FINAL PROJECT ASSIGNMENT:
Giraff robot for question handling in conferences

Nicola Maestri

0 Project Assignment

The final project involves designing a robotic system, which aims to automate the task of handing a microphone to random people in a small theater/conference room. The robot is positioned in the middle of the room and is attached to the ceiling. The room has a height of 4 meters, and the robot should be capable of reaching locations up to 1 meter high within a 5x12 meters area.

The robot is required to have 5 degrees of freedom. The base consists of a spherical joint with two intersecting revolute joints. Additionally, there is a prismatic joint capable of extending significantly, as well as two revolute joints to properly orient the microphone (not necessarily with intersecting axes).

The objective is to enable the robot to position the microphone at any point within the 5x5 conference room. The microphone should be oriented with a pitch angle of 30 degrees relative to the horizontal axis, allowing people to comfortably speak into the microphone. Therefore, the task involves a 4-dimensional space for locating the microphone accurately.

The project can be approached through the following incremental steps:

1. Construct the URDF model of the robot, selecting appropriate link lengths and arranging frames in a suitable manner.
2. Compute the forward kinematics (position/orientation) and differential kinematics (Jacobian) of the end-effector.
3. Use Pinocchio library's RNEA native function to create a simulator of the motion.
4. Plan a polynomial trajectory (in the task space) to move from a coming configuration q_{home} to a given end-effector configuration-orientation $p_{des} + \Theta_{des}$.
5. Write an inverse-dynamics (computed torque) control action in the task space to linearize the system and achieve tracking of the task.
6. Set the PD gains of the Cartesian controller implemented on the linearized system to achieve a settling time of 7s without overshoot.

7. In the null-space of the task minimize the distance with respect to a given configuration q_0 of your choice.

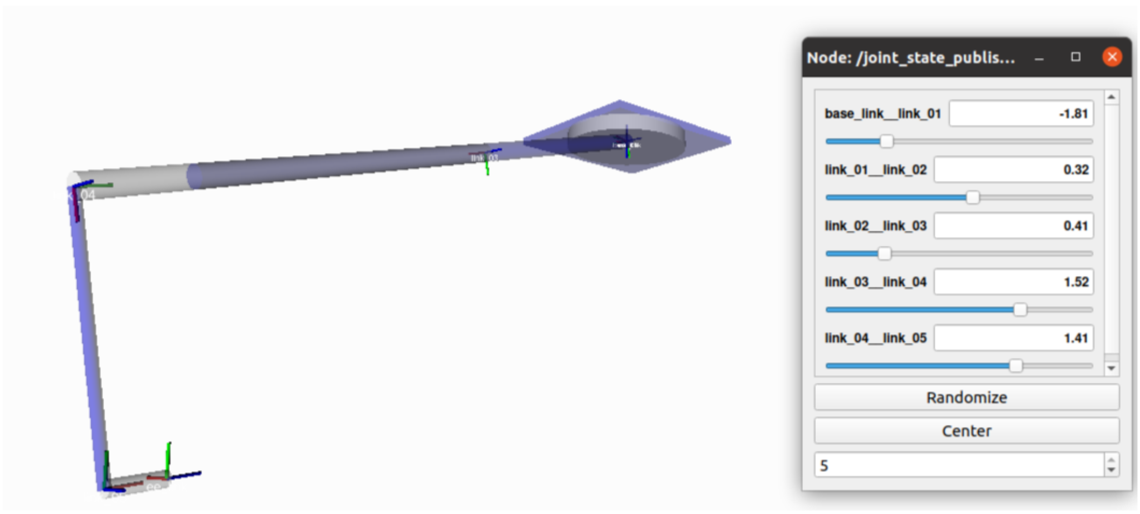
8. Simulate the robot to reach the location $p_{des} = [1, 2, 1]$ from the homing configuration $q_{home} = [0, 0, 0, 0]$.

1 Robot URDF: description and visualization

The robot is a chain of 6 links connected by means of 5 joints (R - R - P - R - R):

- base-link: a 1.0 x 1.0 x 0.05 box of mass 1 kg
- link-1: a cylinder (length = 0.1, radius = 0.40) of mass = 1 kg
- link-2: a cylinder (length = 3.0, radius = 0.05) of mass = 1 kg
- link-3: a cylinder (length = 2.0, radius = 0.05) of mass = 1 kg
- link-4: a cylinder (length = 1.0, radius = 0.02) of mass = 1 kg
- link-1: a cylinder (length = 0.2, radius = 0.03) of mass = 1 kg

The joints are all revolute except the one between link-2 and link-3 which is prismatic. Frames are placed on the links such that the principal direction is along the X-axis and the translation or rotation is along the Z-axis.
Inertia matrices and other important features of the structure are detailed in the URDF file. Figure 1 shows a visualization of the manipulator.


Node: /joint_state_publis...

base_link__link_01 -1.81
link_01__link_02 0.32
link_02__link_03 0.41
link_03__link_04 1.52
link_04__link_05 1.41

Figure 1: Giraff Robot

2 Robot K-inematics

Forward Kinematics

In this section, we derive the homogeneous transformation matrices that map from the world frame to other coordinate frames.

$$
{}_W T_0 = \begin{bmatrix} {}_W R_0 & {}_W P_{W0} \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}_0 T_1 = \begin{bmatrix} {}_0 R_1 & {}_0 P_{01} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} R(q_1) & 0 \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -0.075 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(q_1) & -sin(q_1) & 0 & 0 \\ sin(q_1) & cos(q_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}_1 T_2 = \begin{bmatrix} {}_1 R_2 & {}_1 P_{12} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} R(q_2) & 0 \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(q_2) & -sin(q_2) & 0 & 0 \\ sin(q_2) & cos(q_2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}_2 T_3 = \begin{bmatrix} {}_2 R_3 & {}_2 P_{23} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} I_3 & d(q_3) \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & q_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}_3 T_4 = \begin{bmatrix} {}_3 R_4 & {}_3 P_{34} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} R(q_4) & 0 \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(q_4) & -sin(q_4) & 0 & 0 \\ sin(q_4) & cos(q_4) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}_4 T_5 = \begin{bmatrix} {}_4 R_5 & {}_4 P_{45} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} R(q_5) & 0 \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(q_5) & -sin(q_5) & 0 & 0 \\ sin(q_5) & cos(q_5) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$_5T_{ee} = \begin{bmatrix} _5R_{ee} & _5P_{5ee} \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0.2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$_WT_{ee} = {_WT_0} \cdot {_0T_1} \cdot {_1T_2} \cdot {_2T_3} \cdot {_3T_4} \cdot {_4T_5} \cdot {_5T_{ee}}$$

Differential Kinematics

Now we can compute the differential kinematics as follow:

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_{P1} & \dots & \mathbf{J}_{Pi} \\ \mathbf{J}_{O1} & \dots & \mathbf{J}_{Oi} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{J}_{P1} \\ \mathbf{J}_{O1} \end{bmatrix} = \begin{bmatrix} _Wz_1 \times {_WP_{1ee}} \\ _Wz_1 \end{bmatrix} \qquad \begin{bmatrix} \mathbf{J}_{P2} \\ \mathbf{J}_{O2} \end{bmatrix} = \begin{bmatrix} _Wz_2 \times {_WP_{2ee}} \\ _Wz_2 \end{bmatrix} \qquad \begin{bmatrix} \mathbf{J}_{P3} \\ \mathbf{J}_{O3} \end{bmatrix} = \begin{bmatrix} _Wx_3 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{J}_{P4} \\ \mathbf{J}_{O4} \end{bmatrix} = \begin{bmatrix} _Wz_4 \times {_WP_{4ee}} \\ _Wz_4 \end{bmatrix} \qquad \begin{bmatrix} \mathbf{J}_{P5} \\ \mathbf{J}_{O5} \end{bmatrix} = \begin{bmatrix} _Wz_5 \times {_WP_{5ee}} \\ _Wz_5 \end{bmatrix}$$

From the Geometric Jacobian, it is possible to compute the analytic Jacobian through the formula $\mathbf{J}_a = \mathbf{T}_a\mathbf{J}$.
Here I report an example at $q_0 = [0, 0, 0, 0, 0]$.

$$J_G = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -4.2 & 0 & 0 & 0 & 0 \\ 0 & -4.2 & 0 & -1.2 & -0.2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$T_a = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad J_A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -4.2 & 0 & 0 & 0 & 0 \\ 0 & -4.2 & 0 & -1.2 & -0.2 \\ 0 & -1 & 0 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Inverse Kinematics

We solved the inverse kinematic problem using a numerical approach by implementing an iterative algorithm. To verify the accuracy of the solution, we computed the error between the desired end-effector position and the obtained position. Although our implemented solution is capable of finding a valid configuration, it does not take into account any limitations of the joints. For instance, the prismatic joint can move without any constraints, which is unrealistic. Furthermore, even the revolute joint could achieve a more optimal configuration by introducing a postural task.

Polinomial Trajectory

I calculate the trajectory using a fifth-degree polynomial which interpolates between the initial and final joint configuration. However, the resulting solution is not optimal, because it does not give any guarantees about the intermediate configurations.
To address this issue, I tried to implement the trajectory directly in the task space instead. The idea was to compute a straight trajectory in the task space between the start position and the desired position and, subsequently, to smoothly move the end effector along this trajectory.

3 Robot Dynamic

I used the Pinocchio implementation of the Recursive Newton-Euler Algorithm (RNEA) to calculate the Dynamics term necessary for the forward dynamics simulation. Assuming no torques or forces were applied to the joints, the simulation shows the robot arm naturally descending under the influence of gravity.

I have not accounted for any constraints on the prismatic joint, however, for a more realistic representation, it would be advisable to introduce a constraint reaction once the joint reaches its limit.

4 Robot Controll

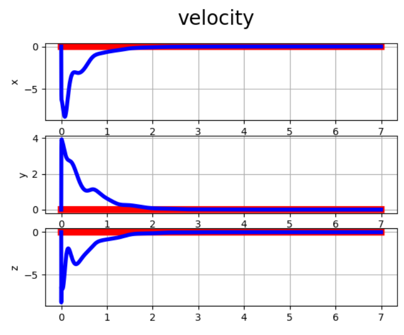
In this section, I design a motion controller in the task space following a centralized approach. We assume a free motion of the manipulator from an initial configuration to a fixed position trying to keep the whole configuration as close as possible to a postural task $q_0 = [\ 0,\ 0.523,\ 0.5,\ 1.57,\ 1.57]$.
In order to linearize the dynamic of the system we use a feedback approach:

$$\begin{aligned}
&F^d = \ddot{p}^d + K_x(p^d - p) + D_x(\dot{p}^d - \dot{p}) \\
&\Lambda = (JM^{-1}J^T)^{-1} \\
&\mu(q,\dot{q}) = J^{T\#}h - \Lambda\dot{J}\dot{q} \\
&\tau = J^T\left(\Lambda F^d + \mu(q,\dot{q})\right)\ [\,+\ \tau_{null}\ 0]
\end{aligned}$$

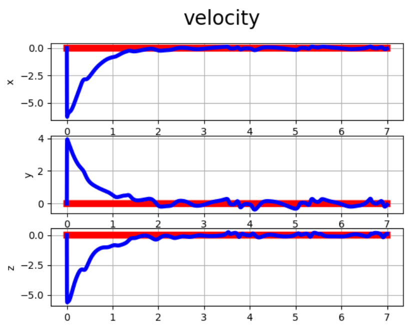
We have to use a pseudo-inverse matrix $J^{T\#}$, because J is rectangular.
K_x e D_x are diagonal matrixes therefore the system is completely decoupled. The time evolution is governed by the roots of the polynomial $s^2 + K_xS + D_x$, hence setting properly the parameters can speed up convergence but also introduce instabilities.
In my model, I keep K_p big enough to ensure convergence and adjust K_d in order to avoid overshoots.
The robot is redundant, therefore this postural task helps to keep stable the movement of the robot arm.

$$\begin{aligned}
&\tau_0 = K_q(q_0 - q) - K_{\dot{q}}\dot{q} \\
&\tau_{null} = [I\ \ -\ \ J^TJ^{T\#}]\ \tau_0
\end{aligned}$$

Here we can see the effect on the stability of the end effector, however it is much more evident at the joint level.



ee-velocity
with Postural-Task

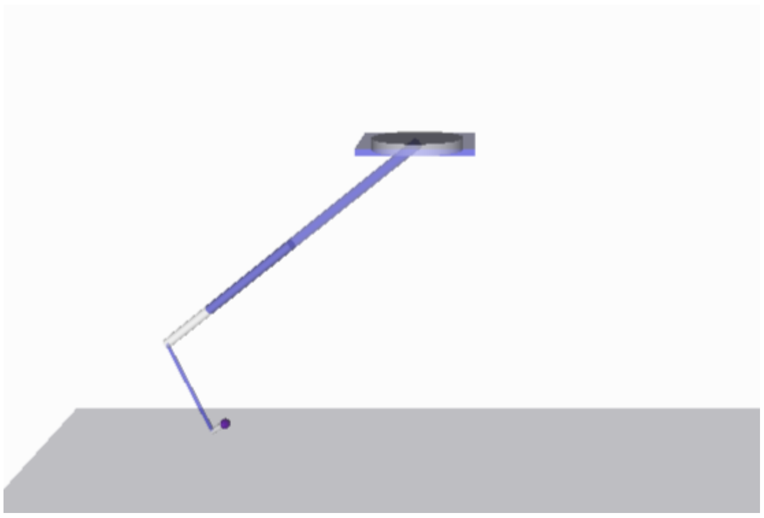


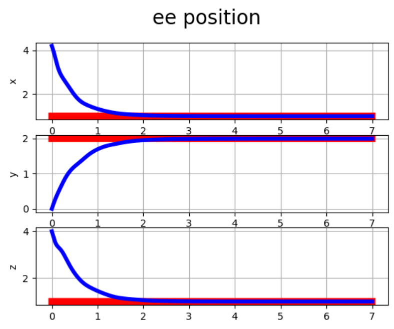
ee-velocity
without Postural-Task

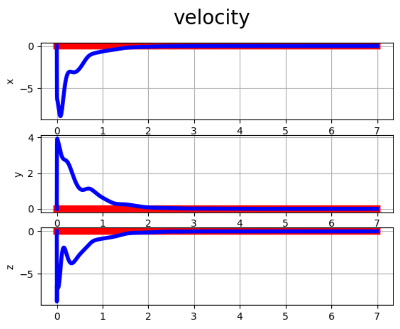
We now report some visualization of the manipulator in action with different tasks.

Reach a constant reference

Reach a constant reference $q_{des} = [1, 2, 1]$

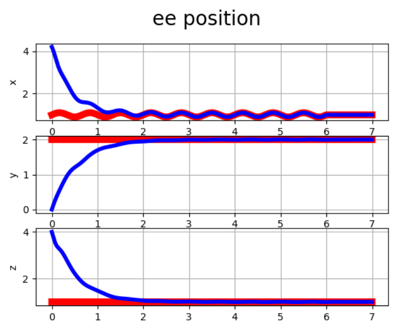


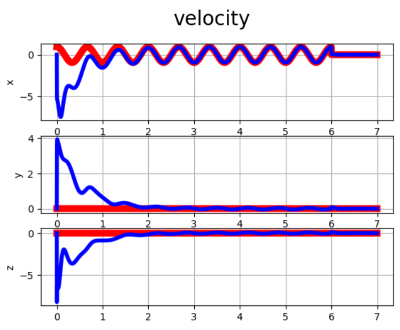




Track a sinusoidal trajectory

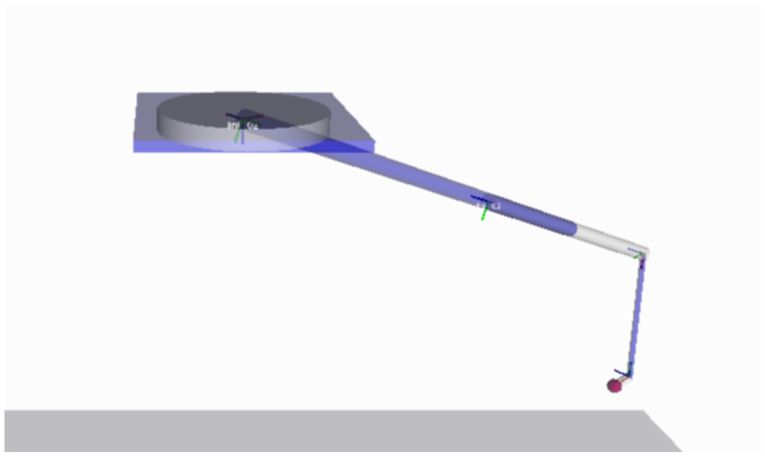
The robot should track a sinusoidal trajectory which at the end stops in a certain position.

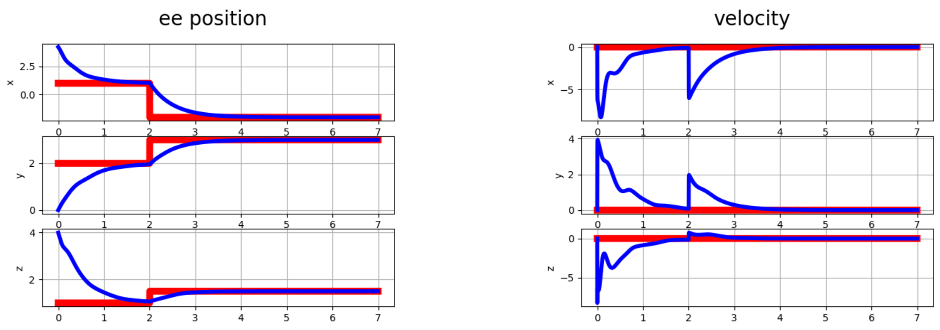




Track a step-constant reference

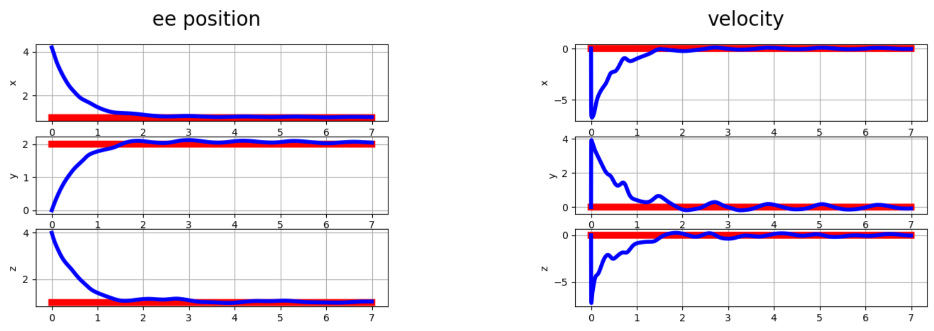
The robot should track a step-constant trajectory readjusting the joints once the reference is changed.





Relevance of parameter tuning

We close this part by putting in evidence the importance of parameter tuning to ensure stability. Adjusting properly the parameters of the system can avoid many issues such as overshooting. The figures below show the unstable behaviour of the end effector in the case of rough tuning.


ee position
velocity
x
y
z