UNIVERSITÀ DEGLI STUDI DI GENOVA

DIBRIS

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY,
BIOENGINEERING, ROBOTICS AND SYSTEM ENGINEERING

MODELLING AND CONTROL OF MANIPULATORS

Third Assignment

Jacobian Matrices and Inverse Kinematics

Author:

Miceli Carmine

Student ID:

s5626492

Professors:

Giovanni Indiveri
Enrico Simetti
Giorgio Cannata

Tutors:

Andrea Tiranti
Francesco Giovinazzo

December 14, 2022

Contents

Mathematical expression	Definition	MATLAB expression
$< w >$	World Coordinate Frame	w
${}^{a}_{b}R$	Rotation matrix of frame $< b >$ with respect to frame $< a >$	aRb
${}^{a}_{b}T$	Transformation matrix of frame $< b >$ with respect to frame $< a >$	aTb

Table 1: Nomenclature Table

1 Introduction

The third assignment of Modelling and Control of Manipulators focuses on the definition of the Jacobian matrices for a robotic manipulator and the computation of its inverse kinematics.

2 Exercise 1

Given the CAD model of the robotic manipulator from the previous assignment, it is required to compute the Jacobian matrix for some given joint configurations.
It is recalled that the Jacobian is a 6*xn* matrix, where the number of rows take into account the possible rotations and translations around the x-y-z axes; whereas **n** stands for the number of joints present in the manipulator.
It is used to correlate the joints velocities to the linear and angular velocity of the end-effector with the following equation:

$$\begin{bmatrix} \boldsymbol{\omega_{n/0}} \\ \boldsymbol{v_{n/0}} \end{bmatrix} = \begin{bmatrix} J^A_{n/0} \\ J^L_{n/0} \end{bmatrix} \dot{\mathbf{q}}$$

In this case it was assumed that no rigid tool was attached to the end-effector and therefore the notation $n/0$ was used and it is equivalent to $e/0$, in the subsequent sections additional explanations would be provided for the case of a manipulator involving a rigid tool.
Therefore, we could further develop the expressions of the Jacobian to understand how they are computed by observing the following:

$$\begin{bmatrix} J^A_{n/0} \\ J^L_{n/0} \end{bmatrix} = \begin{bmatrix} J^A_{n/1} & J^A_{n/2} & \dots & J^A_{n/n} \\ J^L_{n/1} & J^L_{n/2} & \dots & J^L_{n/n} \end{bmatrix}$$

where each term results to be:

$$\boldsymbol{J^A_{n/j}} = \begin{cases} \boldsymbol{{}^b k_j}, & \text{if } \Gamma_j = RJ \\ 0, & \text{if } \Gamma_j = PJ \end{cases}$$

The *angular* component of the Jacobian is equal to zero for a prismatic joint, whereas for a revolute joint is equal to the axis of rotation retrieved by solving the inverse angle-axis problem for the rotational matrix ${}^b_jR$.

$$\boldsymbol{J^L_{n/j}} = \begin{cases} \boldsymbol{{}^b k_j} \times \boldsymbol{{}^b r_{n/j}}, & \text{if } \Gamma_j = RJ \\ \boldsymbol{{}^b k_j}, & \text{if } \Gamma_j = PJ \end{cases}$$

The *linear* component of the Jacobian is equal to the z-axis component of the rotational matrix ${}^b_jR$ for a prismatic joint, whereas for a revolute joint is equal to the cross product between the axis of rotation retrieved by solving the inverse angle-axis problem for the rotational matrix ${}^b_jR$ and the distance between the *n-th* joint and the *j-th* joint.

2.1 Q1.1

For the joint configuration $\mathbf{q}_1 = [1.3, 1.3, 1.3, 1.3, 1.3, 1.3, 1.3]$, the Jacobian matrix obtained is the following:

$$J_1 = \begin{bmatrix} 0 & 0 & 0 & 0.9438 & 0.7960 & 0.2973 & 0.7092 \\ 0 & 0.9660 & -0.1400 & -0.0977 & 0.4915 & 0.2381 & 0.3005 \\ 1 & 0.2584 & 0.9902 & -0.3158 & 0.3532 & 0.9246 & 0.6377 \\ 95.5649 & 70.9611 & 23.7072 & 107.6084 & 42.8244 & -106.7185 & 0 \\ 300.9868 & 255.6091 & 309.0003 & -193.3893 & -22.0533 & 98.3882 & 0 \\ 0 & 11.5687 & 84.4178 & 406.2391 & 139.3330 & 48.3729 & 0 \end{bmatrix}$$

It is recalled that the joint configuration is strictly related to the computation of the Jacobian matrix because it is involved in the determination of the transformation matrix ${}^b_jT$, from which the relevant quantities previously explained, such as the rotation axis and the distance, are retrieved.

The role of the Jacobian is to map velocities from the joints space to the end-effector space, we could further analyze it by observing the results obtained.
For example, the entries 1,1 - 1,2 - 1,3 are zero and this means that the first three joints cannot provide any rotation around the x-axis for the end-effector; the entry 6,1 is zero and this means that the first joint cannot provide any linear velocity along the z-axis for the end-effector.
These observations made are general and hold for the subsequent joints configuration as well.

2.2 Q1.2

For the joint configuration $\mathbf{q}_2 = [1.3, 0.4, 0.1, 0, 0.5, 1.1, 0]$, the Jacobian matrix obtained is the following:

$$J_2 = \begin{bmatrix} 0 & -0.3630 & 0.4049 & 0.7319 & 0.8249 & 0.8562 & 0.5351 \\ 0 & 0.7515 & -0.5919 & 0.3705 & 0.3985 & 0.2037 & 0.1097 \\ 1 & 0.5508 & 0.6969 & -0.5719 & 0.4008 & 0.4748 & 0.8376 \\ -152.7221 & 59.3837 & -256.4934 & 25.8352 & 115.4830 & -0.8959 & 0 \\ 82.4301 & 213.9062 & 86.7958 & -89.5885 & -53.5197 & -104.2584 & 0 \\ 0 & -169.2066 & 35.5283 & -495.1143 & -48.9074 & 111.9749 & 0 \end{bmatrix}$$

2.3 Q1.3

For the joint configuration $\mathbf{q}_3 = [1.3, 0.1, 0.1, 1, 0.2, 0.3, 1.3]$, the Jacobian matrix obtained is the following:

$$J_3 = \begin{bmatrix} 0 & -0.4485 & 0.4599 & 0.7595 & -0.8258 & 0.5641 & -0.9861 \\ 0 & 0.6556 & -0.6723 & 0.0799 & -0.5257 & 0.8083 & 0.0815 \\ 1 & 0.6075 & 0.5801 & -0.6456 & -0.2042 & 0.1686 & -0.1449 \\ 113.5256 & -79.0058 & 294.4988 & -96.5365 & 40.9293 & 45.4622 & 0 \\ 9.5643 & -284.5869 & -77.6563 & -475.8164 & -19.2100 & 98.8312 & 0 \\ 0 & 106.8301 & 39.3860 & -294.5973 & 0.3514 & 107.5852 & 0 \end{bmatrix}$$

2.4 Q1.4

For the joint configuration $\mathbf{q}_4 = [2, 2, 2, 2, 2, 2, 2]$, the Jacobian matrix obtained is the following:

$$J_4 = \begin{bmatrix} 0 & 0 & 0 & -0.7389 & 0.1928 & -0.0406 & -0.1842 \\ 0 & -0.9232 & 0.2331 & -0.2508 & 0.6090 & -0.1049 & -0.9825 \\ 1 & 0.3842 & 0.9725 & -0.6254 & 0.7694 & 0.9937 & 0.0266 \\ -419.72 & -203.91 & -196.23 & -186.43 & -13.32 & -21.26 & 0 \\ -95.5690 & 445.5547 & -2.0435 & -31.6789 & -28.0081 & -147.9558 & 0 \\ 0 & -421.4253 & -36.5237 & 307.5100 & 135.6206 & -32.6460 & 0 \end{bmatrix}$$

3 Exercise 2

Given the model of a Panda robot by Franka Emika, it is required to solve the inverse kinematic problem in order to compute the position for each joint in order to make the end-effector reach a specific point in the space.

3.1 Q2.1

The first thing to compute is the cartesian error between the robot end-effector frame ${}^b_eT$ and the goal frame ${}^b_{ge}T$, which has been defined knowing that:

- The goal position with respect to the base frame is ${}^bO_g = [0.6, 0.4, 0.4]^T$
- The goal frame is rotated of $\theta = -\pi/4$ around the z-axis of the robot end-effector initial configuration.

The matrix ${}^b_eT$ obtained with the built-in function *getTransfrom()* is the following:

$${}^b_eT = \begin{bmatrix} 0.729 & -0.682 & -0.041 & 0.314 \\ -0.683 & -0.729 & -0.0162 & -0.004 \\ -0.0189 & 0.0400 & -0.999 & 0.693 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Concerning the matrix ${}^b_{ge}T$, the distance was given whereas the rotational component was computed as the product ${}^b_eR\,{}^e_{ge}R$, where the latter is an elementary rotation around the z-axis.
Overall, the resulting matrix is the following:

$${}^b_{ge}T = \begin{bmatrix} 0.9986 & 0.0335 & -0.0412 & 0.6 \\ 0.0329 & -0.999 & -0.0163 & 0.4 \\ -0.0417 & 0.0149 & -0.9991 & 0.4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The general procedure for the algorithm that solves the inverse kinematic problem is to compute at each iteration, i.e. at each time the joints change their positions, the new values for ${}^b_eT$.
From this matrix, it is possible to compute the cartesian error with the following expression:

$$ {}^e_{ge}R = {}^b_e R^T \, {}^b_{ge}R $$

This is needed because what the algorithm tries to achieve is to have ${}^e_{ge}R$ equal to the identity matrix and the distance between the end-effector and the goal corresponding to zero.

3.2 Q2.2

Starting from the rotation matrix ${}^e_{ge}R$, the equivalent angle-axis representation ***h*** - θ is obtained in order to compute the desired angular and linear reference velocities of the end-effector with respect to the base:

$$ {}^b\nu^*_{e/0} = \alpha \cdot \begin{bmatrix} \omega^*_{e/0} \\ v^*_{e/0} \end{bmatrix} $$

where $\alpha = 0.2$ is the gain and the expressions for the velocities are composed of the position and orientation error:

$$ v^*_{e/0} = {}^b_{ge} r - {}^b_e r $$

$$ \omega^*_{e/0} = h\theta $$

3.3 Q2.3

At this point, it is possible to compute the desired joint velocities with the solution obtained for the inverse kinematic problem for a Jacobian J_{mxn}, where $n > m$, i.e. the number of joints is greater than the task space dimension:

$$ \dot{q}^* = J^T (JJ^T)^{-1} \, {}^b\nu^*_{e/0} $$

in this case *J* stands for the Jacobian matrix ${}^b_eJ$ computed at each iteration.

3.4 Q2.4

The robot motion is then simulated by implementing the function *"KinematicSimulation()"*, which updates each new joint position:

$$ q_i = q_{i-1} + \dot{q}_i t $$

where *t* is a set time interval between each iteration and $\dot{q}_i$ is the desired joint velocity previously obtained.

The initial configuration of the robot can be observed in Figure 1.

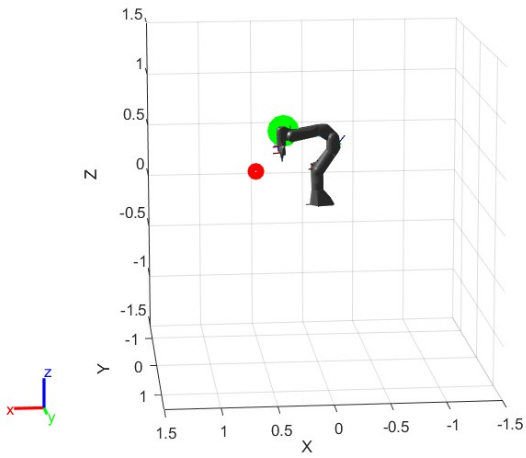

Figure 1: Panda robot initial configuration

The final configuration of the robot that reached the goal, with the translations and rotations involved highlighted in green, can be observed in Figure 2.

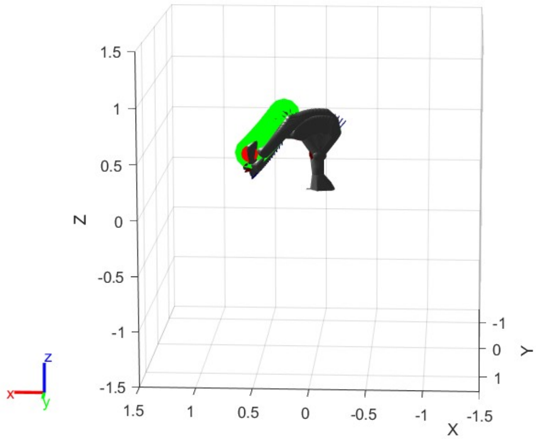

Figure 2: Panda robot final configuration

4 Exercise 3

In case we consider a tool frame rigidly attached to the robot end-effector according to the following transformation matrix:

$$ {}^{e}_{t}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

The resulting matrix between the tool with respect to the base is the following:

$$ {}^{b}_{t}T = \begin{bmatrix} 0.729 & -0.682 & -0.041 & 0.305 \\ -0.683 & -0.729 & -0.016 & -0.008 \\ -0.019 & 0.040 & -0.999 & 0.493 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

4.1 Q3.1

It is necessary to compute the cartesian error between the robot tool frame ${}^{b}_{t}T$ and the goal frame ${}^{b}_{gt}T$, which has been defined knowing that:

- The goal position with respect to the base frame is ${}^{b}O_{g} = [0.6, 0.4, 0.4]^{T}$
- The goal frame is rotated of $\theta = -\pi/4$ around the z-axis of the robot tool frame initial configuration.

As it was done in the previous section, the resulting matrix is:

$$ {}^{b}_{gt}T = \begin{bmatrix} 0.998 & 0.033 & -0.041 & 0.6 \\ 0.032 & -0.999 & -0.016 & 0.4 \\ -0.041 & 0.014 & -0.999 & 0.4 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

From this matrix, it is possible to compute the cartesian error with the following expression:

$$ {}^{t}_{gt}R = {}^{b}_{t}R^{T}\,{}^{b}_{gt}R $$

This is needed because what the algorithm tries to achieve is to have ${}^{t}_{gt}R$ equal to the identity matrix and the distance between the tool and the goal corresponding to zero.

4.2 Q3.2

Since there is a rigid tool attached to the end-effector, in order to compute the desired angular and linear reference velocities of the tool with respect to the base, i.e. :

$$^{b}\nu^{*}_{t/0} = \alpha \cdot \begin{bmatrix} \omega^{*}_{t/0} \\ v^{*}_{t/0} \end{bmatrix}$$

where $\alpha = 0.2$ is the gain and the expressions for the velocities are the same as the previous case but the distance and the rotation matrix are from the rigid tool *t* w.r.t the base, it is necessary to take into account an additional term called rigid body Jacobian matrix $^{b}S_{t/e}$ that has the followig structure:

$$^{b}S_{t/e} = \begin{bmatrix} I_3 & 0_3 \\ [^{b}r_{t/e}]^T & I_3 \end{bmatrix}$$

In the case of study, the matrix obtained is:

$$^{b}S_{t/e} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -0.2 & 0 & 1 & 0 & 0 \\ 0.2 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

4.3 Q3.3

It is possible to compute the desired joint velocities with the solution obtained for the inverse kinematic problem for a Jacobian J_{mxn}, where $n > m$, i.e. the number of joints is greater than the task space dimension:

$$\dot{q}^{*} = J^T (JJ^T)^{-1} {}^{b}\nu^{*}_{e/0}$$

but in this case *J* stands for the Jacobian matrix $^{b}_{t}J$ computed at each iteration, where $^{b}_{t}J = {}^{b}S_{t/e}\,{}^{b}_{e}J$ and the rigid body Jacobian has been added to move from the end-effector to the rigid tool attached.

4.4 Q3.4 - Q3.5

The robot motion is then simulated with the function *"KinematicSimulation()"*, which updates each new joint position:

$$q_i = q_{i-1} + \dot{q}_i t$$

where *t* is a set time interval between each iteration and $\dot{q}_i$ is the desired joint velocity previously obtained. The initial configuration of the robot can be observed in Figure 3.

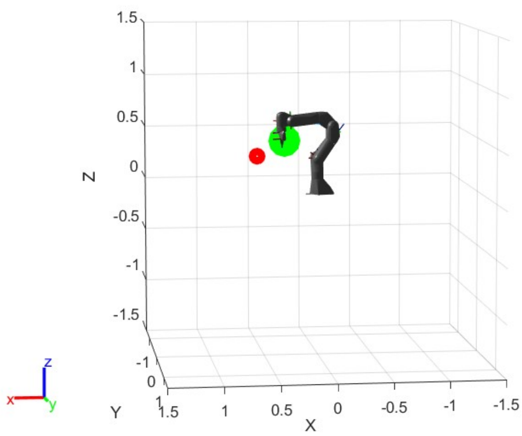

Figure 3: Panda robot initial configuration with tool attached

The final configuration of the robot that reached the goal can be observed in Figure 4.

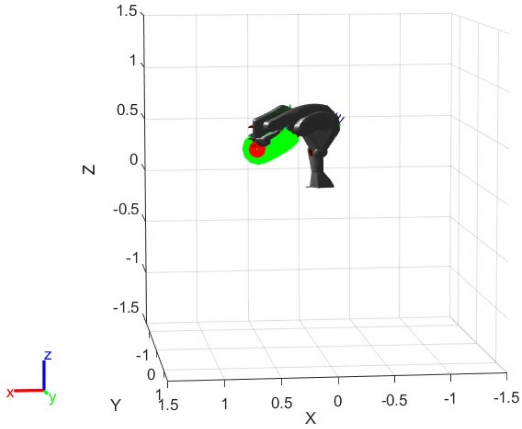

Figure 4: Panda robot final configuration with tool attached

The difference with respect to exercise 2 is related to the presence of the rigid body Jacobian matrix ${}^{b}S_{t/e}$, which is an additional term that must be carefully taken into account when computing the needed Jacobian to map correctly the tool velocities to the joints velocities.

4.5 Q3.6

If we consider a new tool goal, the procedure is the same as the previous sections and knowing that the transformation matrix of the goal with respect to the robot base is:

$$ {}^{b}_{gt}T = \begin{bmatrix} 0.7071 & 0 & -0.7071 & 0.6 \\ -0.7071 & 0 & -0.7071 & 0.5 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

The initial configuration of the robot can be observed in Figure 5.

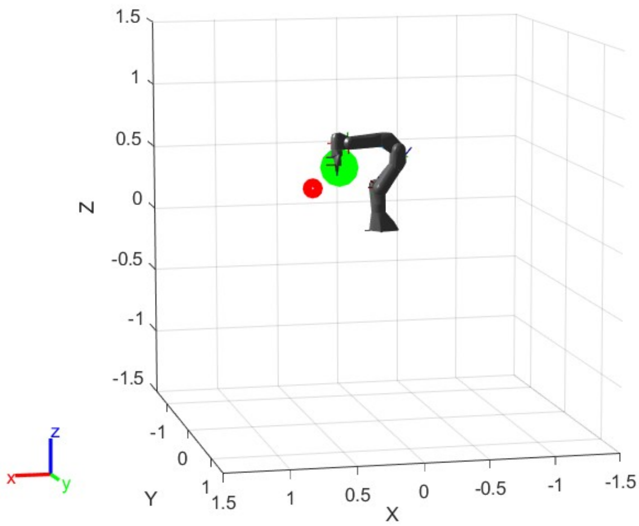

Figure 5: Panda robot initial configuration with tool attached

The final configuration of the robot that reached the goal can be observed in Figure 6.

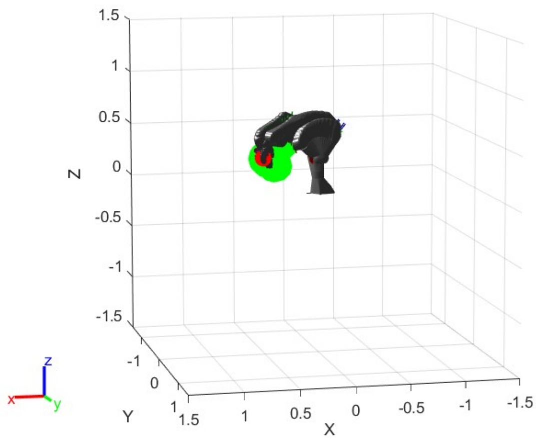

Figure 6: Panda robot final configuration with tool attached

5 Appendix

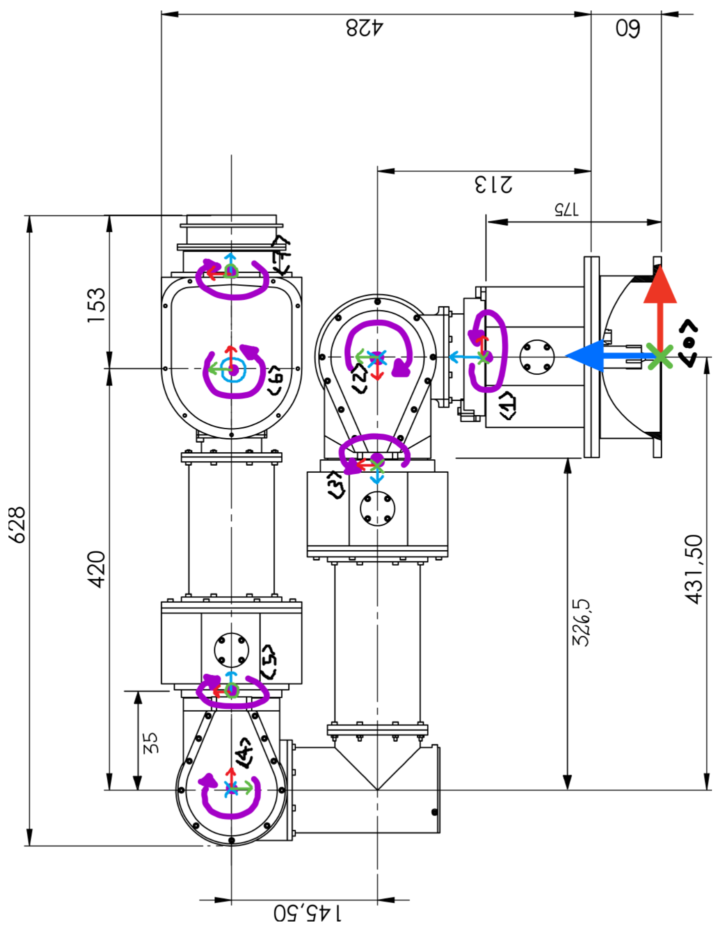

Figure 7: CAD model of the robot