diag(h_1, ..., h_m)$ represents the joint limit activation matrix.

- $e_1 = \lambda_{JL} x_{des}$ represents the desired velocities to walk away from the joint limit.
- $J_2 = \sum_{i=1}^{n_0} w_i \alpha_i J_{d0i}$ is the sum of the obstacle avoidance Jacobian in the reduced operational space.
- $e_2 = \dot{x}_0$ is the desired velocity to escape from the critical point.
- $J_3 = J$ is the Jacobian matrix of the end-effector.
- $e_3 = x_d - x$ is the operational space error between the desired and the actual end-effector pose.

The presented formulation guarantee always the achievement of the primary task, i.e. the joint limits, while the obstacle avoidance and end-effector pose solutions are projected into the null space of the higher priority solution. Thanks to the inclusion of the transition variable, it is possible to try to accomplish with all the three tasks, guaranteeing a smooth transition between them.

On the other hand, when the majority of the joints are near their corresponding limits, it is possible to notice a deterioration of the solution for what concern the lower priority tasks. Moreover, defining the problem as in equation (3.19), the solution to the obstacle avoidance and to end-effector tasks are found in a standard way and then projected into the null space of the previous task, without taking into account of the effect that the higher priority tasks give to the solution. This could rise in a not proper contribution to the solution or non convergence to the solution by the secondary and third task.

In order to manage the mentioned problem, the exact solution formulation is taken into account.

3.1.4 Exact Solution

The exact solution is introduced in order to improve the solution of the lower priority tasks with respect to the higher priority task. It is necessary to begin from the standard formulation, as in equation (2.4), before extend it into the final one.

The equation describing the solution to the inverse kinematics problem in the exact solution is defined as [10]

$$\dot{q} = J_1^{\dagger} \dot{x}_1 + (J_2 N_1)^{\dagger} (\dot{x}_2 - J_2 J_1 \dot{x}_1) \tag{3.20}$$

The difference with the (2.4) formulation regards the definition of the secondary task. Recalling that the Jacobian is the mapping between the joint space velocities and the end-effector velocity, the term $J_2 J_1 \dot{x}_1$ represent the contribution that the primary task gives in the space of the secondary task. Thus, the secondary objective function in now defined as $(\dot{x}_2 - J_2 J_1 \dot{x}_1)$, so in the form of an error. Similarly

of the definition of the operational space error for the end-effector, $e = x_d - x_e$, also in this case the objective is to make the error null or inside a given threshold.

Now the previous formulation has to be extended in our case study, including the three desired tasks, in order to solve the inverse kinematics problem with the exact solution. In [10] the definition was done in a recursive way, obtaining the following equation equivalent to (3.19)

$$\dot{q} = J_1 x_1 + (J_2 N_1)^{\dagger}(\dot{x}_2 - J_2 J_1^{\dagger}\dot{x}_1) + (J_3 N_1 N_2)^{\dagger}(\dot{x}_3 - J_3 J_1^{\dagger}\dot{x}_1 - J_3 J_2^{\dagger}(\dot{x}_2 - J_2 J_1^{\dagger}\dot{x}_1)) \tag{3.21}$$

Taking advantage of this formulation, the highest priority tasks are no more affected by algorithmic singularities and the lower priority tasks solution is more accurate because it takes into account the contribution that the previous task produce on the solution. For example, in the secondary task, i.e. the obstacle avoidance, the term $(\dot{x}_2 - J_2 J_1^{\dagger}\dot{x}_1)$ is the difference between the desired escape velocity from the obstacle and the contribution that the joint limit primary task motion cause on the critical point. The pose, settled as third task, take into account both contribution of the two preceding tasks.

Chapter 4

Simulations

The simulation phase has been performed in the Matlab environment. Thus, together with the design of the algorithm to implement the inverse kinematics solution, it has been possible to evaluate the behavior of the robot represented by means of multiple plots. Moreover, two different simulations have been covered. The first one is related to a configuration of the robot that is used only during the simulation phase, while the second one is referred to the subsequent implementation on the robot prototype available in the laboratory.

4.1 Simulation

The first type of simulation regards the employment of the solution to the inverse kinematics problem found in the previous chapter on a planar hyper-redundant robot for inspection purposes.

Before proceeding with the results of the simulation, in the next sections is introduced the configuration of the robot used during the simulation and the working environment chosen in order to perform the inspection task.

4.1.1 Robot Configuration

The configuration of the robot used in this simulation is a planar hyper-redundant robot. The number of joints is 13 and are all revolute. The robot is fixed to the base frame by means of a revolute joint as well, that allows to unroll it during the execution of the motion. Moreover, the robot links are considered all of the same length, that is $0{,}5m$.

With exception to the central base joint, which does not have a joint limit, the other joint limits are settled to $60° = 1{,}0472$ rad. The joint limit property are resumed in the following table

n° joint	type [R|P]	initial angle [rad]	limit [rad]
1	R	-π	none
2	R	-$\pi/3$	$\pm\pi/3$
3	R	-$\pi/3$	$\pm\pi/3$
4	R	-$\pi/3.5$	$\pm\pi/3$
5	R	-$\pi/4.5$	$\pm\pi/3$
6	R	-$\pi/5$	$\pm\pi/3$
7	R	-$\pi/5$	$\pm\pi/3$
8	R	-$\pi/5$	$\pm\pi/3$
9	R	-$\pi/5.5$	$\pm\pi/3$
10	R	-$\pi/5.5$	$\pm\pi/3$
11	R	-$\pi/5.5$	$\pm\pi/3$
12	R	-$\pi/5.5$	$\pm\pi/3$
13	R	-$\pi/7.5$	$\pm\pi/3$

Table 4.1: The joint limit table. The type is R for revolute and P for prismatic, initial angle and limit

4.1.2 Toroidal Environment

In the introduction, several types of confined spaces difficult to be reached by humans has been mentioned, such as vessels, pipes, combustion chambers or similar structures.

Recalling the aim of the thesis, i.e. the inverse kinematics solution for an hyper-redundant robot for inspection purposes, the choice between the possible environments to be inspected is a toroidal environment. An example is shown in figure 4.1.

The challenge is that the annular region has large dimension meanwhile the space to be inspected and where the robot should go through is restricted.

In order to illustrate the working environment for the simulation, one possibility is given in figure 4.2, where the starting configuration and the final configuration are represented. Regarding the initial configuration, the robot is thought to be coiled around a central base, that allow the robot to unwind during the task execution.

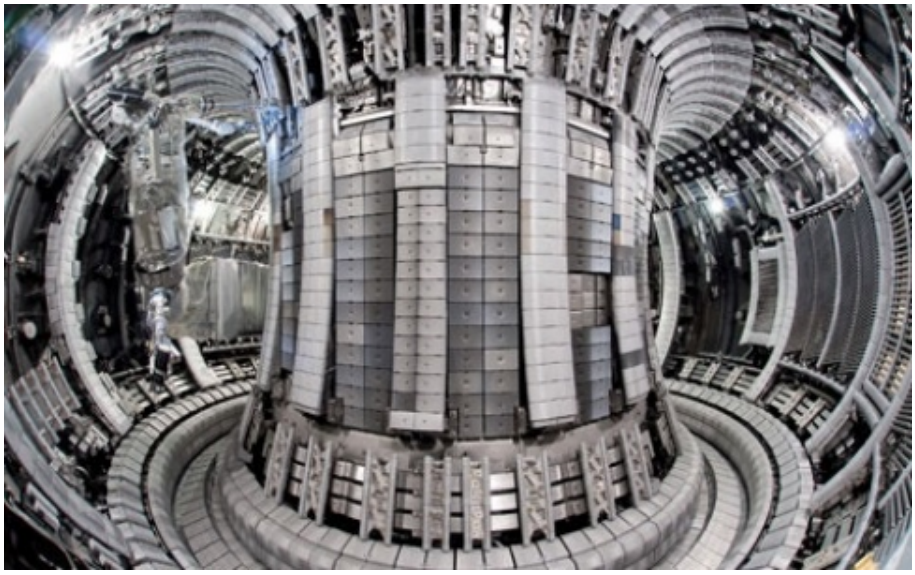

Figure 4.1: An example of torus environment in a nuclear fusion power plant.

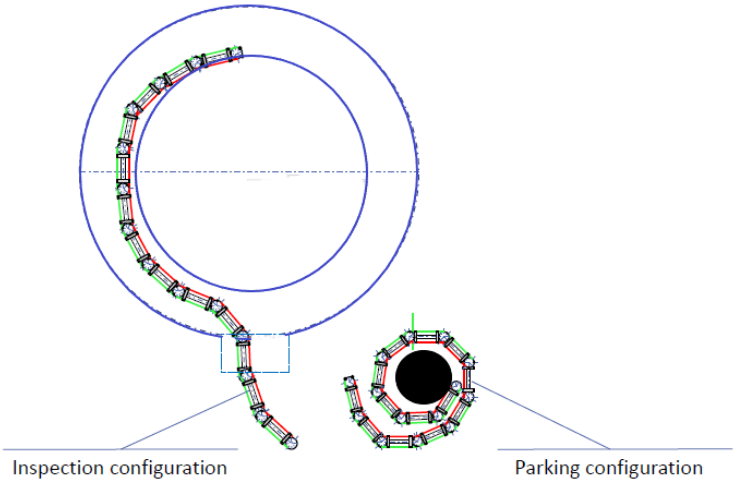

Figure 4.2: Representation of the desired working environment to be inspected by means of planar hyper-redundant robot

4.1.3 Obstacle Modelling

The desired working environment is represented in figure 4.2. The aim is now to built a similar ambient for the simulation in Matlab.

The robot is fixed on a rotating base that is handled as an obstacle, in such a way that the robot, while trying to follow the desired pose, tends to unroll itself.

Moreover, in order to define the entrance area, two smaller obstacles are located at the same eight in proximity of the desired zone, so the robot is allowed only to go through that space.

Similarly, the area of the toroidal environment where the robot should execute the motion can be represented as two obstacles. Taking into account the method seen in chapter 3 for the obstacle avoidance, when the end-effector is approaching the entrance zone two problems arise: when the manipulator is close to an obstacle, an escape velocity is assigned in order to walk away from that point. Thus, the external obstacle cannot be modeled in the mentioned way, because it would not allow to enter inside the inspection area. To solve this problem, an entrance zone is defined. In this area, the problem of the obstacle avoidance with reference to the external obstacle is not considered and the robot is allowed to enter inside the chamber.

The second problem arise when the manipulator is inside the two obstacles, after crossing the entrance zone. Once again, if the external obstacle is considered as the others, the obstacle avoidance task tends to make the robot go away from the obstacle, and so the manipulator try to exit from the toroid structure. The solution of this problem is to consider the external obstacle as attractive instead of repulsive. In this way, the motion in the internal area between the two obstacles is allowed, taking into account the obstacle avoidance problem at the same time.

A structure representing the properties of the obstacle is given in the following table

n° obstacle	type [R|A]	center [m]	radius [m]	$d_i[m]$	$d_m[m]$
1	R	[0 0 0]	0.4	0.1	0.05
2	R	[-1.25 0 1.4]	0.15	0.1	0.05
3	R	[-0.6 0 1.4]	0.15	0.1	0.05
4	R	[-0.9 0 3]	0.95	0.1	0.05
5	A	[-0.9 0 3]	1.45	0.1	0.05

Table 4.2: The obstacle parameters table. The type R is for repulsive obstacle and A for attractive obstacle. The critical distance d_m and d_i are defined for each obstacle.

4.1.4 Cubic Spline

Inverse kinematics is a mapping between the desired position and orientation of the end-effector and the corresponding joint configuration that allows that pose.

The set of point that the manipulator should follow during the motion execution defined in the operational space is named path. Moreover, the trajectory represent the path which a specified timing law. In order to define the motion of the robot trough the desired trajectory in the operational space, different techniques can be used. In this work of thesis the spline method is used.

A spline is a piecewise mathematical function defined by means of polynomials of possible different orders and it is the solution with minimum curvature among all interpolating functions having continuous second derivative. In this work of thesis the cubic spline are taken into account.

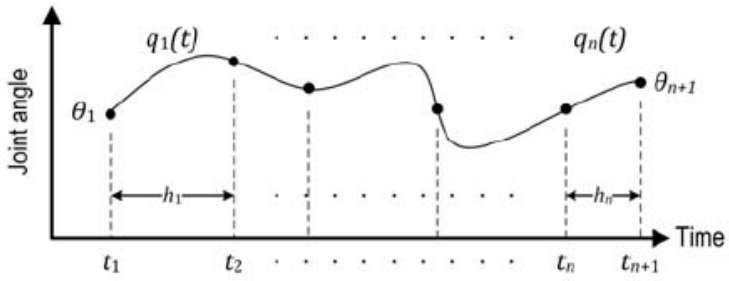

Figure 4.3: Example of trajectory defined by means of cubic spline

Cubic polynomials allow to define the continuity for the initial and final position and velocity, as 4 coefficients could be settled. In order to obtain the continuity also for acceleration, a quintic order polynomials should be used.

On the other hand, making use of the cubic spline allows to interpolate n points guaranteeing the continuity until second derivative, i.e. acceleration, using the third order polynomials. The methods is based on the use of two virtual points, that give rise to two additional free parameter to impose the continuity on the acceleration [11].

4.1.5 Simulation Results

The simulation of the 13 joints planar hyper-redundant robot executing the inspection trajectory inside a toroidal environment on Matlab give rise to the following sequence of figures 4.4-4.11. As mentioned before, the red motion trajectory is defined by means of cubic spline.

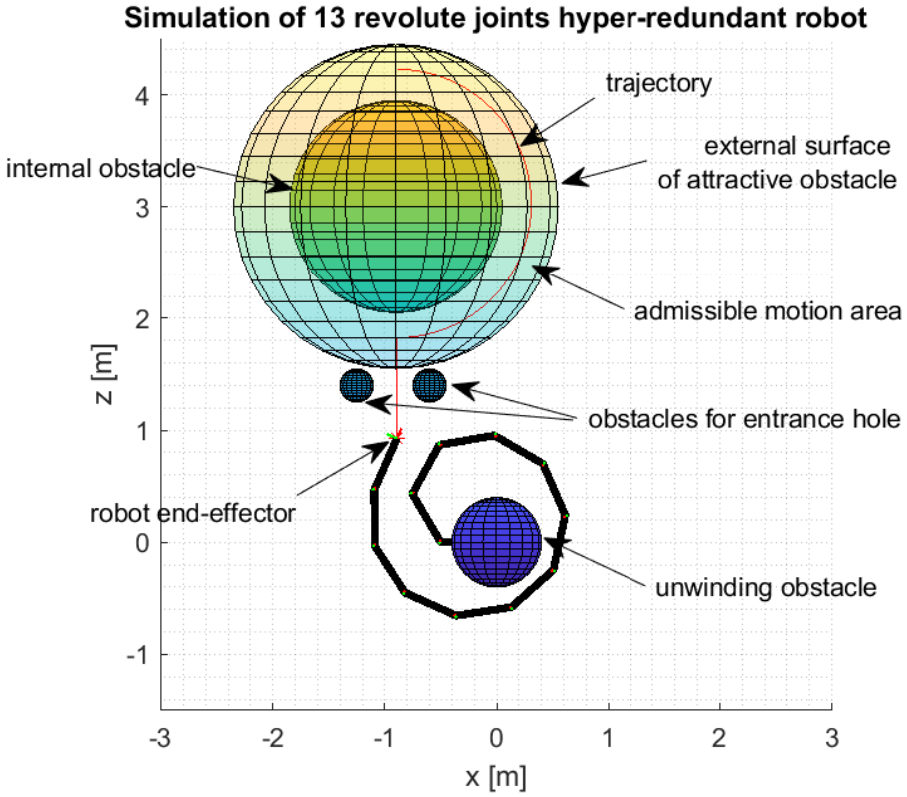

Figure 4.4: The starting configuration of the hyper redundant robot during the toroidal environment inspection.

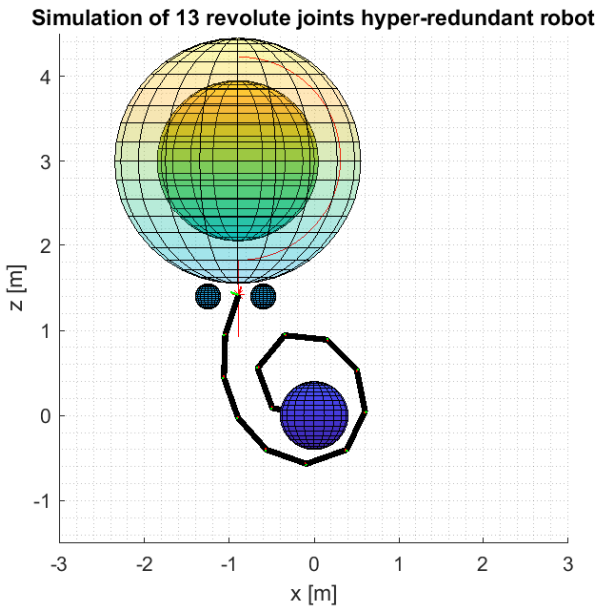

Figure 4.5: The toroidal environment inspection simulation. The robot approaches to entrance area.

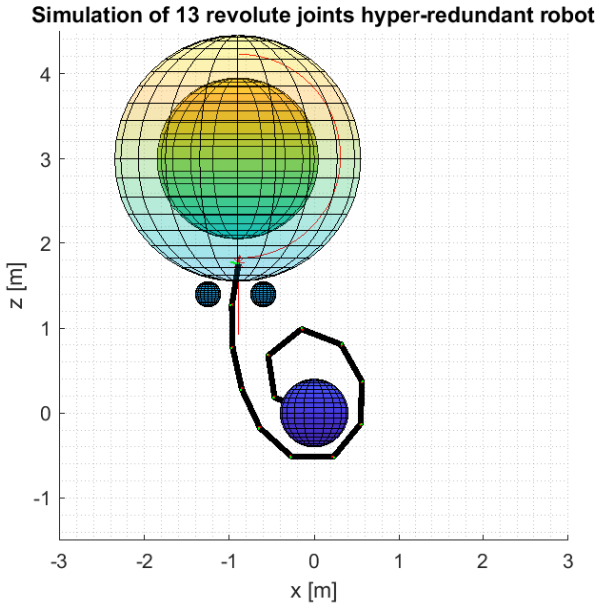

Figure 4.6: The robot entrance in the toroidal environment inspection simulation.

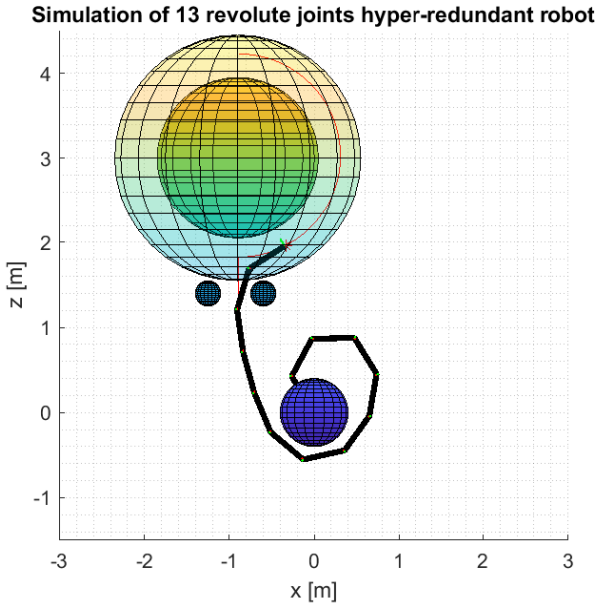

Figure 4.7: The robot obstacle avoidance while trying to follow the inspection trajectory.

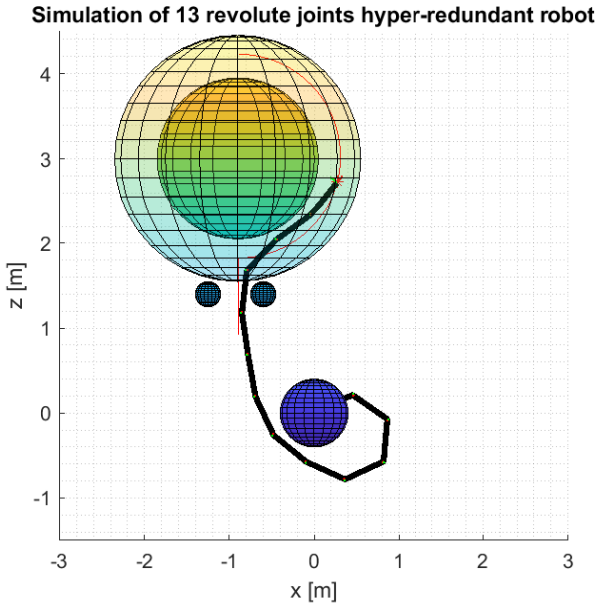

Figure 4.8: The robot obstacle avoidance while trying to follow the inspection trajectory in the toroidal environment.

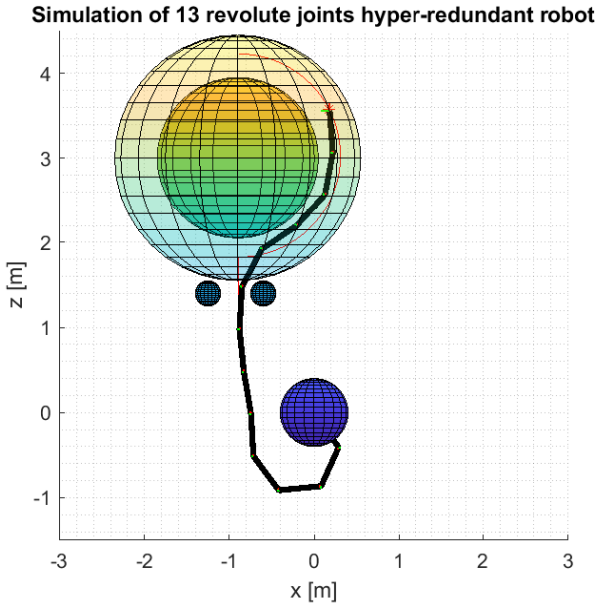

Figure 4.9: The robot obstacle avoidance while trying to follow the inspection trajectory.

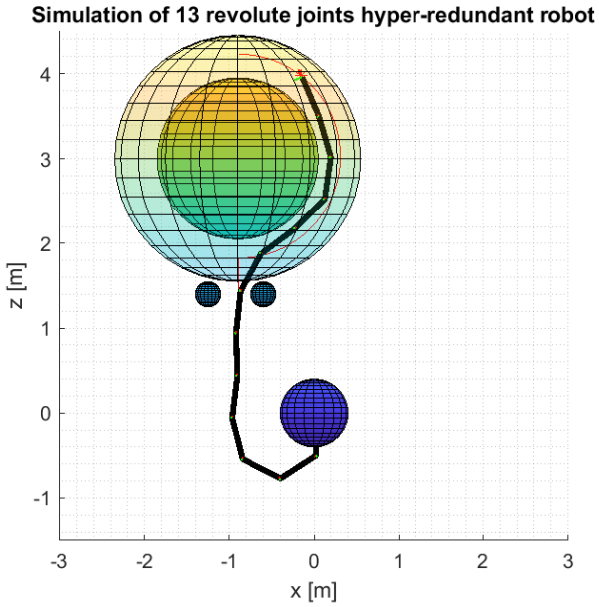

Figure 4.10: The robot obstacle avoidance while trying to follow the inspection trajectory in the toroidal environment.

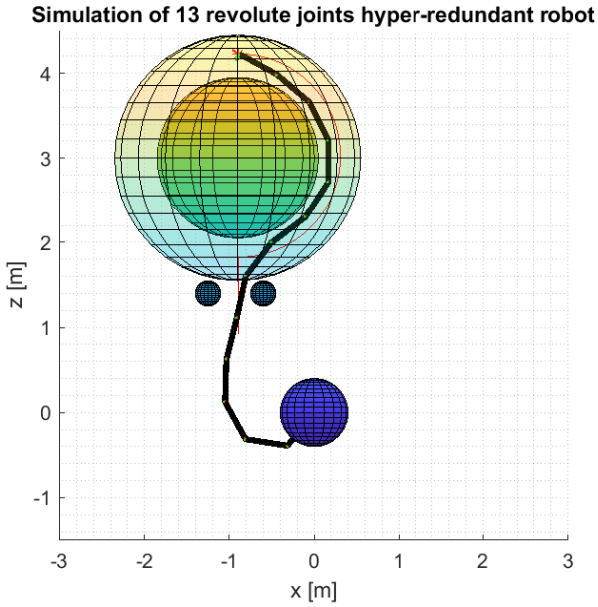

Figure 4.11: The robot final configuration during the simulation of a toroidal environment inspection trajectory.

Taking into account the sequence of figures 4.4-4.11, it has been proved that the solution given in equation (3.21) provides the desired accomplishment of the desired inspection trajectory, while obstacle are avoided and joint limits are respected.

Moreover, in order to show that the desired tasks are effectively respected, the behavior of the joint limits, obstacle distance and pose of the end-effector has been plotted, related to the performed simulation in Matlab.

The pose error of the end-effector is computed as the absolute value of the error vector as shown in figure 4.12. The x-axis represent the point on the trajectory, computed with a given sampling time. The magnitude of the error is represented in meters, thus the maximum error is about 0.035 meters. Recalling that the pose of the end-effector has a third priority in the solution of the inverse kinematics, 3,5 cm is an acceptable error founding the solution.

In the table 4.1 it is represented each joint limit angle in degrees. The corresponding value in radians is 1.0472. The joint limit task is settled as primary task, thus it is expected that the threshold chosen is always respected.

Moreover, it is important to notice that the second and third joint are initialized near their limit, while the first base link has no bound.

In the figure 4.13 the y-axis represent the magnitude of the joint angles in radians, while the x-axis is, as in the case of the end-effector pose error, the trajectory points. Thus, during the execution of the inspection trajectory, all the

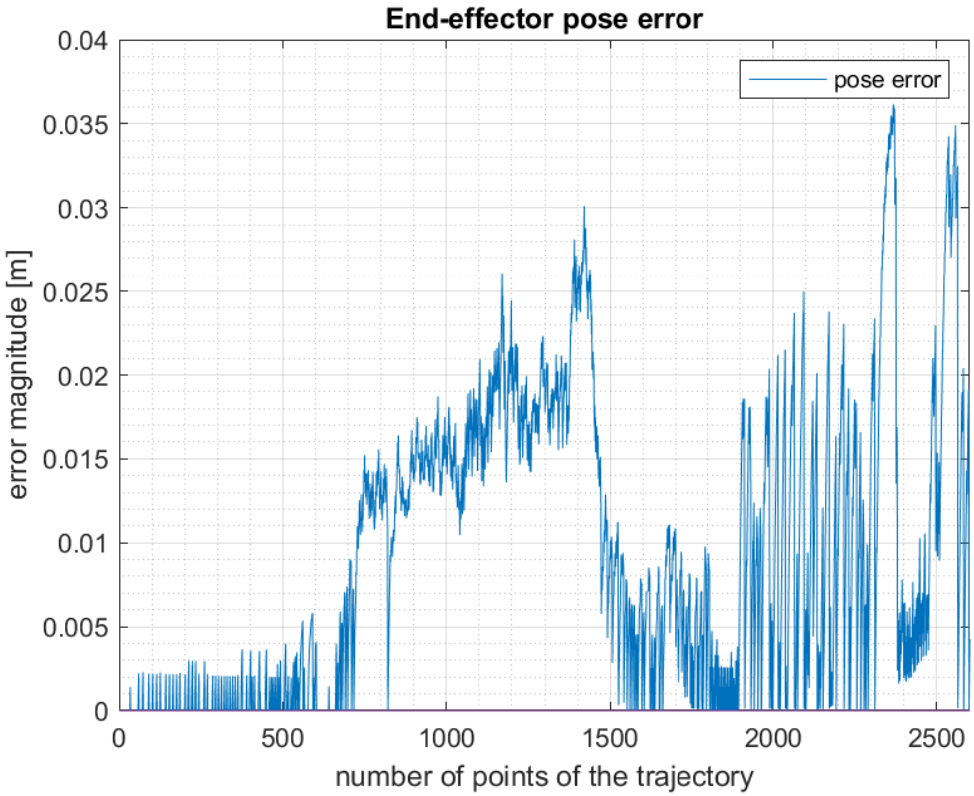

Figure 4.12: The end-effector pose error magnitude behaviour

joints are inside the [-1 1] range, showing the effectiveness of the inverse kinematics solution implemented. The unique joint that is outside the limit is the first one as expected.

Obstacle parameters has been introduced in table 4.2. In this simulation the critical distance d_m and the influencing distance d_i are assigned equal for all the obstacles. Recalling that the obstacle avoidance is established as secondary task, the behavior during the execution of the desired trajectory is shown in the following figures.

In figure 4.14 is represented the distance in meters of the nearest point to the obstacle, i.e. the critical point p_0. The obstacle 1 is the one fixed on the base of the robot and, as mentioned before, allows the robot to unroll. As it can be evaluate from the 4.14, the minimum value of the distance is about 0.1, thus the obstacle avoidance is respected during the whole simulation. The arise of value after the value 1750 on the x-axis is due to the consideration that the first 3 joints are not taken into account in this obstacle avoidance, as they are imposed near their corresponding joint limit and as a consequence they cannot hit the obstacle.

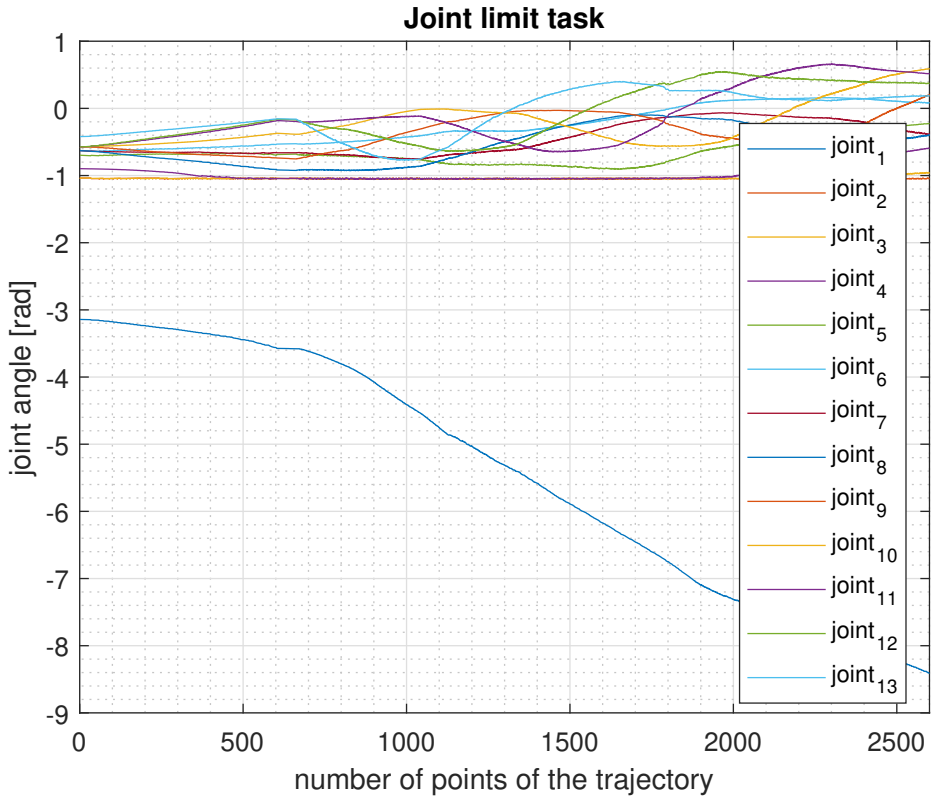

Figure 4.13: The angles of the joint during the simulation, respecting the limits

Figure 4.14 represent the behavior of the distance from the obstacle 2, that is the left side of the entrance zone of the annular region. As the simulation is referred to the inspection of the right side, this distance is always respected. In fact the minimum distance is 0.15 meters.

The obstacle 3 is the right side of the entrance zone and with respect to left side, is always active as the robot tends to hit the obstacle while is performing the inspection trajectory. The distance from the obstacle 3 is shown in figure 4.14. As it can be evaluated, the obstacle avoidance task, after the robot entrance in the toroidal environment, is always active. Moreover, at a certain point, about 1750 value in abscissa axis, the distance is lower than the critical distance, reaching a value of 0.025 meters.

Recalling that the obstacle avoidance is settled as secondary task, thus the joint limit primary task has major priority, obstacle avoidance is always respected and the environment is not damaged as desired.

In the end, in figure 4.14 the toroidal environment internal and external sides are plotted, represented by obstacle 4 and obstacle 5. The aim is to respect

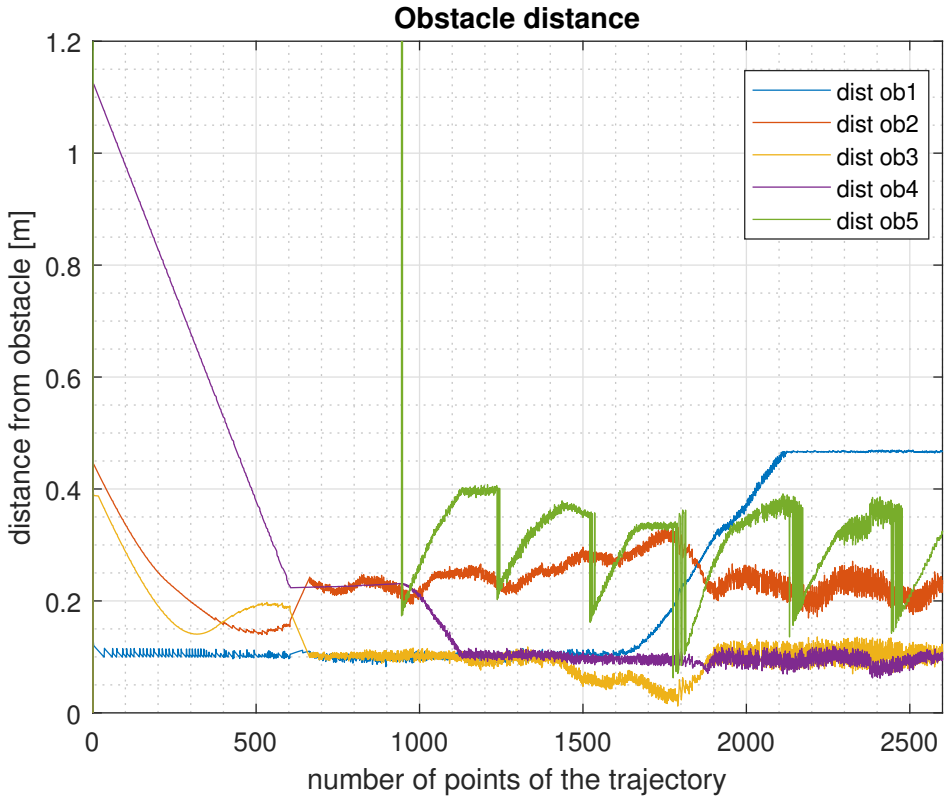

Figure 4.14: The obstacle 1 distance during the execution of the inspection trajectory

the distances while the robot executing the motion. As it can be seen from the mentioned figures, during the simulation the distances are always respected.

Summarizing in this simulation it has shown the application of the inverse kinematics solution given in equation (3.21). The results has been the satisfaction of all the desired tasks together with the execution of the desired inspection trajectory, pointing out the efficiency of the algorithm.

Chapter 5

Experiments

In this chapter it is covered the experiments performed on the planar hyper-redundant robot prototype, testing the efficiency of the inverse kinematics algorithmic solution.

However, the working environment where to perform the experiment is not the one create on Matlab during the simulation, i.e. a toroidal environment. Moreover, the manipulator prototype as well is not the same as the 13 joints planar hyper-redundant robot introduced in chapter 4.

The reason is that due to Covid-19 closure of the laboratory, the team has not been able to finish the design of the final version of the planar robot. Hence, the experiments has been performed on a 5 joints planar redundant robot.

In order to execute the experiment, a new simulation phase has been covered, in such a way to define a feasible trajectory to execute the motion of the manipulator in the new working environment and to fix the position of the obstacles.

The planar robot prototype is composed by 6 revolute joints, the link length is 0.5 meters, as shown in figure 5.1.

The parameters defined in order to perform the simulation and the experiments are the following

- initial joint angles q0=[0.790;0.786;0.787;0.786;0.788];
- the obstacle center position is in [0.75 0 0.4]' with radius 0.225 meters, $d_m = 0.05$, $d_i = 0.1$;
- the ceiling is considered as an obstacle;
- the joint limits are defined in radians as [0.9 1.3 0.95 1.1 1.3];
- the points defining the trajectory are $point_x$=[p_0(1,end) -1.25 0.3 2.1 1.5 1] and $point_z$=[p_0(3,end) 1.2 2 1.1 0.1 0.1]';

Figure 5.1: The 6 revolute joints robot prototype used for experiments

The experiments are performed by using Labview software, a system-design platform and development environment for a visual programming language from National Instruments.

5.1 Simulation for experiments

In order to perform the experiments on the robot prototype, a simulation on the working environment has been performed, with the objective of finding a feasible motion trajectory that the robot should execute. The exact solution algorithm is the one used to solve the inverse kinematics problem given by equation (3.21).

The trajectory is defined by means of cubic spline, interpolating the set of points defined in the previous section. Together with the obstacle placed on the right side of the manipulator, also the ceiling has been introduced as obstacle, as can be seen from figure 5.2.

The inspection trajectory executed during the simulation is represented in the figure sequence 5.2 5.3. The robot respects the joint limit and avoids the obstacle while is trying to follow the desired end-effector pose.

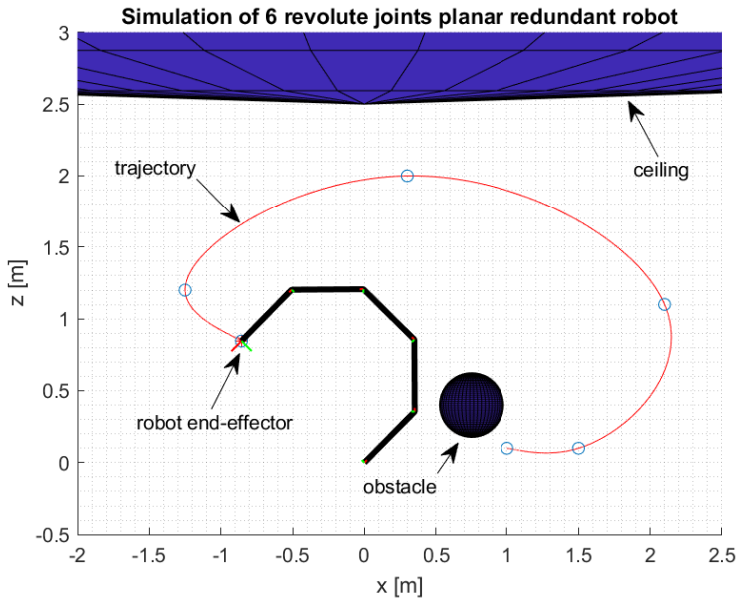

Figure 5.2: The 6 joints planar robot prototype simulation while performing an inspection trajectory. The figure represents the starting configuration.

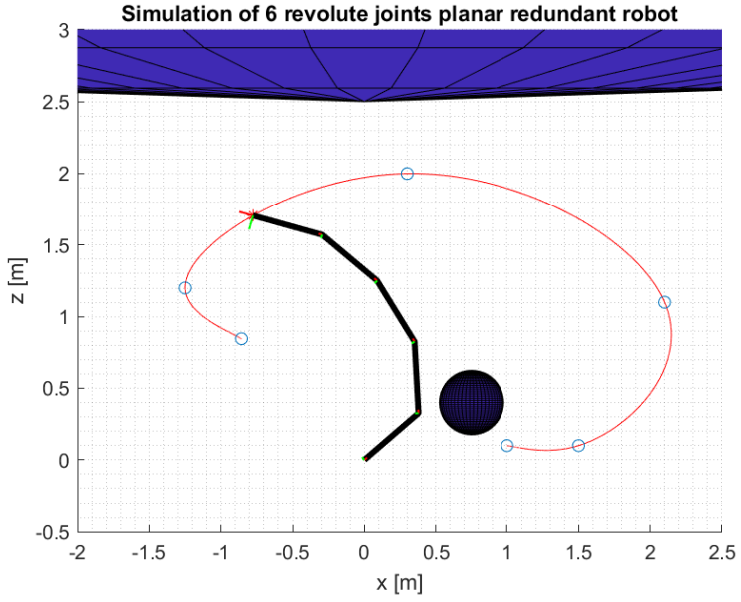

Figure 5.3: The 6 joints planar robot prototype simulation while performing an inspection trajectory. The figure represents an intermediate configuration.

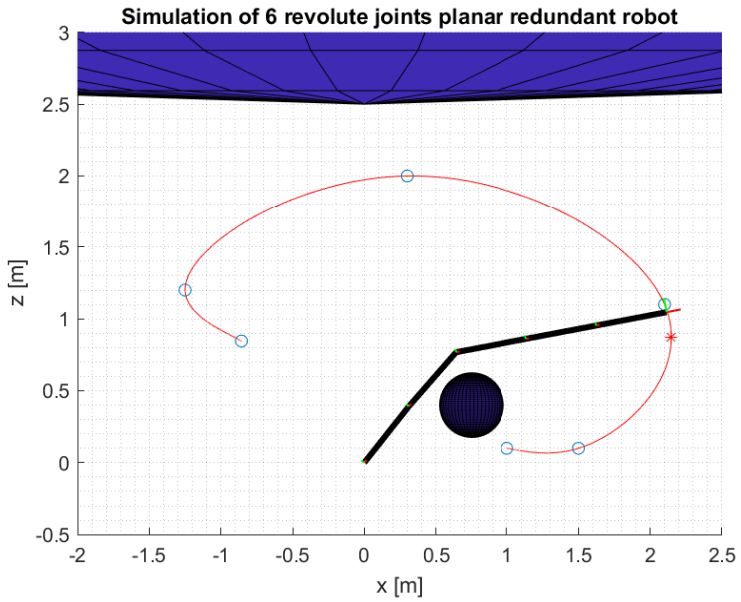

Figure 5.4: The 6 joints planar robot prototype simulation while performing an inspection trajectory. The figure represents the end-effector maximum amplitude pose error.

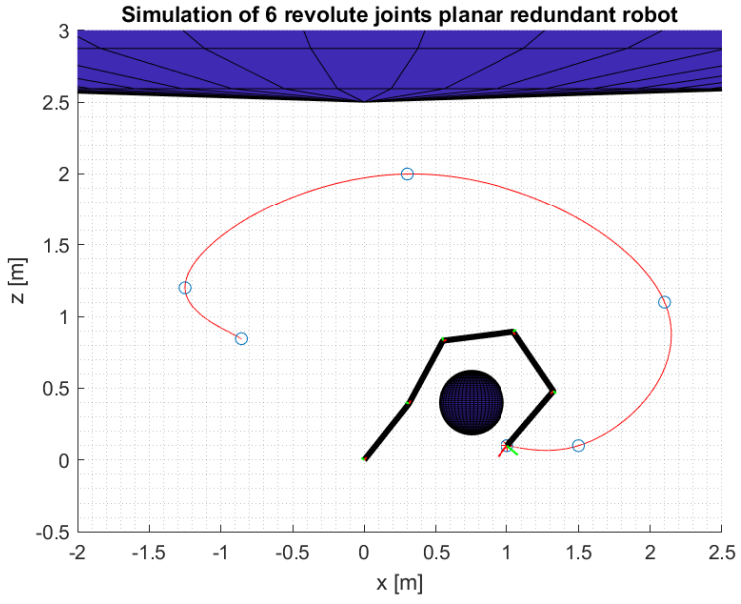

Figure 5.5: The 6 joints planar robot prototype simulation while performing an inspection trajectory. The figure represents the final configuration.

In figure 5.6 is represented the end-effector pose error during the execution of the motion. This is the third priority task, thus following the trajectory an error can be accepted. Moreover the final error is null, while the maximum magnitude is 0.19 meters in corrispondence of the 6000 value in the x-axis.

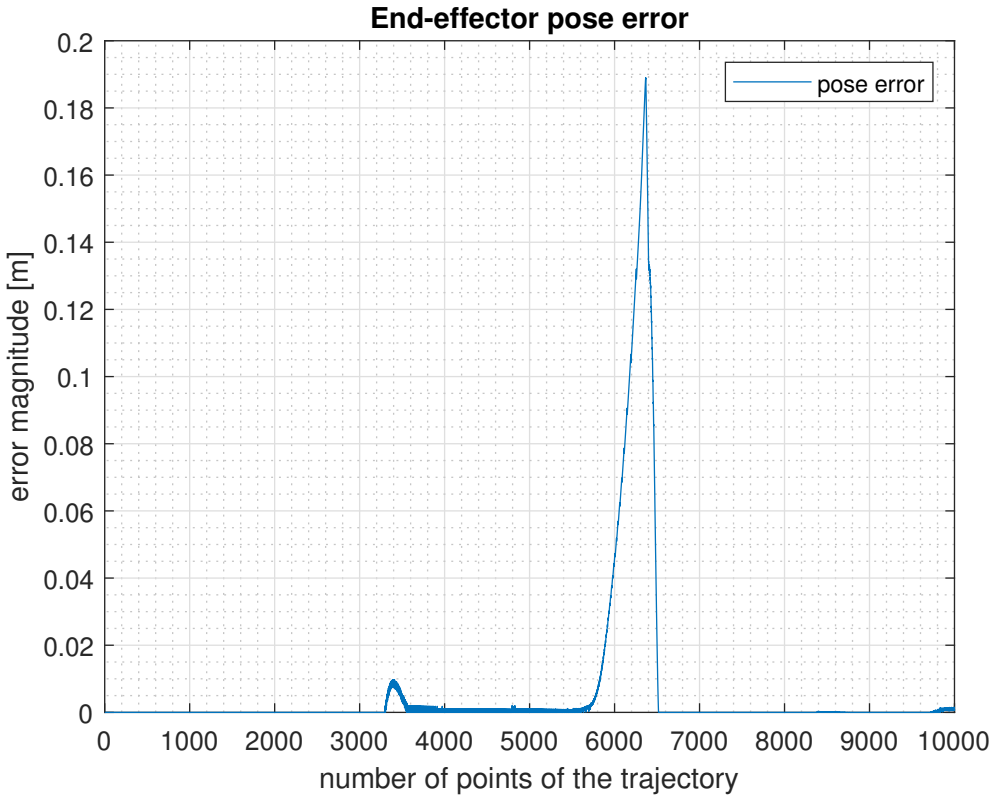

Figure 5.6: The end-effector pose error obtained during the execution of the inspection trajectory simulation of the 6 joints planar robot prototype

The angles assumed by the joint during the execution of the inspection trajectory are in figure 5.7. The x-axis represent the number of point in the trajectory, while the y-axis represent the joint angles in radians.

The limits has been defined in the previous section and as it can be seen from the figure, they are all respected. For example the purple line is for the joint with 1.1 radians limit; when it reaches this value, thanks to the inverse kinematics solution it does not cross the threshold.

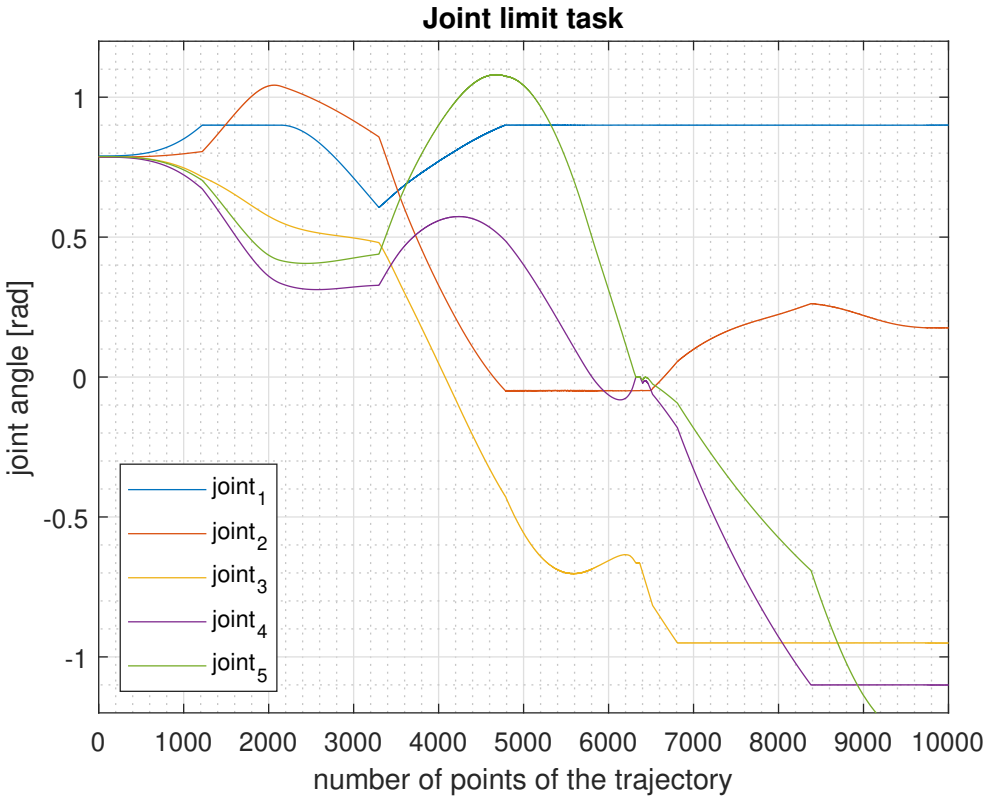

Figure 5.7: The angles of the joint assumed during the execution of the inspection trajectory simulation of the 6 joints planar robot prototype

In the end the distances between the critical points and the obstacles are represented in figure 5.8. The the vertical axis is the distance in meters while the horizontal axis is for the point number in the trajectory. The red line is the distance from the ceiling while the blu line is the distance from the obstacle. The defined threshold d_i is respected during the execution of the whole trajectory.

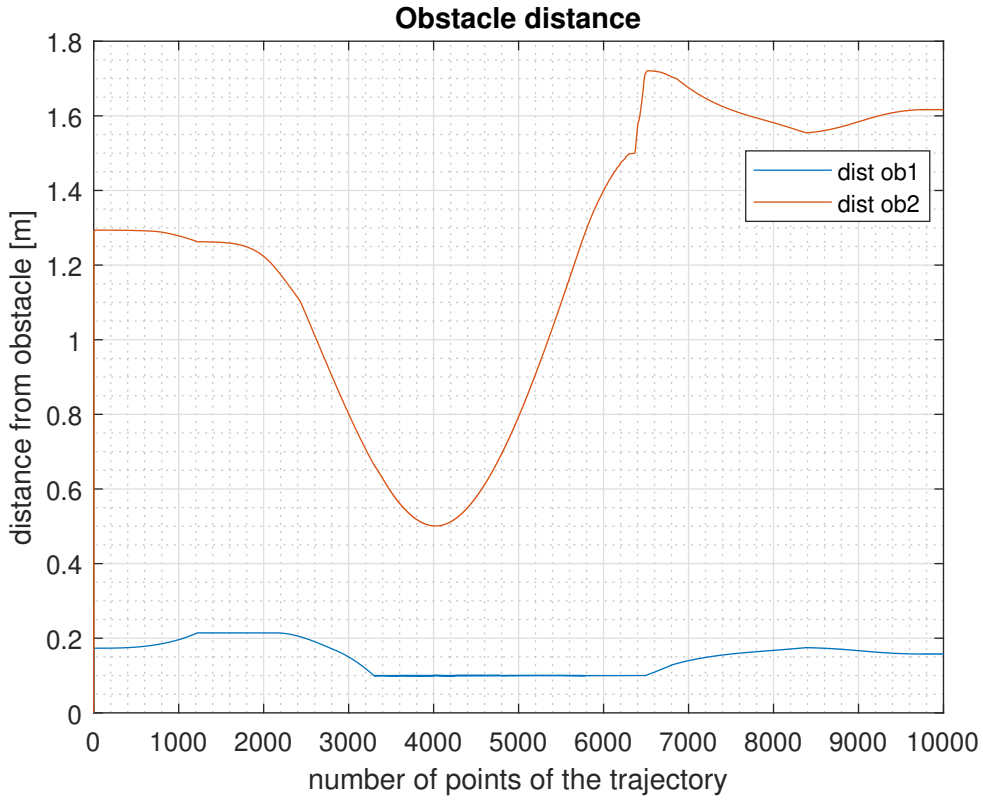

Figure 5.8: The distance of the robot from the obstacles during the execution of the inspection trajectory simulation of the 6 joints planar robot prototype

5.2 Experiments

As mentioned before, the experimental part of this work of thesis has been performed in Labview environment.

In order to understand the efficiency of the algorithmic solution for the inverse kinematics problem developed for an hyper-redundant robot, a comparison between the simulation and experiment has been done.

The parameters for the execution of the experiments are the same defined for the simulation in the first part of this chapter. The robot configuration has been shown in figure 5.1, thus a 6 revolute joints redundant manipulator prototype. Moreover, during the performance of the tests, the base revolute joint is not moved, for a physical limit of the available prototype.

In the control panel in figure 5.9 created in Labview, the trajectory is defined by setting the point for the cubic spline. Moreover all the parameters are defined, starting from the obstacle position and radius (the blue circumference), as well as

the other parameters for joint limit.

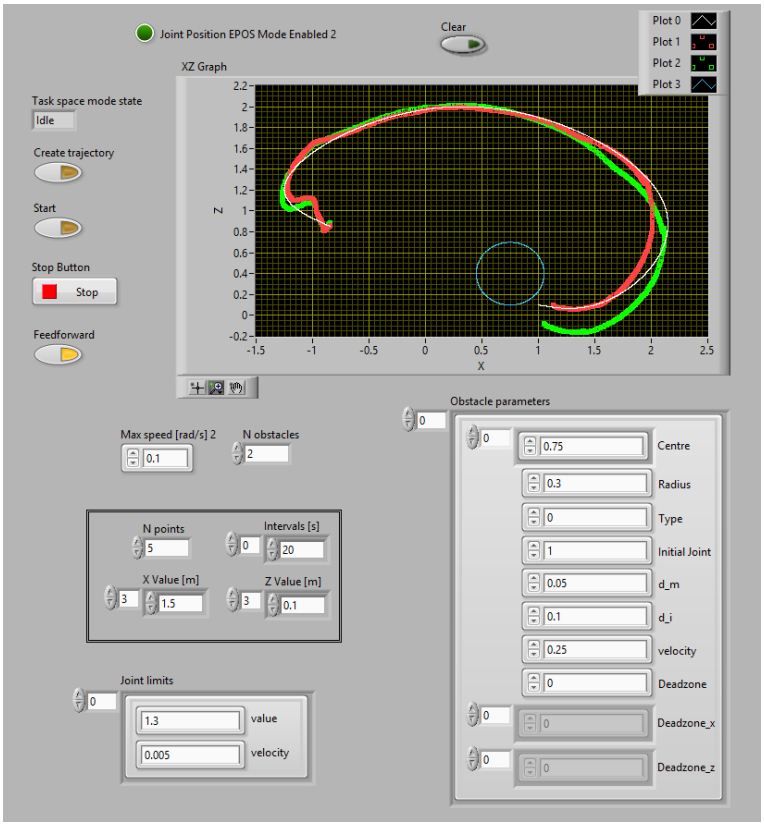

Figure 5.9: The labview control panel created for the parameters definition and trajectory tracking.

Moreover, the encoders are mounted in correspondence of each joint. The encoder is an electro-mechanical device that converts the angular position or motion of a shaft or axle to analog or digital output signals. Taking advantage of this devices, it is possible to track the real angle that the motor perform in the corresponding joint. The behavior of the engines is represented in figure 5.10. The two lines represent the difference between the desired behavior of the engine and the corresponding real behavior tracked by means of the encoders. Only the motor 2 has a not desired shape while the others behave as desired.

The experiments has confirmed the efficiency of the solution to the inverse kinematics algorithm given by equation (3.21). All the joint limits has been respected, the obstacle avoided and the inspection trajectory has been successfully performed, as can be seen from figure 5.11.

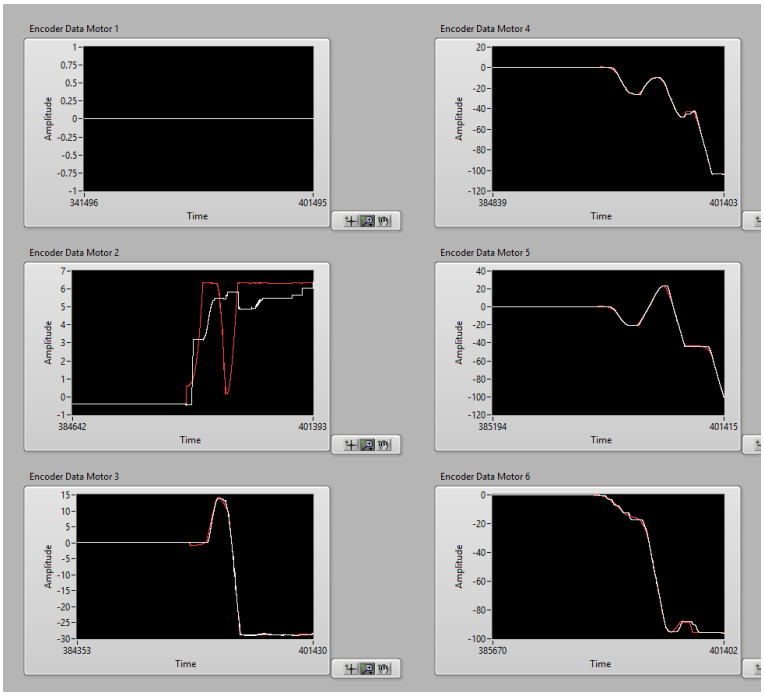

Figure 5.10: The behavior of the engines during the execution of the inspection trajectory. The comparison between real behavior, tracked by means of encoders, and the desired one.

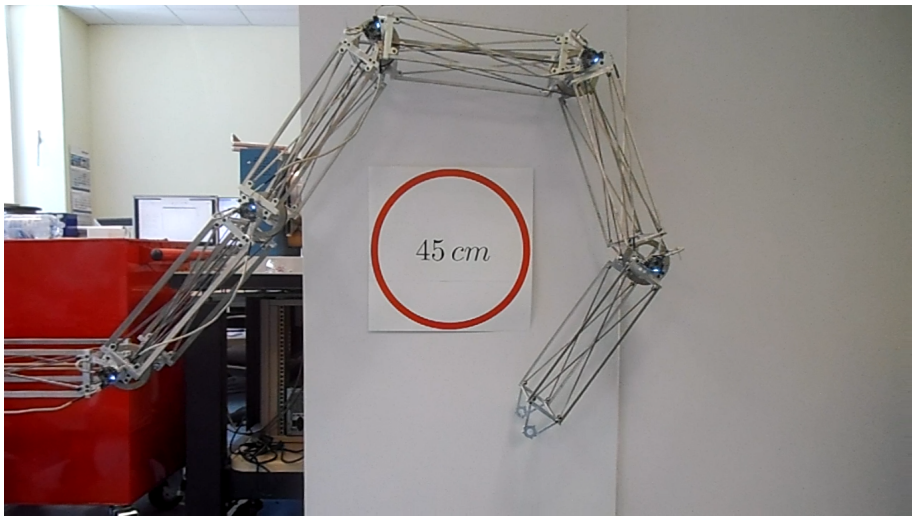

Figure 5.11: The final configuration of the 6 joints planar redundant robot prototype after the execution of the inspection trajectory.

Chapter 6

Conclusion

In this work of thesis has been covered the study of the inverse kinematics problem for an hyper-redundant robot for inspection purposes. The objective has been to find a solution to the inverse kinematics problem in order to satisfy multiple tasks taking advantage of the high degrees of redundancy. Furthermore, the tasks has been defined with a different priority:

- primary task: joint limit
- secondary task: obstacle avoidance
- third task: end-effector pose

An algorithm has been developed, that includes the three desired tasks together in the solution, taking into account the priority mentioned above. The exact solution has been used in such a way to take into account the contribution to the solution of the other tasks, while computing the solution of each task. During the execution of the inspection trajectory, a smooth transition has been ensured between the active tasks, guaranteeing the achievement of the desired tasks.

The simulation of a 13 joints planar hyper-redundant robot executing an inspection trajectory inside on a toroidal environment has been performed on Matlab software. The aim has been the execution of the desired trajectory while avoiding the obstacles and respecting the joint limit. The result of the simulation has been the satisfaction of all the desired tasks together with the execution of the desired inspection trajectory, pointing out the efficiency of the algorithm concerning the solution of the inverse kinematics problem.

After the simulation phase, the algorithm has been implemented on a 6 revolute joints planar redundant robot prototype available at AIAL laboratory at IIT. The robot has been able to perform the desired trajectory, while avoiding the joint limit and the obstacles. The experiments performed successfully on the robot has shown the effectiveness of the developed algorithm.

The work presented in this thesis could be extended or improved in the following ways:

- The developed algorithm could be tested on the final robot configuration, trying to accomplish with more complex inspection trajectory, e.g. in the toroidal environment.
- The algorithm could be tested on a 3D configuration inspection robot.
- The solution to the inverse kinematics could be extended integrating additional method, as the follow the leader solution or others custom solution.
- A different end-effector could be considered, instead of an inspection camera, in order to try to accomplish with others applications. For example a robotic arm in such a way to perform maintenance operations.

Bibliography

[1] Hariharan Ananthanarayanan and Raul Ordonez. "Real-time Inverse Kinematics of (2n + 1) DOF hyper-redundant manipulator arm via a combined numerical and analytical approach". In: *Mechanism and Machine Theory* 91 (Sept. 2015).

[2] Samuel Buss. "Introduction to inverse kinematics with Jacobian transpose, pseudoinverse and damped least squares methods". In: *IEEE Transactions in Robotics and Automation* 17 (May 2004).

[3] Samuel R Buss. "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods". In: *IEEE Journal of Robotics and Automation* 17.1-19 (2004), p. 16.

[4] Pasquale Chiacchio et al. "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy". In: *The International Journal of Robotics Research* 10.4 (1991), pp. 410–425.

[5] S. Chiaverini, B. Siciliano, and O. Egeland. "Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator". In: *IEEE Transactions on Control Systems Technology* 2.2 (1994), pp. 123–134.

[6] Adrià Colomé and Carme Torras. *Reinforcement Learning of Bimanual Robot Skills*. Springer, 2019.

[7] A. S. Deo and I. D. Walker. "Robot subtask performance with singularity robustness using optimal damped least-squares". In: *Proceedings 1992 IEEE International Conference on Robotics and Automation*. 1992, 434–441 vol.1.

[8] Alexander Dietrich, Christian Ott, and Alin Albu-Schäffer. "An overview of null space projections for redundant, torque-controlled robots". In: *The International Journal of Robotics Research* 34.11 (2015), pp. 1385–1400.

[9] Yu Liu, Yanshu Jiang, and Linlin Li. "Multi-objective performance optimization of redundant robots using differential evolution". In: *Proceedings of 2011 6th International Forum on Strategic Technology*. Vol. 1. IEEE. 2011, pp. 410–414.

[10] De Luca. "Kinematic Redundancy". University Lecture. 2020.

[11] De Luca. "Trajectory Planning". University Lecture. 2020.

[12] Nicolas Mansard, Oussama Khatib, and Abderrahmane Kheddar. "A unified approach to integrate unilateral constraints in the stack of tasks". In: *IEEE Transactions on Robotics* 25.3 (2009), pp. 670–685.

[13] T Petrič et al. "Obstacle avoidance with industrial robots". In: *Motion and Operation Planning of Robotic Systems*. Springer, 2015, pp. 113–145.

[14] R. Piziak, P.L. Odell, and R. Hahn. "Constructing projections on sums and intersections". In: *Computers Mathematics with Applications* 37.1 (1999), pp. 67–74.

[15] Lorenzo Sciavicco and Bruno Siciliano. "A solution algorithm to the inverse kinematic problem for redundant manipulators". In: *IEEE Journal on Robotics and Automation* 4.4 (1988), pp. 403–410.

[16] Bruno Siciliano et al. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[17] Haibo Xie et al. "A geometric approach for follow-the-leader motion of serpentine manipulator". In: *International Journal of Advanced Robotic Systems* 16 (Sept. 2019), p. 172988141987463.

[18] Leon Žlajpah and Tadej Petrič. "Obstacle avoidance for redundant manipulators as control problem". In: *Serial and Parallel Robot Manipulators-Kinematics, Dynamics, Control and Optimization* (2012), pp. 203–230.