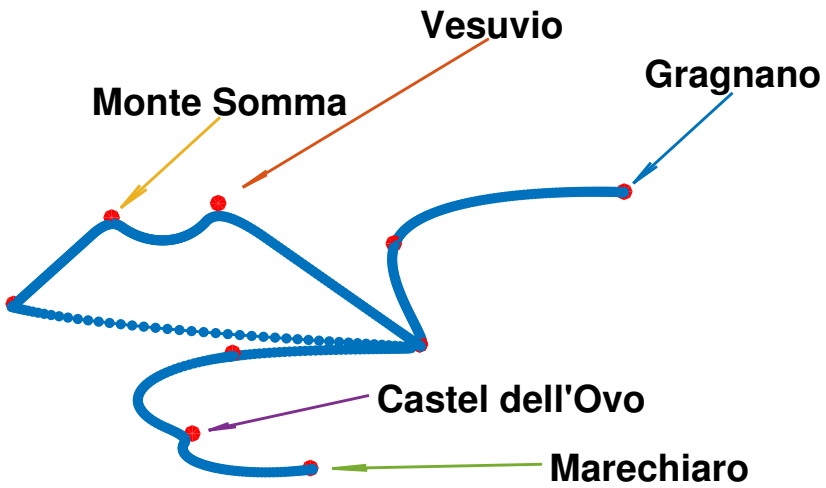


FROM GRAGNANO TO MARECHIARO
A Wonderful Trip To Robot Control ... And Beyond!

GENNARO NOTOMISTA†

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The main objective of this technical report is to answer the questions required for the exam of Robot Control (9 CFU) held by Prof. Bruno Siciliano and belonging to the Master's Course in Automation Engineering.

The report starts from robot kinematics and arrives to advanced techniques of motion planning, passing through motion and force control.

The topics that are presented in this report are: robot kinematics, manipulability, path planning, manipulator redundancy, inverse kinematics, robot dynamics, motion control, robust control, adaptive control, force control, impedance control, force/position control, motion planning via artificial potentials.

ACKNOWLEDGMENTS

Definitely I have to thank my friend Mario

NON SOLUM

for all the time he made me save every day, by means of inspiring, stimulating and more and more interesting walks from Piazzale Tecchio to Via Nuova Agnano

SED ETIAM

for having shared the enthusiasm

- for realizing that the robust control is not only theory and the estimates of the dynamics parameters can actually converge to the real ones
- while watching the SCARA bouncing on a compliant plane and especially while moving towards lower potentials

and for his valuable cinematographic advice!

There's one more thing I'd like to tell him to include everything I forgot to mention:

'E PUNTI 'E VIA!

† B. Sc. Mechanical Engineering, Dipl. Ing. Automotive Engineering
MATRICULATION NUMBER: M64/455
Department of Industrial Engineering *(leider)*
University of Naples "Federico II", Napoli, Italy
EMAIL: g.notomista@studenti.unina.it, g.notomista@gmail.com

1 INTRODUCTION

This introduction is meant to give a rough overview on the whole report and a small description of its protagonist.

Starting from the latter, in Fig. 1 "Oscar 'o SCARA", a SCARA robot that lives somewhere behind the Vesuvio, can be seen in one of the picture taken from Ischia while it is performing a trajectory tracking task. This robot will accompany us in our trip through robot control starting from Gragnano and arriving to Marechiaro.

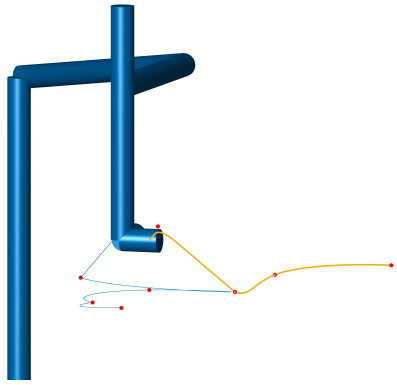


Figure 1: Oscar 'o SCARA

More in detail, this trip will make us explore the basics of Robotics step by step, each of which is represented by a section of this report:

- Section 2: this first section deals with manipulability and dexterity characteristics of the robot represented by the velocity, force and dynamics manipulability ellipsoids.
- Section 3: this second section gives a (panoramic) description of the trajectory designed for the robot.
- Section 4: this section is the most important section since it will present the methods that can be used to solve the inverse kinematics problem in Robotics.
- Section 5: in this section a further step is done considering the robot redundancy and trying to exploiting it in the kinematics inversion.
- Sections 6 and 7: from here on, the dynamics of the robot is taken into consideration and two different techniques used for motion control will be presented right in these two sections.
- Section 8: this section is dedicated to the force control and, in its subsection, 4 force control strategies are explained respectively.
- Section 9: in this last section, that goes a little bit beyond the classic robot control, maybe the most elegant motion planning technique used in Robotics will be presented.

Furthermore, 2 appendices at the end have been used not to include too many formulas in the main part of the report and to explain more in detail some of the functions used in this project. In particular:

- Appendix A: here the kinematics and dynamics characteristics of Oscar 'o SCARA are reported.
- Appendix B: this section is meant to explain in a clearer way how some features of the the trajectory planning function work.

Finally, attached to this PDF report, there are:

- all the files .m (quite) extensively commented and organized in folders,
- the published PDFs of the scripts used to generate the graphs and the videos which are part of this report,
- the HTML *Scara* class reference created for this project using MathWorks® MATLAB.

Before closing this brief introduction, the author would like to excuse for some text or digits which can be present in some figures or videos that, even though they could be irrelevant, are unreadable.

Moreover, throughout this work, vectors and matrices are denoted by lower and upper case bold letters.

2 RIGHT VELOCITIES AND RIGHT FORCES AT THE RIGHT ANGLE

Before going into the details of the manipulability measures that can be used to check the dexterity of a robot, a very interesting and meaningful technique used in matrix analysis is wanted to be introduced.

2.1 From A Symmetric Positive Definite Matrix To The Hyperellipsoid

> *"Simmetrica e definita positiva: queste sono due delle proprietà più nobili che una matrice possa avere."*
>
> L. ROSATI

If a matrix $\mathbf{A}$ is positive definite and symmetric, the quadratic form $\mathbf{q}(\mathbf{x})$ associated with it has a special property: the points satisfying the equation $\mathbf{q}(\mathbf{x}) = 0$ belongs to the surface of a hyperellipsoid.

In the 2-dimensional case, defining the homogeneous coordinate vector

$$\mathbf{x} = [x, y, 1]^T$$

and the matrix

$$\mathbf{A} = \begin{bmatrix} A & \frac{B}{2} & \frac{D}{2} \\ \frac{B}{2} & C & \frac{E}{2} \\ \frac{D}{2} & \frac{E}{2} & F \end{bmatrix},$$

we would have

$$\mathbf{q}(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x} = Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$$

which, if $\mathbf{A}$ is positive definite, is the equation of an ellipse.

This concept, extended to higher-dimensional spaces, can be applied to particular symmetric and positive definite matrices related to the kinematics and the dynamics of a robot.

Indicating with $\mathbf{J}$ the Jacobian of the robot and with $\mathbf{B}$ its inertia matrix, the three matrices [SSVO09]

$$\begin{aligned}\mathbf{M}_v &= (\mathbf{J}\mathbf{J}^T)^{-1}\\ \mathbf{M}_f &= \mathbf{M}_v^{-1}\\ \mathbf{M}_d &= \mathbf{J}^T\mathbf{B}^T\mathbf{B}\mathbf{J}\end{aligned}$$

are associated with three particular quadratic forms representing the

- velocity ellipsoid
- force ellipsoid
- dynamics manipulability ellipsoid.

The length of the semiaxes of these ellipsoids are equal to the singular values[1] of the corresponding matrices. Therefore, it can be said that the velocity and force ellipsoids are *orthogonal*, i. e. the semiaxes are in inverse ratios between each other. The following theorem, together with the figures of Subsection 2.2, will clarify this concept.

Theorem 1. *If a matrix $\mathbf{A}$ is the inverse of $\mathbf{B}$, the eigenvalues of $\mathbf{A}$ are the reciprocal numbers of the eigenvalues of $\mathbf{B}$.*

Proof. The theorem can be easily demonstrated exploiting the eigendecomposition of the matrices $\mathbf{A}$ and $\mathbf{B}$.

In fact, according to the spectral theorem

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T = \mathbf{V}\begin{pmatrix}\lambda_1 & & & \\ & \lambda_2 & & \mathbf{O} \\ & & \ddots & \\ & \mathbf{O} & & \ddots \\ & & & & \lambda_n\end{pmatrix}\mathbf{V}^T,$$

where $\mathbf{V}$ is an orthonormal matrix, i. e. $\mathbf{V}^{-1} = \mathbf{V}^T$

Hence

$$\begin{aligned}\mathbf{B} = \mathbf{A}^{-1} &= (\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T)^{-1} = \mathbf{V}^{-T}\mathbf{\Lambda}^{-1}\mathbf{V}^{-1}\\ &= \mathbf{V}\begin{pmatrix}\frac{1}{\lambda_1} & & & \\ & \frac{1}{\lambda_2} & & \mathbf{O} \\ & & \ddots & \\ & \mathbf{O} & & \ddots \\ & & & & \frac{1}{\lambda_n}\end{pmatrix}\mathbf{V}^T.\end{aligned} \tag{1}$$

As we can see from Equation (1), the first eigenvalue of $\mathbf{B}$ is the reciprocal number of the first eigenvalue of $\mathbf{A}$ and so on.

□

Since, as said before, the absolute values of the eigenvalues are the lengths of the semiaxes, it can be said that the ellipsoids corresponding to the matrices $\mathbf{A}$ and $\mathbf{B}$ are *orthogonal* to each other.

1 In this case, since the matrices are symmetric, their singular values are equal to the absolute value of their eigenvalues, which are all real.

2.2 Manipulability Ellipsoids

In Fig. 2 and Fig. 3 the velocity (*red*) and force (*green*) ellipsoids are depicted for 7 positions of the robot end-effector. Here the *orthogonality* between the two ellipsoids (associated with the matrices $\mathbf{M}_v$ and $\mathbf{M}_f$) can be fully appreciated.

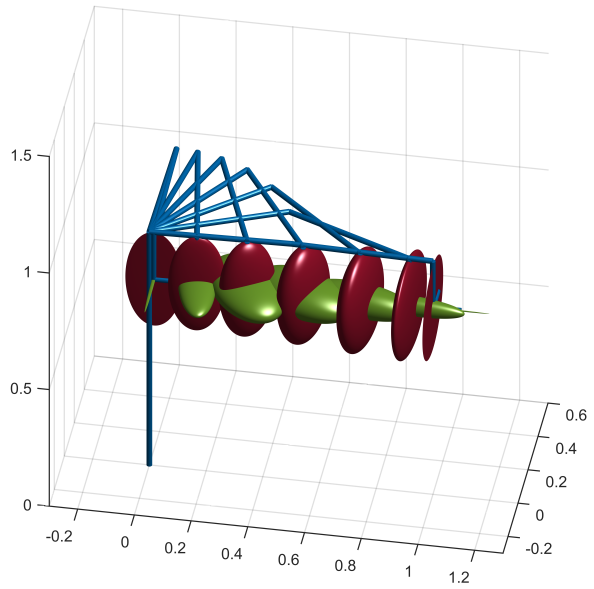


Figure 2: Velocity, force and dynamics manipulability ellipsoids

In particular, from Fig. 3, it can be noticed that the top view of the SCARA manipulability ellipsoids are, on a quality level, equal to those related to the two-link planar arm.

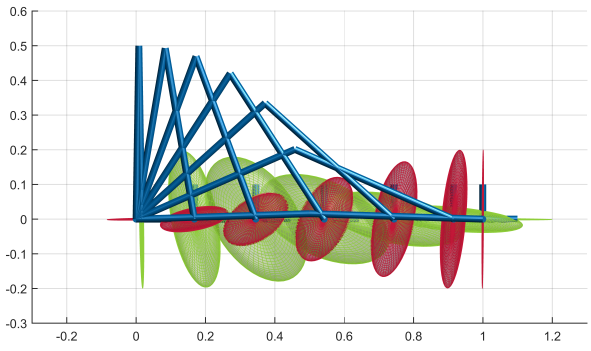


Figure 3: Velocity, force and dynamics manipulability ellipsoids (top view)

Furthermore, in the two limiting cases, in which the robot is either outstretched or retracted, the two ellipsoids degenerate into a circle and a segment orthogonal to it (see Fig. 2).

Finally, the two videos that close this section show the transformation of the two above-mentioned ellipsoids and the dynamics manipulability ellipsoid while the robot is moving in its workspace.

Video: Transformation of the ellipsoids while the SCARA moves in its workspace

Video: Transformation of the ellipsoids while the SCARA moves in its workspace (top view)

3 STAIRWAY TO THE STARS

"Stairway To The Stars" is a popular jazz standard composed by Matty Malneck and Frank Signorelli in 1926.

Fig. 4 shows the path planned to test both the trajectory planning function and the adopted control schemes.

The trajectory is characterized by values of Tables 1 and 3. In the former, the points are described by the 3 components of the position vectors, expressed in the base reference frame of the robot, and by the timing information, given in the first column. The third and the fourth column are needed

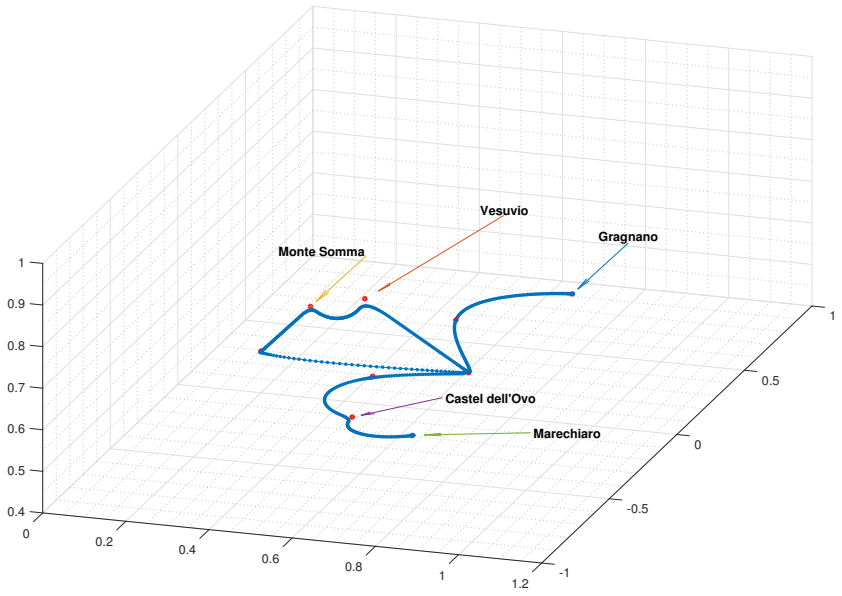


Figure 4: Trajectory planning

to distinguish the type of points, of which the path is made up, i. e. *path* or *via* points. In case of via points, in the fourth column there is the advance (expressed in percentage of the duration of the path segment that starts from that via point) used to anticipate the generation of the trapezoidal profile of the path segment that starts from that via point. The last two columns give the orientation of the robot end-effector using the angle/axis representation, where the axis z_0 is the z-axis of the base reference frame of the robot.

Table 1: Path Description

Time	Point			Orientation	
[s]	Coordinates $[x, y, z]$	Types	Advance	Axis	Angle [rad]
0	$\left[\frac{0.8}{\sqrt{2}} + \frac{1}{5}, \frac{0.8}{\sqrt{2}}, \frac{13}{25}\right]$	path	–	z_0	0
5	$\left[\frac{1}{2\sqrt{2}} + \frac{1}{5}, \frac{1}{2\sqrt{2}}, \frac{1}{2}\right]$	path	–	z_0	$\frac{\pi}{4}$
8	$\left[\frac{7}{10}, 0, \frac{1}{2}\right]$	path	–	z_0	$\frac{\pi}{3}$
12	$\left[\frac{9}{20}, 0, \frac{13}{20}\right]$	via	30%	z_0	$\frac{\pi}{6}$
16	$\left[\frac{8}{25}, 0, \frac{31}{50}\right]$	via	30%	z_0	0
19	$\left[\frac{1}{5}, 0, \frac{1}{2}\right]$	path	–	z_0	$\frac{\pi}{2}$
22	$\left[\frac{1}{2} + \frac{1}{5}, 0, \frac{1}{2}\right]$	via	50%	z_0	$\frac{\pi}{2}$
24	$\left[\frac{1}{2}, -\frac{1}{10}, \frac{1}{2}\right]$	path	–	z_0	$\frac{\pi}{3}$
27	$\left[\frac{11}{20}, -\frac{2}{5}, \frac{1}{2}\right]$	via	20%	z_0	0
30	$\left[\frac{3}{4}\cos\left(\frac{8}{10}\frac{\pi}{2}\right) + \frac{1}{2}, -\frac{3}{4}\sin\left(\frac{8}{10}\frac{\pi}{2}\right) + \frac{1}{5}, \frac{51}{100}\right]$	path	–	z_0	0

The characteristics of the path segments are described in Table 3. Here there is the segment ID, the segment curvature and its speed profile. The

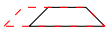
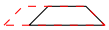
curvature values can assume the values reported in Table 2 with the corresponding meanings.

Table 2: Segment Curvatures

Curvature value	Meaning
-1	spline
0	rectilinear path
$\frac{1}{\rho}^*$	circular path

*ρ is the radius of the circumference of the circular path

Table 3: Pieces Of Trajectory

#	Curvature		Speed Profile
	Value	Properties	
1	-1	–	
2	-1	–	
3	0	–	
4	$\frac{1}{0.08}$	z_0	
5	0	–	
6	-1	–	
7	-1	–	
8	$\frac{1}{0.18}$	z_0	
9	$\frac{1}{0.12}$	z_0	

The *Properties* of the curvature are set, only in correspondence of circular paths, to the unit vector describing the plane in which the circumference will be planned.

The *Speed Profile* column of the table depicts the speed profile assigned to each segment. In case of via points, the red dotted profile is obtained anticipating the generation of the profile. The amount of advance can be read in Table 1.

Moreover, from an implementation point of view, the inputs needed to generate the speed profile are:

- the percentage of the acceleration phase
- the percentage of the deceleration phase.

Starting from this information the acceleration, deceleration and maximum speed values are calculated.

Finally, the video that closes this section shows the path in a moving bird's-eye view. From that panoramic view it can be appreciated the trip starting in Gragnano and reaching Marechiaro passing through the Vesuvio, Monte Somma and Castel dell'Ovo.

Video: Presentation of the trajectory with POIs

4 SHALL WE GO BACK WITH DIEGO?

Did you know that the etymological evolution of the Spanish name Diego shows that it may be originated from Jacob, the name of the Hebrew patriarch?

This section shows the results of the inverse kinematics algorithms tested to evaluate the joint variable functions needed to track the trajectory formerly planned.

The techniques tested here exploit the power that a feedback loop has in reference tracking tasks. In fact, inverting the basic differential kinematic equation [SSVO09]

$$\mathbf{v}_e = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \tag{2}$$

where $\mathbf{v}_e$ is the end-effector velocity, $\mathbf{J}$ is the robot Jacobian and $\mathbf{q}$ is the joint variable vector, one gets

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{q})\mathbf{v}_e. \tag{3}$$

A mere integration of Eq. (3) would lead inevitably to *drift* of the $\mathbf{q}(t)$ function.

That's why the inverse kinematics is performed by the aid of closed loop schemes based on the robot Jacobian inverse and transpose, and considering both the tracking error and its derivative.

A prior distinction can be done between *offline* and *online* inverse kinematics algorithms. An offline algorithm assures a very small error for every point of the trajectory at the price of bigger computational efforts. Whereas an online algorithm is much faster, but it shows an initial bigger error that goes down along the trajectory.

The use of closed-loop schemes leads to the following choices of joint velocities [SSVO09]:

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{q})\left(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}\right) \qquad \text{Jacobian inverse} \tag{4}$$

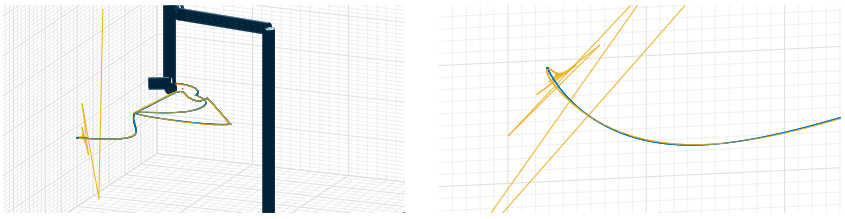
$$\dot{\mathbf{q}} = \mathbf{J}^{T}(\mathbf{q})\mathbf{K}\mathbf{e} \qquad \text{Jacobian transpose} \tag{5}$$

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}(\mathbf{q})\left(\dot{\mathbf{x}}_d + \mathbf{K}\mathbf{e}\right) + \left(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q})\mathbf{J}(\mathbf{q})\right)\dot{\mathbf{q}}_0 \qquad \text{Jacobian pseudo-inverse} \tag{6}$$

$$\ddot{\mathbf{q}} = \mathbf{J}^{\dagger}(\mathbf{q})\left(\ddot{\mathbf{x}}_d + \mathbf{K}_D\dot{\mathbf{e}} + \mathbf{K}_P\mathbf{e} - \dot{\mathbf{J}}(\mathbf{q},\dot{\mathbf{q}})\right) + \left(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q})\mathbf{J}(\mathbf{q})\right)\ddot{\mathbf{q}}_0 \qquad 2^{\text{nd}}\text{-order Jacobian pseudo-inverse,} \tag{7}$$

where the tracking error and its derivative are denoted by $\mathbf{e}$ and $\dot{\mathbf{e}}$.

The key of a closed loop is, as usual, the value of the loop gains ($\mathbf{K}$, $\mathbf{K}_P$ and $\mathbf{K}_D$ in Equations from (4) to (7)). Fig. 5a and Fig. 5b show how increasing the loop gain can bring the system (in this case the numerical integration) close to instability. In Figures 5 the operational space trajectory obtained using the output of the closed loop numerical integration (algorithm based on Eq. (5)) is plotted on the desired trajectory to track. The amplitude of the initial oscillations is very large even though the system manages to converge to the desired trajectory.



(a) Algorithm close to instability **(b)** Algorithm close to instability (close-up)

Figure 5: Jacobian transpose CLIK with too large gains

4.1 'O Sparagno Nun È Maje Guadagno

In the Fig. 6a (close-up in Fig. 6b) there are the position error functions related to the Jacobian inverse and Jacobian transpose CLIK algorithms respectively. Whereas the Fig. 7 shows a comparison between the orientation errors coming from the same two algorithms.

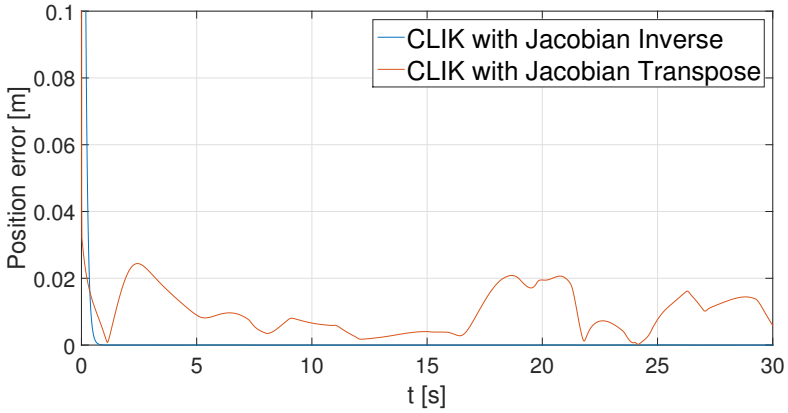
In both cases, position and orientation errors, the CLIK based on the Jacobian inverse performs better. The drawback is a higher computational effort due to the inversion of a matrix at every step of the algorithm. The CLIK based on the Jacobian transpose, on the other hand, is computationally lighter but less precise.

4.2 Double CLIK To Close The Loop

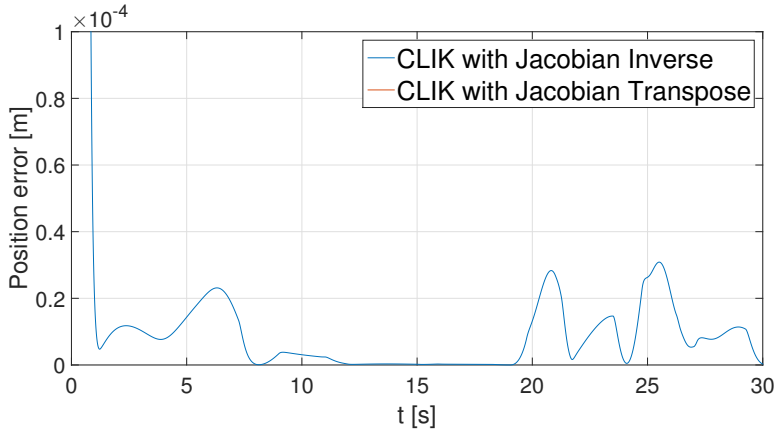
An even more precise CLIK algorithm can be achieved using a second-order algorithm [Sic90]. The operational space acceleration and the derivative of the tracking error come into play allowing an even finer tuning of the *control loop* and, therefore, a better behavior in reference tracking. Here, in fact, both $\mathbf{K}_D$ and $\mathbf{K}_P$ can be assigned to reach the desired transient and tracking behavior.

In Fig. 8a and Fig. 8b the overshoot typical of a second-order system can be observed, while in Fig. 9a and Fig. 9b the loop gains have been tuned to obtain two coincident poles of the *controlled* system. This way, the error goes exponentially to zero without overshoot and with a well-damped behavior.

To close this section a final consideration has to be done. Since for robots with a large number of degrees of freedom the inverse kinematics takes a considerable amount of time, lots of alternative solutions to this problem have been worked out. Among these, just a mention should be done to those based on fuzzy logic [XN93] and on neural networks. The latter is used successfully in advanced application such as the motion control of flexible manipulators in microgravity environments [CL92].



(a) Position errors



(b) Position errors (close-up)

Figure 6: Comparison between operational space position errors obtained using two CLIK algorithms based on the Jacobian inverse and the Jacobian transpose

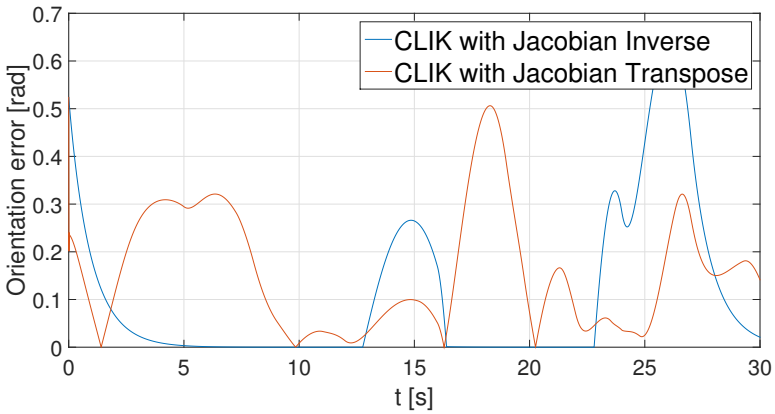
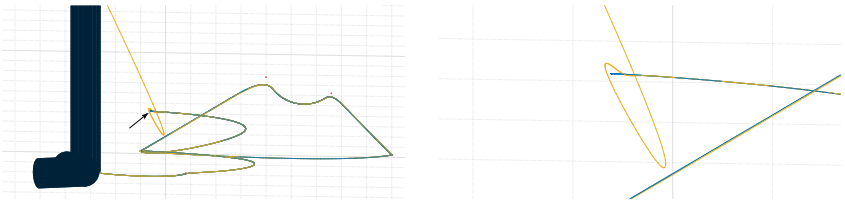


Figure 7: Comparison between operational space orientation errors obtained using two CLIK algorithms based on the Jacobian inverse and the Jacobian transpose

5 LEAP INTO THE VOID

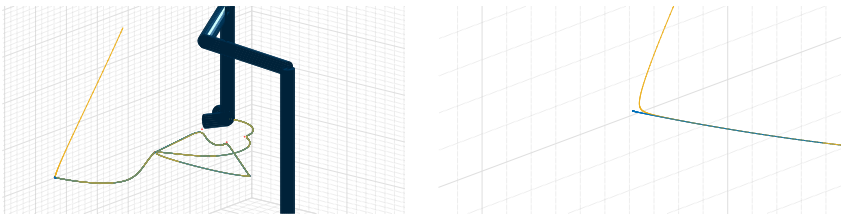
This section shows how manipulator redundancy can be exploited in the inverse kinematics algorithms. In Equations 6 and 7 the terms $\left(\mathbf{I}-\mathbf{J}^{\dagger}(\mathbf{q})\mathbf{J}(\mathbf{q})\right)\dot{\mathbf{q}}_0$



(a) Gain set-up to get overshoot in the second-order behavior

(b) Gain set-up to get overshoot in the second-order behavior (close-up)

Figure 8: Overshoot in second-order CLIK algorithms



(a) Gain set-up to get the critical damping in the second-order behavior

(b) Gain set-up to get the critical damping in the second-order behavior (close-up)

Figure 9: Critical damping in second-order CLIK algorithms

and $(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q})\mathbf{J}(\mathbf{q}))\ddot{\mathbf{q}}_0$ generate additional *internal motions* that don't cause a movement of the robot end-effector, being $(\mathbf{I} - \mathbf{J}^{\dagger}(\mathbf{q})\mathbf{J}(\mathbf{q}) \in \mathcal{N}(\mathbf{J})$ and remembering Eq. (2).

The choice of the values of $\dot{\mathbf{q}}_0$ and $\ddot{\mathbf{q}}_0$ allows a convenient utilization of the redundant degrees of freedom.

In the presented case, it has been decided to maximize the robot manipulability forcing the SCARA robot to be redundant w.r.t. the orientation of its end-effector, renouncing to control the last joint in order to keep the trajectory as close as possible to the planned one.

Making use of a 1st -order algorithm, the value of $\dot{\mathbf{q}}_0$ has been calculated according to the following equation [SSVO09]:

$$\dot{\mathbf{q}}_0 = k_0 \left(\frac{\partial w(\mathbf{q})}{\partial \mathbf{q}} \right)^T,$$

where

$$w(\mathbf{q}) = \sqrt{\det(\mathbf{J}(\mathbf{q})\mathbf{J}^T(\mathbf{q}))}$$

is a manipulability measure.

In Fig. 10 another manipulability function (the ratio between the smallest and the largest singular value of the manipulator Jacobian) is plotted along the trajectory. It turns out to be the same before and after maximizing the manipulability since for the considered robot architecture the manipulability is function of the second joint variable (the angle between the first and the second link). On the other hand, with this technique the last joint is not actuated and therefore the frames attached to the third and the fourth link always coincide.

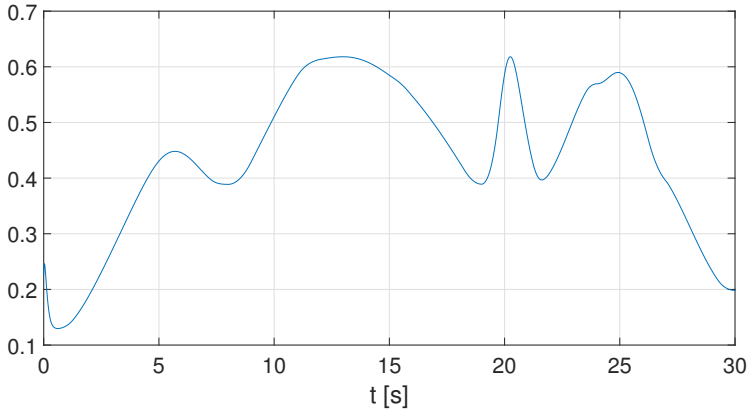


Figure 10: Manipulability measure function (ratio between the smallest and the largest singular value obtained by the SVD of the Jacobian)

6 CAN YOU KEEP THE EQUILIBRIUM ON A SLIDING PLANE?

With this section the study of the control strategies for robot manipulators begins. So far only the kinematic control has been considered, while from now on the robot dynamics comes into play too.

Finding the joint torques that make the robot follow the given path with the given velocities and accelerations means finding the right matrix that is able to rotate the force vector applied to the robot end-effector onto the desired acceleration vector.

The first two strategies (Section 6 and Section 7) are meant to reject special kinds of disturbances that can occur while operating with the robot. In particular an unknown mass of 4 Kg has been hung at the manipulator end-effector. The two considered control schemes are two kinds of centralized control:

- robust control
- adaptive control.

The first one is presented in this section while the next section is dedicated to the second one.

The robust control comes into play whenever the robot dynamic model is not 100 % known. It allows to catch up on the imperfect compensation coming from an inverse dynamics control implemented with the wrong dynamic parameters. The control scheme can be actually divided into three main parts:

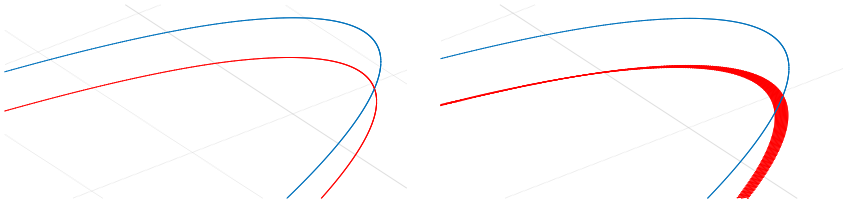
- nonlinear compensation and decoupling
- stabilizing linear control
- unit vector control.

The first two parts are in common with the well-known inverse dynamics control scheme, while the third one takes care of counteracting the uncertainty in the knowledge of the robot dynamics.

The choice of the parameters of the unit vector control can assure robustness as well as both stability and good performances of the control loop,

attracting the error towards a *sliding* subspace of the tracking error state space, to which the error and its derivative are confined.

In Figures 11 it can be appreciated how the unit vector control action can limit, with two different behaviors, the magnitude of the error to small values. To simulate the uncertainty in the robot dynamic model, the **B** and **C** matrices of the dynamic model used for the nonlinear compensation have been set to be diagonal, neglecting the off-diagonal entries. In Fig. 11a the value of the ϵ parameter of the robust control law is set to a big value: this leads to a control action that contains high-frequency components (*chattering*), but also to a better reference tracking, since it is able to bring the error to 0. In Fig. 11b, instead, the value of ϵ has been reduced and the chattering disappears. The error norm is bounded but it never goes to 0.



(a) Close-up on the trajectory (big ϵ)

(b) Close-up on the trajectory (small ϵ eliminates the typical *chattering* behavior due to high-frequency components in the control action)

Figure 11: Comparison between the behaviors of the controlled trajectory using a unit vector control law with different values of the ϵ parameter

In Fig. 12a and Fig. 12b the joint torques related to the two above-described alternatives of the unit vector control are reported as functions of time. Also here it is possible to see how the high-frequency in the commutation of the control action generates smoother reference torques for the servos, which are in reality incapable to follow them.

To close this section there is a video showing Oscar 'o SCARA tracking the planned trajectory by using a robust control. The color of the joints goes from green to red proportionally to joint torque exerted by the joint motor.

7 Y MUST YOU ADAPT ALL THE π RECIPES?!

The title of this section is clearly, explicitly and no doubt dedicated to my mother. She already knows!

The adaptive control allows an on-line adaptation of the dynamics parameters used to compensate the nonlinearities of the system to control. This is achieved exploiting the linearity of the dynamic model w.r.t the dynamic parameters.

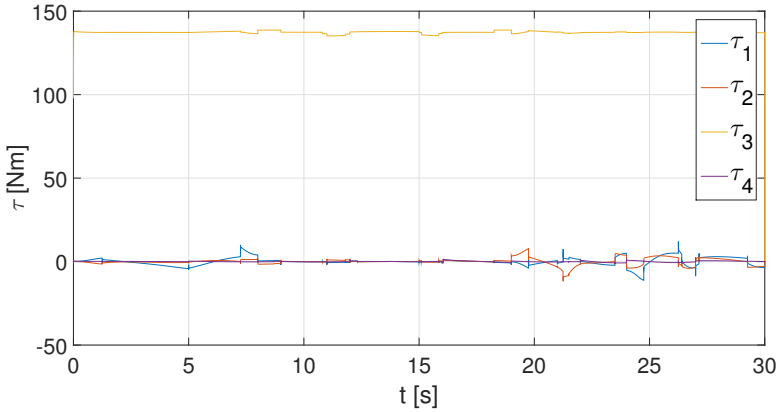
The Eq. (8) sums up the adaptive control law, while the Eq. (9) gives the parameter adaptive law [SSVO09].

$$\mathbf{u} = \underbrace{\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \dot{\mathbf{q}}_r, \ddot{\mathbf{q}}_r)\hat{\boldsymbol{\pi}}}_{\text{inverse dynamics compensation}} + \underbrace{\mathbf{K}_D\boldsymbol{\sigma}}_{\text{PD linear control}} \tag{8}$$

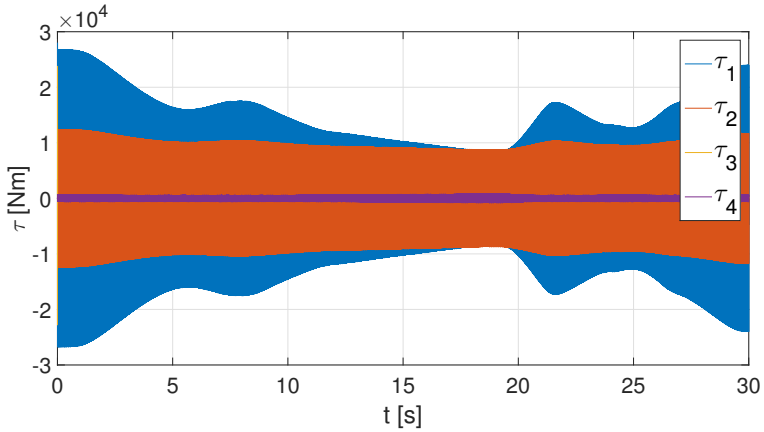
$$\dot{\hat{\boldsymbol{\pi}}} = \mathbf{K}_\pi^{-1}\mathbf{Y}^T\boldsymbol{\sigma} \tag{9}$$

In these equations

- $\mathbf{Y}$ is the *regressor*,



(a) Joint torques (big ϵ)



(b) Joint torques (with small ϵ the *chattering* is also visible in the joint torque functions)

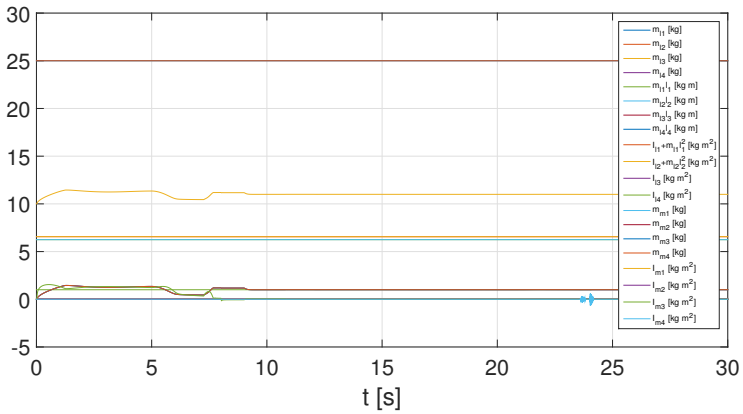
Figure 12: Comparison between the behavior of the joint torques using a unit vector control law with different values of the ϵ parameter

Video: SCARA controlled adopting a robust control law

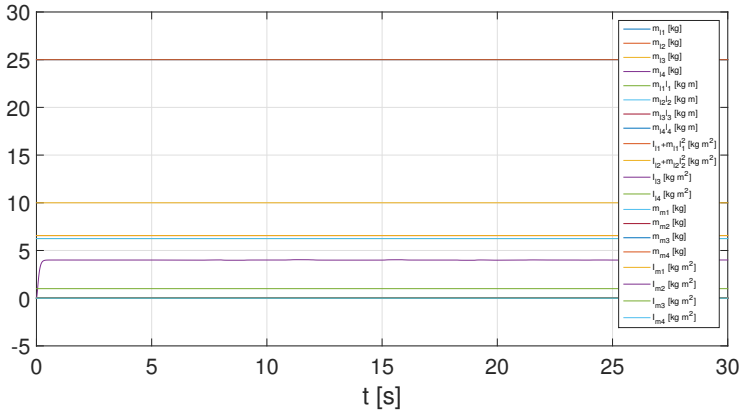
- $\hat{\boldsymbol{\pi}}$ is the vector of the estimated parameters,
- $\mathbf{K}_D$ is the derivative gain matrix,

- $\boldsymbol{\sigma}$ is the joint velocity error vector,
- $\mathbf{K}_\pi$ is the gain matrix that has to be tuned in order to control the behavior of the convergence of the dynamic parameters.

The Figures 13 show two solutions of the parameter adaptation that comes from two different choices of the matrix $\mathbf{K}_\pi$. In Fig. 13a the gain matrix is set to be diagonal and the resulting adaptation process is *handed-out* to more parameters. Whereas in Fig. 13b the entries of $\mathbf{K}_\pi$ have been chosen so that the unknown mass hung to the robot end-effector will make only one dynamic parameter change, namely the mass of the last link, which, of course, converges to 4 Kg.



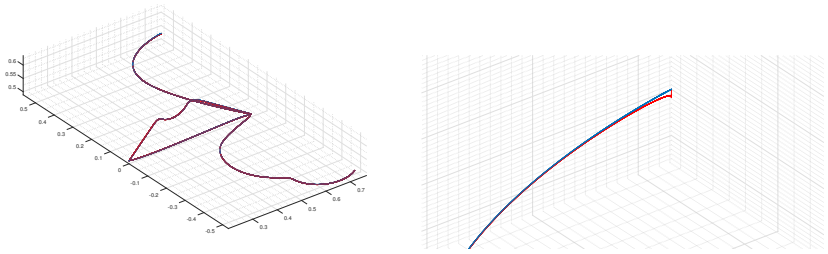
(a) Parameter adaptation diagram obtained using a gain matrix K_π with all eigenvalues equal



(b) Parameter adaptation diagram obtained using different gains for each parameter in order to simulate the estimation process of the single mass of the last link

Figure 13: Adaptation of dynamics parameters

In Fig. 14a and in particular in Fig. 14b the trajectory that the robot follows under an adaptive control law is plotted in red. It can be observed how at the beginning, when the estimations of the dynamic parameters are wrong, there is a certain position error in the z_0 direction; the more the time goes on, the better the parameter estimation is, and the tracking error decreases to 0.



(a) Controlled trajectory obtained using an adaptive control strategy

(b) Controlled trajectory obtained using an adaptive control strategy (close-up on the starting phase while the parameters are being adapted)

Figure 14: Trajectory resulting by the use of an adaptive control law

8 MAY THE FORCE BE WITH YOU

This line has been said at least once in each of the "Star Wars" movies, which are probably the most famous movies whose main characters are robots.

This section is dedicated to the study of force control and its different applications.

In the next subsection an indirect force control is taken into consideration, while in Subsections 8.2, 8.3 and 8.4 the results of 3 direct methods to control force are reported.

8.1 Impedance Control

The impedance control is an indirect force control method since there is not an explicit force reference to which the control system tries to converge. Instead, the end-effector force comes only from the interaction between the robot and the environment.

In order to make a force rise, an elastically compliant plane has been inserted to try to impede the robot movements.

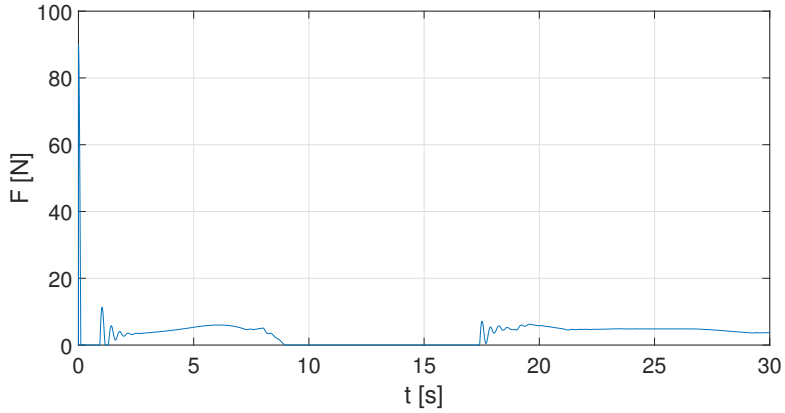
In Fig. 15a the absolute value of the contact force between the robot end-effector and the plane is plotted against the time. In Fig. 15b, that shows only the first 3 seconds, the transient behavior can be observed. This behavior depends on the choice of the control parameters that build-up the control action of Eq. (11) [SSVO09].

$$\mathbf{u} = \mathbf{B}(\mathbf{q})\mathbf{y} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{J}^{T}(\mathbf{q})\mathbf{h}_{e} \tag{10}$$

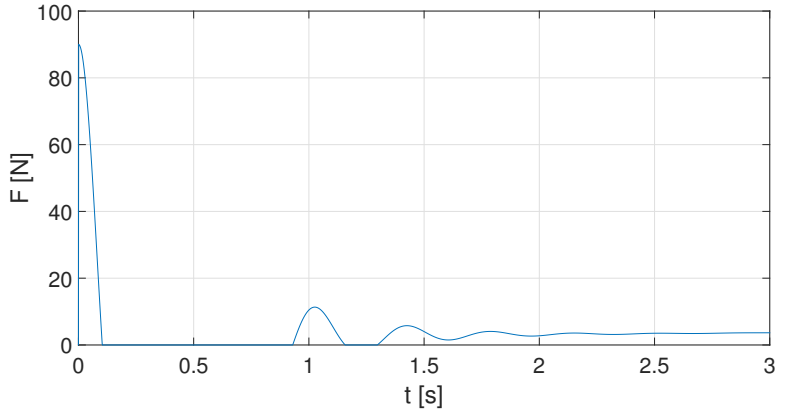
$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\mathbf{M}_{d}^{-1}\left(\mathbf{M}_{d}\ddot{\mathbf{x}}_{d} + \mathbf{K}_{D}\dot{\tilde{\mathbf{x}}} + \mathbf{K}_{P}\tilde{\mathbf{x}} - \mathbf{M}_{d}\dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{h}_{A}\right) \tag{11}$$

With $\mathbf{M}_d$, $\mathbf{K}_D$ and $\mathbf{K}_P$ the damping and the overshoot (i. e. rise time and settling time) of the second-order system by which the robot is approximated can be set.

The video on Page 20 shows Oscar 'o SCARA following the trajectory under an impedance control. The obstacle plane is shown in transparency, the contact force is plotted and the joints torques are depicted as in the video on Page 17. In the video the second order behavior has been intentionally emphasized properly tuning the inertia, stiffness and damping matrices of the controlled system.



(a) Contact force calculated using an impedance control



(b) Contact force calculated using an impedance control (close-up on the first 3 seconds)

Figure 15: Contact force between the SCARA end-effector and an obstacle plane placed in the workspace

Video: SCARA movement obtained by using an impedance control

8.2 Force Control With Inner Position Loop

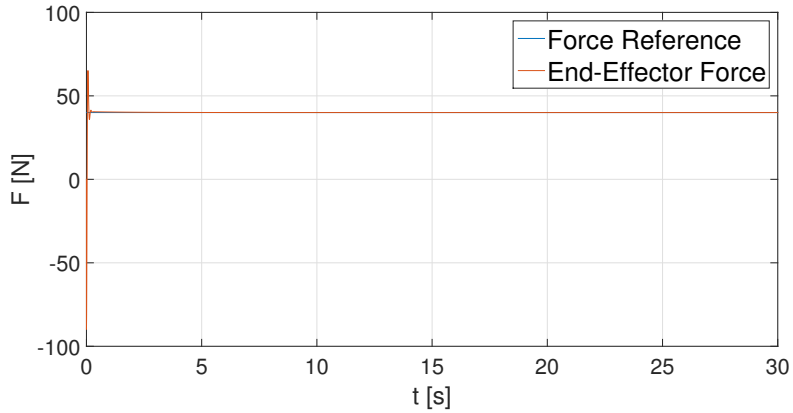
To achieve a direct force control, the control laws reported in this section and in the next two could be used.

A first solution consists in considering a cascade control scheme in which the outer loop takes the force as a reference and gives the reference for the inner position loop. The control law corresponding to this strategy is as in Eq. (10) but the vector $\mathbf{y}$ is given by Eq. (12). The closing of the force loop is assured by Eq. (13) [SSVO09].

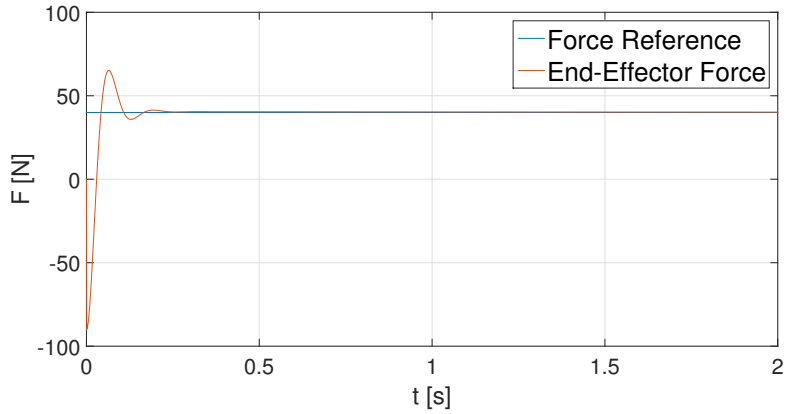
$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\mathbf{M}_d^{-1}\left(\mathbf{K}_D\dot{\mathbf{x}}_e + \mathbf{K}_P(\mathbf{x}_F - \mathbf{x}_e) - \mathbf{M}_d\dot{\mathbf{J}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}\right) \tag{12}$$

$$\mathbf{x}_F = \mathbf{C}_F(\mathbf{f}_d - \mathbf{f}_e) \tag{13}$$

The Fig. 16 (zoom to the first 2 seconds in Fig. 16b) shows the reference force along the z_0 axis (set to 40 N) and the actual end-effector force. This transient behavior, as well as the astatism, have been achieved by using and tuning the transfer function of a PI controller in place of the $\mathbf{C}_F$ of Eq. 13.



(a) End-effector force with the reference value



(b) End-effector force with the reference value (close-up on the first 2 seconds)

Figure 16: Force control with inner position loop

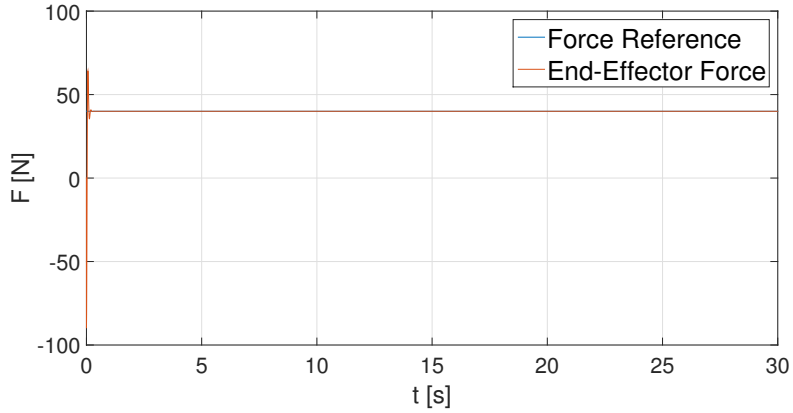
8.3 Force Control With Inner Velocity Loop

An easier control scheme can be obtained if one considers the same force control of Subsection 8.2 but with an inner velocity loop. The control input is as in Eq. (10), the vector $\mathbf{y}$ changes to Eq. (14) [SSVO09].

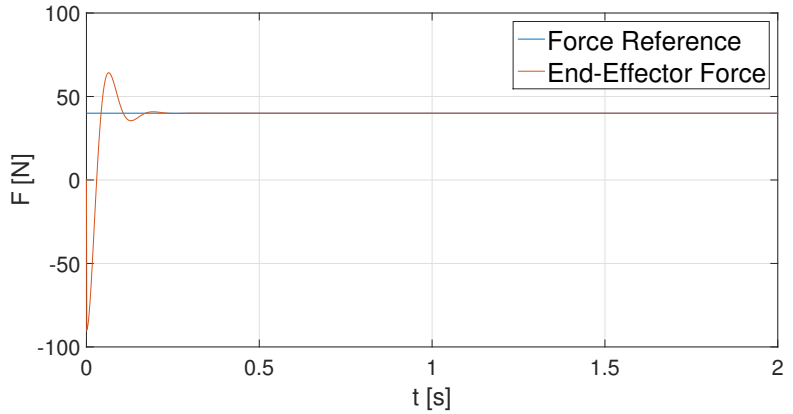
$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\mathbf{M}_d^{-1}\left(-\mathbf{K}_D\dot{\mathbf{x}}_e + \mathbf{K}_P\mathbf{x}_F - \mathbf{M}_d\dot{\mathbf{J}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}\right) \tag{14}$$

In this case, in fact, both the astatism and a good transient behavior, can be achieved by using a simple P controller in place of $\mathbf{C}_F$.

From Fig. 17a and Fig. 17b it can be easily noticed that there is almost no difference between the transient of the end-effector force of Fig. 16a and Fig. 16b.



(a) End-effector force with the reference value



(b) End-effector force with the reference value (close-up on the first 2 seconds)

Figure 17: Force control with inner velocity loop

8.4 Parallel Force/Position Control

Since in the previous two control laws the position reference does not appear explicitly, if the desired force has zero components along certain operational space directions (i. e. $\mathbf{f}_d \notin \mathcal{R}(\mathbf{C}_F)$), the position reference generated by the outer force loop gives an end-effector velocity reference that will cause inevitably a drift of the end-effector position[2].

If it is desired to control both the end-effector force and its position, the control action given by Eq. (15) can be used [SSVO09].

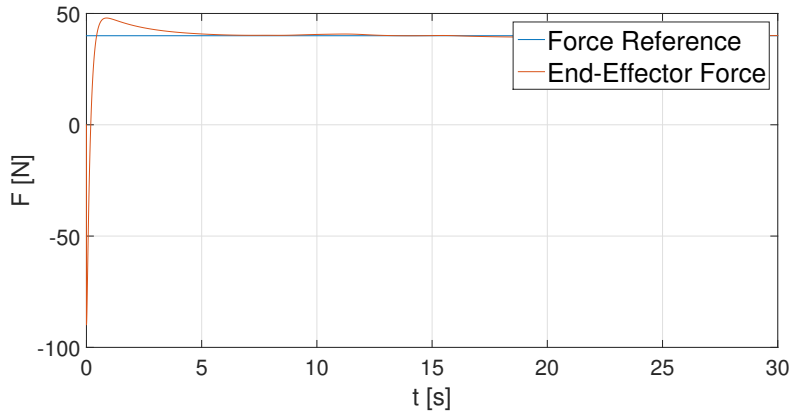
2 In the considered case, in fact, since the force reference vector is $\mathbf{f}_d = [0, 0, 40]$ N, only the z component of the planned path is tracked. The x and y are 0 and the robot stands still almost right above the initial position, at the height at which the force equilibrium is fulfilled.

$$\mathbf{y} = \mathbf{J}^{-1}(\mathbf{q})\mathbf{M}_d^{-1}\left(-\mathbf{K}_D\dot{\mathbf{x}}_e + \mathbf{K}_P(\tilde{\mathbf{x}} + \mathbf{x}_F) - \mathbf{M}_d\dot{\mathbf{J}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}\right) \tag{15}$$

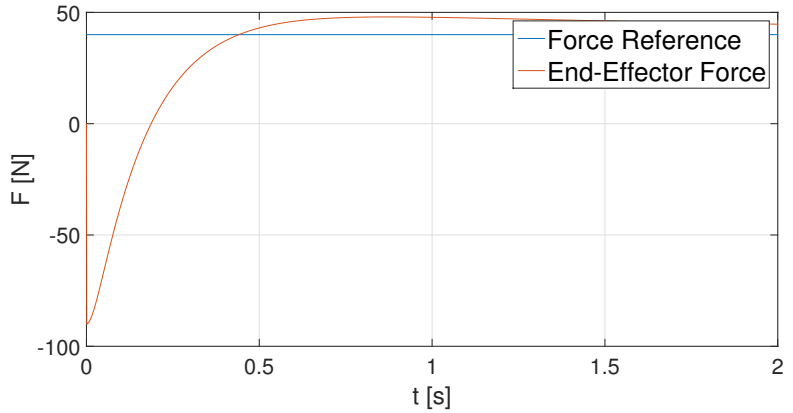
$$\tilde{\mathbf{x}} = \mathbf{x}_d - \mathbf{x}_e \tag{16}$$

The position error of Eq. (16) is typical of pure position control schemes.

Also for this case, Fig. 18a and Fig. 18b show the end-effector force, always along z axis, together with its reference value.



(a) End-effector force with the reference value



(b) End-effector force with the reference value (close-up on the first 2 seconds)

Figure 18: Parallel Force/Position Control

The Fig. 19, furthermore, shows the position of the end-effector under the parallel force/position control.

9 TRIBUTE TO OUSSAMA

This section, that goes beyond the classic robot control, deals with the motion planning technique that makes use of artificial potentials.

The 3 videos of Page 25 show Oscar 'o SCARA moving in its workspace guided (pushed or pulled) by a force applied to its end-effector proportional to the gradient of a particular potential function. Both the shape and the position of Gaussian attractive and repulsive potentials have been changed in order to make the robot perform different *slaloms* passing very close or even through its arm singularities.

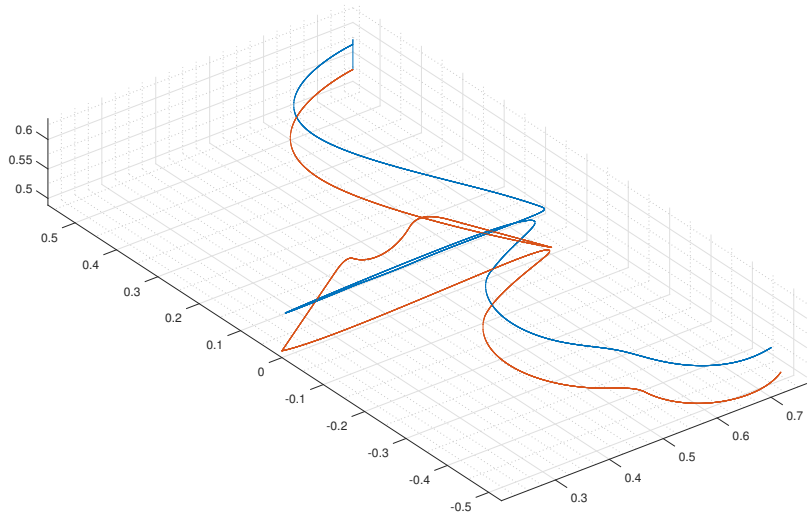


Figure 19: Trajectory executed by the SCARA controlled via a parallel force/position control strategy

Also the effect of local minima has been tested trying to attract the robot end-effector to a force equilibrium point.

Another funny application has been developed to interact with Oscar 'o SCARA while it is moving towards lower potentials. This application consists in controlling the position of the attractive potential (which determines the position of the *goal*) by means of the gyro sensor of a mobile phone. The video on Page 26 shows the author of this report playing with the robot. As can be noticed, compared to the previous videos, no damping force has been introduced not to make the robot slowing down while approaching the goal, but instead to reproduce exactly the behavior of a nonlinear mass-spring system. Again here, it has been tried to make the robot pass through its kinematics singularities to observe the effects coming from the bad-conditioned Jacobian and the improvements that can be achieved using the *damped least-squares* (DLS) Jacobian inverse.

10 CONCLUSION

To conclude this report some further developments and future scopes to keep on improving this project are mentioned below:

1. improve the MATLAB functions written to perform simulations
2. translate the improved MATLAB functions in C++ in order to use them on a real system
3. develop a small application to play with the robot tip to test directly the influence impedance control parameters
4. integrate the functions developed during the course of FEM *in Nonlinear Structural Analysis* to study the problem of controlling flexible manipulators

(a) First configuration of obstacles

(b) Second configuration of obstacles

(c) Third configuration of obstacles

Video: Trajectory planning using artificial potentials

Video: Oscar 'o SCARA chasing the goal

5. build a quadcopter with a smaller version of Oscar 'o SCARA mounted upside-down right below it[3] and use it to test higher-level algorithms developed during other projects and in the free time

The author hopes you enjoyed the trip and . . . don't miss the next one:

"From Marechiaro to Stanford".

3 The production of the mechanical components of the robot will be done using a CNC, which right now is in the design phase.

A SCARA ROBOT KINEMATICS AND DYNAMICS CHARACTERISTICS

A.1 DH Parameters

According to the Denavit-Hartenberg convention, the considered DH parameters have been reported in the following table (units of measurement in meters and radians):

Table 4: DH parameters for the SCARA

Link	a_i	α_i	d_i	ϑ_i
1	0.5	0	0	ϑ_1
2	0.5	0	0	ϑ_2
3	0	0	d_3	0
4	0	0	0	ϑ_4

A.2 Geometric Jacobian

Defining the vector $\mathbf{q} = [\vartheta_1, \vartheta_2, d_3, \vartheta_4]^T$ the geometric Jacobian, used throughout this work starting from the kinematics till the control strategies simulations, can be written as follows:

$$\mathbf{J}(\mathbf{q}) = \begin{bmatrix} -a_1 sin(\vartheta_1) - a_2 sin(\vartheta_1 + \vartheta_2) & -a_2 sin(\vartheta_1 + \vartheta_2) & 0 & 0 \\ a_1 cos(\vartheta_1) + a_2 cos(\vartheta_1 + \vartheta_2) & a_2 cos(\vartheta_1 + \vartheta_2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}. \quad (17)$$

A.3 Inertia, Centrifugal-Coriolis and Gravity Matrices

In this subsection the matrices needed for the simulation of the robot dynamics are reported.

The inertia matrix, which multiplied by the joint variable accelerations gives the inertial generalized forces, is:

$$\mathbf{B}(\mathbf{q}) = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix}, \quad (18)$$

whose non-zero components are reported below:

$$
\begin{aligned}
b_{11} &= I_{l_1} + I_{l_2} + I_{l_3} + I_{l_4} + I_{m_2} + I_{m_3} + I_{m_4} + I_{m_1} k_{r_1}^2 + m_{l_1} l_1^2 \\
&\quad + m_{l_2} a_1^2 + m_{l_2} l_2^2 + m_{l_3} a_1^2 + m_{l_3} a_2^2 + m_{l_4} a_1^2 + m_{l_4} a_2^2 \\
&\quad + m_{m_2} a_1^2 + m_{m_3} a_1^2 + m_{m_4} a_1^2 \\
&\quad + 2\big(m_{l_2} l_2 + (m_{l_3} + m_{l_4}) a_2\big) a_1 \cos(\vartheta_2) \\
b_{12} &= I_{l_2} + I_{l_3} + I_{l_4} + I_{m_3} + I_{m_4} + I_{m_2} k_{r_2} \\
&\quad + m_{l_2} l_2^2 + m_{l_3} a_2^2 + m_{l_4} a_2^2 + m_{m_3} a_1^2 + m_{m_4} a_1^2 \\
&\quad + \big(m_{l_2} l_2 + (m_{l_3} + m_{l_4} + m_{m_3} + m_{m_4}) a_2\big) a_1 \cos(\vartheta_2) \\
b_{13} &= I_{m_3} k_{r_3} \\
b_{14} &= I_{l_4} + I_{m_4} k_{r_4} \\
b_{21} &= I_{l_2} + I_{l_3} + I_{l_4} + I_{m_3} + I_{m_4} + I_{m_2} k_{r_2} \\
&\quad + m_{l_2} l_2^2 + m_{l_3} a_2^2 + m_{l_4} a_2^2 + m_{m_3} a_1^2 + m_{m_4} a_1^2 \\
&\quad + \big(m_{l_2} l_2 + (m_{l_3} + m_{l_4} + m_{m_3} + m_{m_4}) a_2\big) a_1 \cos(\vartheta_2) \\
b_{22} &= I_{l_2} + I_{l_3} + I_{l_4} + I_{m_3} + I_{m_4} + I_{m_2} k_{r_2}^2 \\
&\quad + m_{l_2} l_2^2 + m_{l_3} a_2^2 + m_{l_4} a_2^2 + m_{m_3} a_1^2 + m_{m_3} a_2^2 \\
&\quad + m_{m_4} a_1^2 + m_{m_4} a_2^2 + 2(m_{m_3} + m_{m_4}) a_1 a_2 \cos(\vartheta_2) \\
b_{23} &= I_{m_3} k_{r_3} \\
b_{24} &= I_{l_4} + I_{m_4} k_{r_4} \\
b_{31} &= I_{m_3} k_{r_3} \\
b_{32} &= I_{m_3} k_{r_3} \\
b_{33} &= I_{m_3} k_{r_3}^2 + m_{l_3} + m_{l_4} + m_{m_3} + m_{m_4} \\
b_{41} &= I_{l_4} + I_{m_4} k_{r_4} \\
b_{42} &= I_{l_4} + I_{m_4} k_{r_4} \\
b_{44} &= I_{l_4} + I_{m_4} k_{r_4}^2 .
\end{aligned}
\tag{19}
$$

The C matrix that multiplies the joint velocities and contains both the centrifugal and the Coriolis effects is:

$$
\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{bmatrix}, \tag{20}
$$

whose components have been calculated starting from the *Christoffel symbols of the first type* and whose non-zero components are reported below:

$$
\begin{aligned}
c_{11} &= -\big(m_{l_2} l_2 + (m_{l_3} + m_{l_4}) a_2\big) a_1 \sin(\vartheta_2) \dot{\vartheta}_2 \\
c_{12} &= -\big(m_{l_2} l_2 + (m_{l_3} + m_{l_4}) a_2\big) a_1 \sin(\vartheta_2) \dot{\vartheta}_1 \\
&\quad - \big(m_{l_2} l_2 + (m_{l_3} + m_{l_4} + m_{m_3} + m_{m_4}) a_2\big) a_1 \sin(\vartheta_2) \dot{\vartheta}_2 \\
c_{21} &= \big(m_{l_2} l_2 + (m_{l_3} + m_{l_4}) a_2\big) a_1 \sin(\vartheta_2) \dot{\vartheta}_1 \\
c_{22} &= -(m_{m_3} + m_{m_4}) a_1 a_2 \sin(\vartheta_2) \dot{\vartheta}_2
\end{aligned}
\tag{21}
$$

$$
\mathbf{g}(\mathbf{q}) = [0, 0, (m_{l_3} + m_{m_3} + m_{l_4} + m_{m_4}) g, 0]^T, \tag{22}
$$

where the scalar g is the absolute value of the gravity acceleration.

A.4 Regressor matrix

Defining the vector of dynamics parameters

$$\begin{aligned}\boldsymbol{\pi} = [&m_{l_1}, m_{l_2}, m_{l_3}, m_{l_4}, m_{l_1} l_1, m_{l_2} l_2, m_{l_3} l_3, m_{l_4} l_4, \\ &I_{l_1} + m_{l_1} l_1^2, I_{l_2} + m_{l_2} l_2^2, I_{l_3}, I_{l_4}, m_{m_1}, m_{m_2}, m_{m_3}, m_{m_4}, \\ &I_{m_1}, I_{m_2}, I_{m_3}, I_{m_4}]^T,\end{aligned} \tag{23}$$

the regressor matrix $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}})$, used to obtain an adaptive control law exploiting the linearity of the dynamic model w.r.t. the dynamics parameters $\boldsymbol{\pi}$, is

a $4\times20-$matrix whose non-zero components from $y_{1,1}$ to $y_{4,20}$ are reported below:

$$
\begin{aligned}
y_{1,2} &= a_1^2\ddot{\vartheta}_1\\
y_{1,3} &= (a_1^2+a_2^2)\ddot{\vartheta}_1 + a_2^2\ddot{\vartheta}_2 + a_1a_2(2\ddot{\vartheta}_1+\ddot{\vartheta}_2)\cos(\vartheta_2)\\
&\quad - a_1a_2\dot{\vartheta}_2(2\dot{\vartheta}_1+\dot{\vartheta}_2)\sin(\vartheta_2)\\
y_{1,4} &= (a_1^2+a_2^2)\ddot{\vartheta}_1 + a_2^2\ddot{\vartheta}_2 + a_1a_2(2\ddot{\vartheta}_1+\ddot{\vartheta}_2)\cos(\vartheta_2)\\
&\quad - a_1a_2\dot{\vartheta}_2(2\dot{\vartheta}_2+\dot{\vartheta}_2)\sin(\vartheta_2)\\
y_{1,6} &= a_1(2\ddot{\vartheta}_1+\ddot{\vartheta}_2)\cos(\vartheta_2) - a_1\dot{\vartheta}_2(2\dot{\vartheta}_1+\dot{\vartheta}_2)\sin(\vartheta_2)\\
y_{1,9} &= \ddot{\vartheta}_1\\
y_{1,10} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2\\
y_{1,11} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2\\
y_{1,12} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2+\ddot{\vartheta}_4\\
y_{1,14} &= a_1^2\ddot{\vartheta}_1\\
y_{1,15} &= a_1^2(\ddot{\vartheta}_1+\ddot{\vartheta}_2) + a_1a_2\ddot{\vartheta}_2\cos(\vartheta_2) - a_1a_2\dot{\vartheta}_2^2\sin(\vartheta_2)\\
y_{1,16} &= a_1^2(\ddot{\vartheta}_1+\ddot{\vartheta}_2) + a_1a_2\ddot{\vartheta}_2\cos(\vartheta_2) - a_1a_2\dot{\vartheta}_2^2\sin(\vartheta_2)\\
y_{1,17} &= k_{r_1}^2\ddot{\vartheta}_1\\
y_{1,18} &= \ddot{\vartheta}_1 + k_{r_2}^2\ddot{\vartheta}_2\\
y_{1,19} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2+k_{r_3}^2\ddot{d}_3\\
y_{1,20} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2+k_{r_4}^2\ddot{\vartheta}_4\\
y_{2,3} &= a_2^2(\ddot{\vartheta}_1+\ddot{\vartheta}_2) + a_1a_2\ddot{\vartheta}_1\cos(\vartheta_2) + a_1a_2\dot{\vartheta}_1^2\sin(\vartheta_2)\\
y_{2,4} &= a_2^2(\ddot{\vartheta}_1+\ddot{\vartheta}_2) + a_1a_2\ddot{\vartheta}_1\cos(\vartheta_2) + a_1a_2\dot{\vartheta}_1^2\sin(\vartheta_2).\\
y_{2,6} &= a_1\ddot{\vartheta}_1\cos(\vartheta_2) + a_1\dot{\vartheta}_1^2\sin(\vartheta_2)\\
y_{2,10} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2\\
y_{2,11} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2\\
y_{2,12} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2+\ddot{\vartheta}_4\\
y_{2,15} &= a_1^2(\ddot{\vartheta}_1+\ddot{\vartheta}_2) + a_2^2\ddot{\vartheta}_2 + a_1a_2(\ddot{\vartheta}_1+2\ddot{\vartheta}_2)\cos(\vartheta_2)\\
&\quad - a_1a_2\dot{\vartheta}_2^2\sin(\vartheta_2)\\
y_{2,16} &= a_1^2(\ddot{\vartheta}_1+\ddot{\vartheta}_2) + a_2^2\ddot{\vartheta}_2 + a_1a_2(\ddot{\vartheta}_1+2\ddot{\vartheta}_2)\cos(\vartheta_2)\\
&\quad - a_1a_2\dot{\vartheta}_2^2\sin(\vartheta_2)\\
y_{2,18} &= k_{r_2}\ddot{\vartheta}_1 + k_{r_2}^2\ddot{\vartheta}_2\\
y_{2,19} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2+k_{r_3}\ddot{d}_3\\
y_{2,20} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2+k_{r_4}\ddot{\vartheta}_4\\
y_{3,3} &= g+\ddot{d}_3\\
y_{3,4} &= g+\ddot{d}_3\\
y_{3,15} &= g+\ddot{d}_3\\
y_{3,16} &= g+\ddot{d}_3\\
y_{3,19} &= k_{r_3}^2\ddot{d}_3 + k_{r_3}(\ddot{\vartheta}_1+\ddot{\vartheta}_2)\\
y_{4,12} &= \ddot{\vartheta}_1+\ddot{\vartheta}_2+\ddot{\vartheta}_4\\
y_{4,20} &= k_{r_4}(\ddot{\vartheta}_1+\ddot{\vartheta}_2) + k_{r_4}^2\ddot{\vartheta}_4
\end{aligned}
\tag{24}
$$

B TRAJECTORY PLANNING FUNCTION

The trajectory planning function that has been developed during this project takes as inputs:

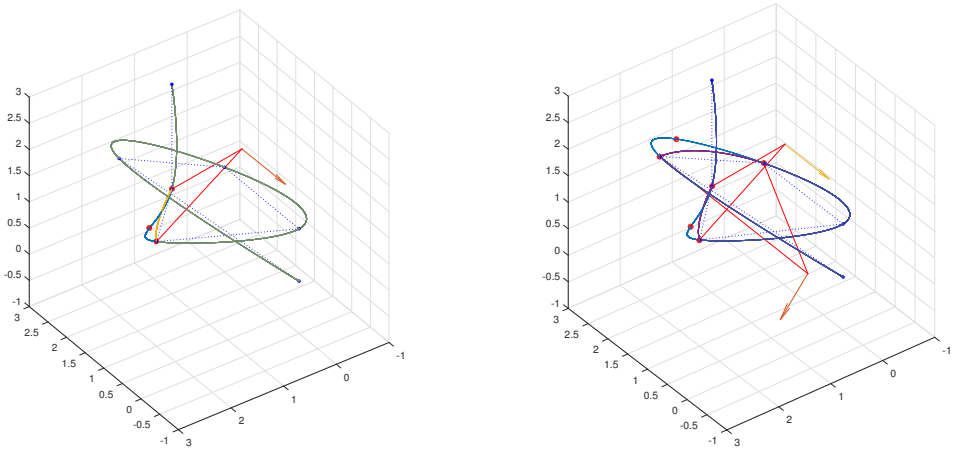
1. the timing vector containing the time instants at which a point has to be reached
2. the coordinates and the type of the points building-up the trajectory (see Table 1)
3. the orientations that the end-effector has to have at the given time instants
4. the curvature and the velocity profile of the trajectory segment (see Table 2).

For a circular path also the plane on which the circumference has to be drawn should be specified. If this is not done, the plane is considered to be the one to which 3 particular points belong:

- the initial point of the segment
- the final point of the segment
- the point that is in the middle of the spline generated considering all the given trajectory points as path points to be interpolated with continuous second derivative.

This way it has been tried to minimize the acceleration during the transitions between segments. Nevertheless, it is always advisable to reduce the speed to 0 whenever there is a discontinuity in the direction of the velocity vector.

In Fig. 20a and Fig. 20b there are two examples of substitution of the spline path with an *optimum* circular path. The center, the radius and the vector related to the plane of the circumference are highlighted in red, orange and yellow.



(a) First example **(b)** Second example

Figure 20: Circular path automatic substitution

REFERENCES

[ÅM09] K. J. Åström and R. M. Murray. *Feedback Systems*. Princeton University Press, Princeton, New Jersey, USA, 2009.

[Ban05] S. Banerjee. *Dynamics for Engineers*. John Wiley & Sons Ltd, Chichester, England, 2005.

[CC10] G. Celentano and L. Celentano. *Fondamenti di Dinamica dei Sistemi*. Edises, Napoli, Italy, 2010.

[CL92] B. B. Choi and C. Lawrence. Inverse kinematics problem in robotics using neural networks. Technical Memo 105869, Lewis Research Center (NASA), Cleveland, Ohio, USA, October 1992.

[Cor11] P. Corke. *Robotics, Vision and Control. Fundamental Algorithms in MATLAB*. Springer-Verlag, Berlin Heidelberg, Germany, 2011.

[Nis13] N. Nise. *Controlli Automatici*. CittàStudiEdizioni, Torino, Italy, 2013.

[PB08] N. Schiavoni P. Bolzern, R. Scattolini. *Fondamenti di Controlli Automatici*. McGraw-Hill, Milano, Italy, 2008.

[Sic90] B. Siciliano. A closed-loop inverse kinematic scheme for on-line joint-based robot control. **8**:231–243, July 1990.

[SK08] B. Siciliano and O. Kathib. *Springer Handbook of Robotics*. Springer-Verlag, Berlin Heidelberg, Germany, 2008.

[SSVO09] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics. Modelling, Planning and Control*. Springer-Verlag, London, England, 2009.

[XN93] Y. Xu and M. Nechyba. Fuzzy inverse kinematic mapping: Rule generation, efficiency, and implementation. Technical Report CMU-RI-TR-93-02, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, January 1993.