5, May, 2022

Fabio Conti
Mat.4693053

Robot Dynamics and Control

First assignment: report

Introduction

This assignment covers the basics of force/torque equilibrium for a robotic manipulator, a fundamental concept for **dynamic control**. The main relation used in the assignment is the following:

$$\boldsymbol{\tau_{eq}} = -\boldsymbol{J}^{T}\boldsymbol{W}_{ext}$$

$\boldsymbol{W}_{ext}$: cartesian wrench (torque, force) applied to an arbitrary position on the robot body.
$\boldsymbol{J}^{T}$: transposed Jacobian matrix of the robot's links.
$\boldsymbol{\tau_{eq}}$: vector of the generalized actuator forces.

The aim of the assignment was to find the $\boldsymbol{\tau_{eq}}$ of 10 different configurations of a simple **planar manipulator**. The basic parameter collection needed for dealing with the robot simulation had to be extracted from the manipulator's CAD model as a preliminary step of the assignment.

1. CAD model

The instructions for the making of the CAD model represented a simple 3D manipulator composed by two links connected to each other and onto a base through motors. The extrusion, assembly, and exporting of all the components needed to build the robot's computer-aided design were delt on Autodesk Software.

1.1 CAD Assembly

The Main part of the CAD was developed on the Autodesk FUSION program. The whole Arm structure was composed by the following 3D bodies:

- A **base** that anchors the manipulator to the ground.
- The **motor 1** connecting the first link to the base.
- The **first link**.
- The **motor 2**, geometrically identical to the first one, connecting the first link to the second
- The **second link.**

Each component was extruded in a different *Fusion project* and later assembled in an assembly document.
In the **assembly** project took care of constraining the motion of each component using rigid and rotational constraints.
The constraints applied for the construction of the manipulator are the following:

- The **motor 1** is linked to the **base** through a rotational joint which will let the motor spin within 0º to 180º from the original position.
- The **link 1** il connected to the **motor 1** through a rigid constrain to rotate in solidarity to the motor 1 with respect to the base.
- The **motor 2** is connected to **link 1** with a rotational joint just like the motor 1 is linked to the base. Though, differently from the first one, this rotational joint is constrained to rotate from -90º to 90º degrees with respect to the first link in the initial configuration.
- The **link 2** is attached to **motor 2** through a rigid constraint. This joint allows the second link to rotate in solidarity to the motor 2 with respect to the first link.

Measurements, parameters, and materials of each component were listed in the assignment instructions. The following picture shows the final assembly:

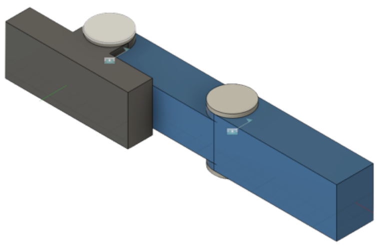


Fig.1: 3D manipulator in the initial configuration.

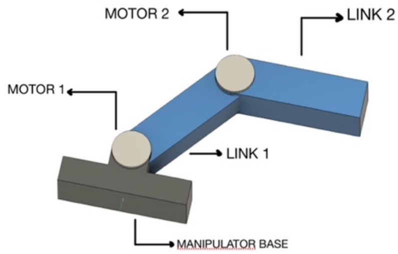


Fig.2: Random configuration of the robot.

Every component of the entire model was extruded following the **Denavit-Hatemberg** rules. This set of conventions is an efficient criterion for assigning the reference frame relative to each link of a certain manipulator.
In the case of our two configuration (R-R and R-P) a simplified D-H convention states the following:

- The "n" number of links always must follow the "n" number of joints.
- An origin frame "O" (base frame) must be defined.
- The reference frame of the i-th link is in such way that the **origin** of the frame is located along all the motion axis of the corresponding i-th joint. Moreover, the origin of the frame could be placed in correspondence of the center of the i-th joint. The reference frame **z-axis** must be located along the axis of the joint's motion (/actuation).

In the CAD model the following rules were applied:

- For the *revolute* joint:
  1. The **z-axis** of the links' reference frame represents its axis of rotation.
  2. The **x-axis** of the revolute joint is placed along the direction of the following link.
  3. The **y-axis** must be determined starting from the positioning of the previous two according to the right-hand rule.

- For the *Prismatic* joint:
  1. The **z-axis** of the prismatic joint's reference frame must be placed along the direction of displacement.

The following picture represents all the relative and absolute reference frames assigned following the just mentioned rules:

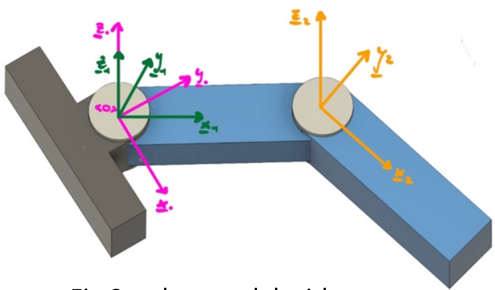


Fig.3: robot model with two rotational joints.

1.2 Parameter collection

To extract all the parameters from the manipulator model it is necessary to go through the following steps:

- Export the Assembly file from Fusion with the extension *.iam*. This format will make the file recognizable by the **INVENTOR** Autodesk software. This program will be capable of extracting all the data from the model and transfer it to **Matlab**.
- Also, the latest version of Matlab should be installed to proceed with the data collection.
- To export the data form Inventor in a readable format for Matlab, The *Simscape Multibody* plugin should be installed on inventor from Matlab.
- Once installed the plugin, the Project can be exported in an XML format on the Matlab application ready to be opened.
- With the Matlab application opened onto the directory of the newly exported file running the *smimport(<name_of_the_file>)* command should generate two new script inside the same directory. A Symulink Diagram and a *.m* script.
- Running the second one will generate a *SmiData* file.

This *SmiData* file will contain all the data related to the mass, inertia, positioning in the space, etc.. of the CAD model. This Data can later be processed by the Matlab software to calculate the vector of generalized actuator forces for each of the configurations proposed.

2. Theory behind the solution of the given problems

In the proposed conditions, the basic manipulator is subjected to different kind of forces and torques applied to different points of the robot's body. The aim of each exercise is to find the conditions the robot control actions must satisfy to keep the robot in an equilibrium condition.
Keeping a robot in a state of equilibrium means that each of its components are in a state of equilibrium. This means that the position and orientation of every body of the assembly will remain constant in time.
The **Newton Euler** equation states the following:

$$\begin{cases} m_i\, \dot{\boldsymbol{v}}_{ci} = \boldsymbol{F}_i \\ \boldsymbol{I}_{ci}\dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \boldsymbol{I}_{ci}\boldsymbol{\omega}_i = \boldsymbol{M}_i \end{cases} \qquad i = 1 \div n$$

- Where $\boldsymbol{F}_i$ represents the resultant of all the forces acting on body *i*.

This formulation represents the N-E equation for every body of the kinematic chain. Assuming that the right hand side terms and the initial position of the robot are known, it is possible to integrate those differential equations and describe the motion of the kinematic chain in time.
Since we want to find the equilibrium conditions of the robot, we can consider the LHS of the equations as equal to 0 given that:

$$\begin{cases} \boldsymbol{v}_{ci} = 0 \\ \boldsymbol{\omega}_i = 0 \end{cases}$$

Keeping this condition in mind we can state that the resultant of all the Forces and the resultant of all the Moments acting on each body must be equal to 0.

$$\begin{cases} \boldsymbol{F}_i = \boldsymbol{0} \\ \boldsymbol{M}c_i = \boldsymbol{0} \end{cases} \qquad i = 1 \div n$$

If now we consider the total virtual work of forces and moments acting on each body of the configuration, we can state that in case of equilibrium conditions:

$$\sum_{i=1}^{n} \delta\boldsymbol{P}_i \cdot \boldsymbol{F}_i + \delta\boldsymbol{\theta}_i \cdot \boldsymbol{M}_i = \boldsymbol{0}$$

Knowing that the bodies are all chained together we can distinguish the forces and the moments as follows:

$$\begin{cases} \boldsymbol{F}_i = \boldsymbol{F}^{(A)}{}_i + \boldsymbol{F}^{(R)}{}_i \\ \boldsymbol{M}^{(R)}{}_i = \boldsymbol{M}^{(A)}{}_i + \boldsymbol{M}^{(R)}{}_i \end{cases} \quad i = 1 \div n$$

Where (A) stands for **Active** (**external** and **actuation**) forces and moments and (R) stands for **Reactive** forces and moments (quantities generated from the interaction of the constrained bodies). Substituting these terms in the previous equation we get:

$$\sum_{i=1}^{n} \delta\boldsymbol{P}_i \cdot [\boldsymbol{F}^{(A)}{}_i + \boldsymbol{F}^{(R)}{}_i] + \delta\boldsymbol{\theta}_i \cdot [\boldsymbol{M}^{(A)}{}_i + \boldsymbol{M}^{(R)}{}_i] = \boldsymbol{0}$$

Moreover, if we rewrite the equation properly, we can clearly distinguish the contribution of the total virtual work of forces and torques in an active contribution and a reactive one.

$$\sum_{i=1}^{n} \delta\boldsymbol{P}_i \cdot \boldsymbol{F}^{(A)}{}_i + \delta\boldsymbol{\theta}_i \cdot \boldsymbol{M}^{(A)}{}_i + \sum_{i=1}^{n} \delta\boldsymbol{\theta}_i \cdot \boldsymbol{M}^{(R)}{}_i + \delta\boldsymbol{P}_i \cdot \boldsymbol{F}^{(R)}{}_i = \boldsymbol{0}$$

Where the first contribution is the total virtual work of all forces and torques due to actuation and externa actions on the body:

$$\delta\boldsymbol{L}^{(A)}{}_{tot}$$

And the second one representing the reaction component to the total virtual work:

$$\delta \wedge_{tot}$$

For our computation we will consider the following assumptions:

1. The total reactive virtual work is considered equal to 0 by the **virtual work principal** for any possible joint displacement consistent with the constraints, namely:

   $$\delta \wedge_{tot} = \boldsymbol{0}$$

2. The active moments and forces can be split int external and actuation forces, namely:

   $$\begin{cases} \boldsymbol{F}^{(A)}{}_i = \boldsymbol{f}_i{}^{(act)} + \boldsymbol{f}_i{}^{(ext)} \\ \boldsymbol{M}^{(A)}{}_i = \boldsymbol{m}_i{}^{(act)} + \boldsymbol{m}_i{}^{(ext)} \end{cases} \quad i = 1 \div n$$

Rewriting the virtual work of forces and torques equation we get:

$$\sum_{i=1}^{n} \delta\boldsymbol{P}_i \cdot \boldsymbol{f}^{(act)}{}_i + \delta\boldsymbol{\theta}_i \cdot \boldsymbol{m}^{(act)}{}_i + \sum_{i=1}^{n} \delta\boldsymbol{P}_i \cdot \boldsymbol{f}^{(ext)}{}_i + \delta\boldsymbol{\theta}_i \cdot \boldsymbol{m}^{(ext)}{}_i = \boldsymbol{0}$$

We can simplify the equation by writing:

$$\delta L^{(act)}{}_{tot} + \sum_{i=1}^{n} \delta L_i = \mathbf{0}$$

Where $\delta \boldsymbol{L}^{(act)}{}_{tot}$ represents total virtual work of actuation forces and torques.
Considering that:

$$\delta \boldsymbol{L}^{(act)}{}_{tot} = \delta \boldsymbol{q}^{\boldsymbol{T}} \cdot \boldsymbol{\tau}$$

and

$$\delta L_i = \delta \boldsymbol{q}^{\boldsymbol{T}} \;\; {}^{\boldsymbol{o}}\boldsymbol{J}^{T}{}_{i/o} \begin{pmatrix} {}^{\boldsymbol{o}}\boldsymbol{M}_i{}^{(ext)} \\ {}^{\boldsymbol{o}}\boldsymbol{F}_i{}^{(ext)} \end{pmatrix}$$

The same formula can be simplified into the following relation:

$$\delta \boldsymbol{q}^{\boldsymbol{T}} \cdot \left[\boldsymbol{\tau} + \sum_{i=1}^{n} \;\; {}^{\boldsymbol{o}}\boldsymbol{J}^{T}{}_{i/o} \begin{pmatrix} {}^{\boldsymbol{o}}\boldsymbol{M}_i{}^{(ext)} \\ {}^{\boldsymbol{o}}\boldsymbol{F}_i{}^{(ext)} \end{pmatrix} \right] = \mathbf{0}$$

So:

$$\boldsymbol{\tau} + \sum_{i=1}^{n} \;\; {}^{\boldsymbol{o}}\boldsymbol{J}^{T}{}_{i/o} \begin{pmatrix} {}^{\boldsymbol{o}}\boldsymbol{M}_i{}^{(ext)} \\ {}^{\boldsymbol{o}}\boldsymbol{F}_i{}^{(ext)} \end{pmatrix} = \mathbf{0} \quad \rightarrow \quad \boldsymbol{\tau}_{eq} = -\sum_{i=1}^{n} \;\; {}^{\boldsymbol{o}}\boldsymbol{J}^{T}{}_{i/o} \begin{pmatrix} {}^{\boldsymbol{o}}\boldsymbol{M}_i{}^{(ext)} \\ {}^{\boldsymbol{o}}\boldsymbol{F}_i{}^{(ext)} \end{pmatrix} \qquad (*)$$

This equation represents a set of ‘n’ equations each one representing the generalized actuation force/ moment of each component of the kinematic chain.

3. Computational solution of the exercise

The (*) relation is the formula needed to find the $\boldsymbol{\tau}_{eq}$ for every element of our kinematic chain.
All the data required to find $\boldsymbol{\tau}_{eq}$ is contained in the *SmiData* file of the manipulator.
This document is a model data file derived from a *Simscape Multibody* Import XML file using the *smimport* function. The data in this file sets the block parameter values in an imported *Simscape Multibody model* containing all the needed info about our rigidbody chain.
The *smiData* file is divided in two main components:

- *RigidTransform*: which contains multiple all the info about translation, angle of rotation, and axis of rotation of the bodies ‘relative reference frame with respect to the base frame of the whole structure in the zero position of the robot (*fig.1*).
- *Solid*: containing the info regarding the center of mass, moment of inertia, and product of inertia for each body of the configuration. All the data is expressed with respect to the relative reference frame of the body.

Both these elements were crucial for the analysis of the robot configurations.

3.1 Data collection from the *smiData* document

The first section of the code is dedicated to the extraction of the data from the *smiData* file.
Running the *Assembly_Datafile* derived from the Inventor’s xml document, the *smiData* will be loaded in the workspace.

From the *Solid* part of the data diagram the **mass** and the **center of mass** with respect to the relative frame will be extracted for every component of the kinematic chain except for the base frame (considered as anchored to ground).
Since **link 1** moves in solidarity with **motor 1** and **link 2** moves in solidarity with **motor 2**, it is convenient to calculate a common center of mass between the two couples. The effect of external forces and moments will later be referred to those new centers of mass.

Since the reference frames of the couples are coincident in every configuration, the computation of the common center of mass is simply:

Function: <*ComputeCenterOfMass()*>

```
function [cm] = Compute_c_m(m1,m2, cm1, cm2)
       cm = zeros(3, 1);

       cm (1) = (m1 * cm1(1) + m2* cm2(1)) / (m1 + m2);
       cm (2) = (m1 * cm1(2) + m2* cm2(2)) / (m1 + m2);
       cm (3) = (m1 * cm1(3) + m2* cm2(3)) / (m1 + m2);
end
```

Input: mass of body_1, mass of body 2, center of mass of body 1, center of mass of body 2.
Output: general center of mass.
Comment: the function will evaluate the center of mass for every axis of the reference frame and will give as an output the value of the common center of mass referred to that specific reference frame.

Every force that acts onto the manipulator has a reflection on the joints. The joints are the only things we can use to control the motion and the Behaviour of the robot.

3.2 Initialization of the problems' configuration.

The following section of the code is the collection of data regarding the positioning of the links in the problems configuration.

- I organized every **angle configuration** for each of the exercises in a vector line. Each of these lines contains the relative angle between the base frame and the first relative reference frame and the relative angle of the second relative frame with respect to the first one. All the 10 angles vectors are contained in **2 by 5 cell**. The rows represent the exercise, and the columns stand for the point where that specific configuration is analyzed. All the other problems' variables are organized in the same way throughout all the code.
- The **points of application of forces and moments** were organized in a **2 by 3 cell**. Each point is described with a column vector of three elements. Each spot represents the linear displacement of the point of application of forces and moments with respect to the relative reference frame of the link they apply effect on.
- All the **forces and moments** are collected in a **2 by 5 cell**. from a screw theory point of view, the vector $\begin{pmatrix} {}^{o}\boldsymbol{M}_i{}^{(ext)} \\ {}^{o}\boldsymbol{F}_i{}^{(ext)} \end{pmatrix}$ can be seen as a unique vector called **wrench** ($\boldsymbol{W}_{ext}$). I organized all the forces acting on the body as a series of wrenches organized in column matrices. Each column of the *i-th, j-th* matrix represents the force vector acting either on one of the points of application of the force or in the center of mass of one of the two bodies.
- Other 2 by 5 cells were created to collect all the data needed to compute the **Jacobian matrices**.

3.3 Computation of the Jacobian matrices.

The Jacobian matrix is the geometric element which provides the relationship between joint velocities and the velocity of any point on the rigidbody components of a manipulator.
In our computation the Jacobian will allow us to switch between two different space coordinates: the **Cartesian space** where the wrenches are applied and the **joint space**. As previously demonstrated, the transpose of the Jacobian will be the main tool to calculate the generalized forces and torques the manipulator needs to apply to keep the kinematic chain in a static configuration.

To get to the computation of the Jacobian implies the computation of other crucial geometrical elements.

Computation of the Rotational Matrices.

The rotational matrix is a geometrical element that expresses the orientation of a base frame with respect to another one. This geometrical element is basically a 3 by 3 matrix composed of columns representing the unit vectors along the axes of one frame, relative to the other, reference frame.
The general structure of a rotational matrix computed between frame $< a >$ and frame $< b >$ is the following:

$$ {}_a^b\boldsymbol{R} = \begin{bmatrix} \boldsymbol{x}^b \cdot \boldsymbol{x}^a & \boldsymbol{y}^b \cdot \boldsymbol{x}^a & \boldsymbol{z}^b \cdot \boldsymbol{x}^a \\ \boldsymbol{x}^b \cdot \boldsymbol{y}^a & \boldsymbol{y}^b \cdot \boldsymbol{y}^a & \boldsymbol{z}^b \cdot \boldsymbol{y}^a \\ \boldsymbol{x}^b \cdot \boldsymbol{z}^a & \boldsymbol{y}^b \cdot \boldsymbol{z}^a & \boldsymbol{z}^b \cdot \boldsymbol{z}^a \end{bmatrix} $$

For the composition of our CAD model, we took advantage of the **Denavit-Hatemberg** configuration, the only rotation we will encounter will be only either around the **z-axis** in the case of rotational joint and around the **y-axis** in the case of translational ones. This assumption simplifies the structure of the rotational matrix and its calculation. Knowing the axis of rotation of the first link's reference frame and also the angle with which it rotates we can simply apply the *axang2rotm(axis, angle)* function to calculate the rotational matrix.
For a rotation around the z-axis the rotational matrix will be constructed as follows:

$$ \boldsymbol{R}_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

For a rotation around the y-axis the rotational matrix will be constructed as follows:

$$ \boldsymbol{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} $$

The rotational matrices were computed for each link for all the exercises' configurations.
The first link rotation's angle is the one contained in the configuration's angles vector. The second one is the sum of the first one with the second one since the second angle is expressed with respect to the link 1 reference frame.

Computation of the Transformation Matrices.

The Transformation matrix between frame $< a >$ and $< b >$ (${}_a^b\boldsymbol{T}$) contains information about the orientation of frame b with respect to a and about the linear displacement between the two (${}_a^b\boldsymbol{O}$).
This geometrical tool is used for describing both the position and orientation of the manipulator links with respect to the base frame. The Transformation matrix is also a fundamental computational step for evaluating the Jacobian matrix.

Finding the ${}^{link-i}_{base}\boldsymbol{O}$ linear displacement is an easy task given the data we already collected. The *SimData.RigidTransform* data retrieves the distance between the relative reference frame of that body part with respect to the base frame in the initial configuration of the manipulator.
This distance represents the linear displacement between the base frame and the reference frame related to link 1 and link 2. Though, this distance must be projected on the base frame for every configuration of the manipulator.
To achieve the ${}^{link-i}_{base}\boldsymbol{O}$ final computation, the *SimData.RigidTransform* distance vector must be pre-multiplied by the **rotational matrix of the *i-th - 1* link**. This computation will retrieve the actual distance between the origin of the second link frame and the base frame. The relative distances were all collected into a **2 by 5 cell** containing only the distances relative to the second link since the center of the first link will always be coincident with the base frame.
The general structure of a transformation matrix between frame $< a >$ and $< b >$ if the following:

$${}^{b}_{a}\boldsymbol{T} = [{}^{b}_{a}\boldsymbol{R} \quad {}^{b}_{a}\boldsymbol{O}]$$

This was the version of the transformation matrix used in the code. There are also other structures of this matrix (ex. Homogenic form: ${}^{b}_{a}\boldsymbol{T} = \begin{bmatrix} {}^{b}_{a}\boldsymbol{R} & {}^{b}_{a}\boldsymbol{O} \\ zeros(1,3) & \boldsymbol{1} \end{bmatrix}$) but they will not be considered throughout the code.

Computation of the Jacobian:

The Jacobian matrix represents the causal relations between the velocities of the manipulator joint and the angular and linear velocity of a point of *i* of the manipulator chain.

The velocity of a point *i,* with respect to the base frame <O> can be written as:

Linear component:

$${}^{o}(\boldsymbol{v}_{i/o}) = {}^{o}\left(\sum_{j=1}^{i} \boldsymbol{J}^{\boldsymbol{L}}_{i/j} \cdot \dot{\boldsymbol{q}}_{J}\right) = \begin{bmatrix} {}^{o}\boldsymbol{J}^{\boldsymbol{L}}_{\frac{i}{1}} & {}^{o}\boldsymbol{J}^{\boldsymbol{L}}_{\frac{i}{2}} & \cdots & {}^{o}\boldsymbol{J}^{\boldsymbol{L}}_{\frac{i}{i}} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_1 \\ \vdots \\ \dot{\boldsymbol{q}}_i \end{bmatrix}$$

Angular component:

$${}^{o}(\boldsymbol{\omega}_{i/o}) = {}^{o}\left(\sum_{j=1}^{i} \boldsymbol{J}^{\boldsymbol{A}}_{i/j} \cdot \dot{\boldsymbol{q}}_{J}\right) = \begin{bmatrix} {}^{o}\boldsymbol{J}^{\boldsymbol{A}}_{\frac{i}{1}} & {}^{o}\boldsymbol{J}^{\boldsymbol{A}}_{\frac{i}{2}} & \cdots & {}^{o}\boldsymbol{J}^{\boldsymbol{A}}_{\frac{i}{i}} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_1 \\ \vdots \\ \dot{\boldsymbol{q}}_i \end{bmatrix}$$

Where:

- $\begin{bmatrix} {}^{o}\boldsymbol{J}^{\boldsymbol{L}}_{\frac{i}{1}} & {}^{o}\boldsymbol{J}^{\boldsymbol{L}}_{\frac{i}{2}} & \cdots & {}^{o}\boldsymbol{J}^{\boldsymbol{L}}_{\frac{i}{i}} \end{bmatrix} = {}^{o}\boldsymbol{J}^{\boldsymbol{L}}_{i/j}$ → represent the linear component of the Jacobian projected onto the base frame <O>.

- $\begin{bmatrix} {}^{o}\boldsymbol{J}^{\boldsymbol{A}}_{\frac{i}{1}} & {}^{o}\boldsymbol{J}^{\boldsymbol{A}}_{\frac{i}{2}} & \cdots & {}^{o}\boldsymbol{J}^{\boldsymbol{A}}_{\frac{i}{i}} \end{bmatrix} = {}^{o}\boldsymbol{J}^{\boldsymbol{A}}_{i/j}$ → represent the angular component of the Jacobian projected onto the base frame <O>.

For our computation these two matrices' size will be a 3 by *2* with where 2 is the number of links. The Jacobians referred to the first link will have a ***null*** secondo column. This is because the j-th column of the Jacobian matrix expresses the effect of the motion of joint *j* on the velocity of point *i*. Since the point we consider for both the computations of the Jacobians are the center of masses of each joint, the second joint's motor will not have any effect on the first center of mass.

- $\begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_i \end{bmatrix}$ → joint velocity vector

Our objective is to calculate the **total Jacobian** which is the composition of these two rearranged in a column vector.
Total jacobian:

$$\boldsymbol{J^{tot}_{i/o}} = \begin{bmatrix} {}^{o}\boldsymbol{J^{A}_{i/j}} \\ {}^{o}\boldsymbol{J^{L}_{i/j}} \end{bmatrix}$$

The angular and linear components of the Jacobian can be reconstructed as follows:

- Linear component:

$$\boldsymbol{J^{L}_{i/j}} = \begin{cases} (\boldsymbol{K_j} \times \boldsymbol{r_{i/j}}) & rotational\ joint \\ \boldsymbol{K_j} & translational\ joint \end{cases}$$

- Angular component:

$$\boldsymbol{J^{A}_{i/j}} = \begin{cases} \boldsymbol{K_j} & rotational\ joint \\ \boldsymbol{0} & translational\ joint \end{cases}$$

Where:

- $\boldsymbol{K_j}$ referrers to axis of rotation of the joint
- $\boldsymbol{r_{i/j}}$ referrers to the relative distance between the base frame and the i-th point.

The code is structured to find the Jacobian matrices using two different functions:

- *Get_J_C()*: which will allow us to calculate a single column of the Jacobian.
- *Get_J()*: structured to put together the different columns and to give the correct input to the *Get_J_C()* function.

Get_J_C():

```
function [J_col] = Get_J_Col (k, r, jointType)

if jointType == 1
    J_a = [0; 0; 0];
    J_l = k;
end
if jointType == 2
    J_a = k;
    J_l = cross(k, r);
end
J_col = [J_a;J_l];
end
```

Input: ***k*** = axis of rotation with respect to the base, ***r*** = distance of the i-th center of mass with respect to the i-th link's relative frame all projected onto the base, *jointType* matrix.
Output: i-th Jacobian column.
Comment: Depending on the value of the joint-type matrix (either 1 or 2), the function will assign the linear and angular Jacobian vectors and it will merge them together into a singular total column. The geometrical elements passed to the function are all referred to the base frame because that's where all the cartesian

forces and torques are referred to. Projecting all the elements on the base frame will let us apply the previously demonstrated formula for the calculation of the **vector of the generalized actuator forces and torques** by simply recursively multiply the Jacobian matrix with the given external wrenches.

Get_J():

```
function J = Get_J(T, cm, jointType, link)

    numLinks = length(jointType);
    J = zeros(6, numLinks);
    r = zeros(3, numLinks);
    z_ax = [0 0 1]';
    for i = 1 : link

        r(:, i) = cm(1:3) - T{1}((1:3),4,i);

        k = T{1}((1:3),(1:3),i) * z_ax;

        J(:, i) = Get_J_Col(k, r(:,i), jointType(i));

    end
end
```

Input: ***T*** = Transformation matrix of the i-th frame with respect to the base, ***cm*** = i-th center of mass with respect to the base, ***jointType***, ***links*** = link number.
Output: ***J*** = total Jacobian matrix.
Comment: the function will evaluate the parameters needed for the computation of the Jacobian column, projecting every needed geometrical component on the base frame. The internal recursion of the function will build up the **total Jacobian** for both the first and second link.

3.4 Computation of the Rigidbody Jacobian.

A robotic arm is rarely only subjected to the force of gravity. Most of the time also other external factors apply forces and moments to the rigid body chain. In our computation we are referring our Jacobian matrices to the center of mass of each link. If an externa force (or moment) was applied to a different point other than the center of mass, it'd be necessary to study the effect of this external element on the center of mass of the chain component it is applied on. The **rigidbody Jacobian** comes into play for dealing with this kind of external events.
The **rigidbody Jacobian** is a 6 by 6 matrix structured as follows:

$$ {}^{o}\boldsymbol{S}_{p_i/c_i} = \begin{bmatrix} \boldsymbol{I}^{3\times3} & \boldsymbol{0}^{3\times3} \\ -[\boldsymbol{r}_{p_i/c_i} \times] & \boldsymbol{I}^{3\times3} \end{bmatrix} $$

The $[\boldsymbol{r}_{p_i/c_i} \times]$ matrix represents the skew matrix of the distance vector between the point *pi* and the center of mass *ci*. By multiplying the skew matrix with any vector in the cartesian space it will give as output the vectorial product between the two. Thanks to this property, by pre-multiplying the external wrench vector with this geometrical element referred to the i-th center of mass, we will obtain the effect of that wrench as it was applied in the center of mass.

$$ \boldsymbol{\tau}_{eq} = -\sum_{i=1}^{n} {}^{o}\boldsymbol{J}^{T}_{c_i/o}\ {}^{o}\boldsymbol{S}_{p_i/c_i} \begin{pmatrix} {}^{o}\boldsymbol{M}_{i}^{(ext)} \\ {}^{o}\boldsymbol{F}_{i}^{(ext)} \end{pmatrix} = -\sum_{i=1}^{n} {}^{o}\boldsymbol{J}^{T}_{c_i/o} \begin{pmatrix} {}^{o}\boldsymbol{M}_{c_i}^{(ext)} \\ {}^{o}\boldsymbol{F}_{c_i}^{(ext)} \end{pmatrix} $$

The following function computes the rigid body Jacobian matrix for a force applied on a known point.

RigidBody_J():

```
function [S] = RigidBody_J(v)

skew = [0 -v(3) v(2);v(3) 0 -v(1);-v(2) v(1) 0];
S = [eye(3), zeros(3,3); skew , eye(3)];

end
```

Input: ***v*** = vector of the relative distance between the i-th link's center of mass and the *p* point of application of the force, all projected onto the base:
Example:

```
P1_wrt_base= T{2,3}(:,:,2) * [P{2,1}(:,1);1];

S_p1 = RigidBody_J(CM_link_i_t_b{2,3}(:,2)-P1_wrt_base);
```

Where:

- *T{2,3}(: , : , 2)* = rappresents the transformation matrix of the second link of the chain for the exercise 2 problem 3. By multiping this values with the distance of the point of application of the force with respect to the reference frame of the second link (*[P{2, 1}(:, 1);1]*), the projection of that point with respect to the base frame will be obtained.
- By subtractiong the projection of point *P1* to the vector of the center of mass of the second link with respect to the base frame(*CM_link_i_t_b{2,3}(:,2)*), the relative distance between the center of mass and the point i will be calculated. This vector will be used in the function for building the ckew section of the rigidbody Jacobian matrix.

Output: ***S*** = rigid body Jacobian matrix.

3.5 Calculating $\boldsymbol{\tau}_{eq}$

knowing that external actions are additive, we can view the problem of calculating the $\boldsymbol{\tau}_{eq}$ as a recursive sum of effects concerning the forces applied on the manipulator.
For every wrench applied on the center of mass (ex. Force of gravity) the Jacobian matrix will be simply multiplied by the wrench. In case of an external force applying onto the manipulator's links, the rigid body Jacobian matrix will pre multiply the wrench vector to transport it onto the center of mass of the link the force is applied on.

Example ex. 1.5:

Formula:

$$\boldsymbol{\tau}_{eq} = -\, {}^{o}\boldsymbol{J}^{T}_{c_1/o}\ {}^{o}\boldsymbol{S}_{p_3/c_1}\begin{pmatrix}\mathbf{0}\\ {}^{o}\boldsymbol{F}_3^{(ext)}\end{pmatrix} - {}^{o}\boldsymbol{J}^{T}_{c_2/o}\ {}^{o}\boldsymbol{S}_{p_2/c_2}\begin{pmatrix}\mathbf{0}\\ {}^{o}\boldsymbol{F}_2^{(ext)}\end{pmatrix} - {}^{o}\boldsymbol{J}^{T}_{c_1/o}\begin{pmatrix}\mathbf{0}\\ m_1\,{}^{o}\boldsymbol{g}_y^{(ext)}\end{pmatrix} - {}^{o}\boldsymbol{J}^{T}_{c_2/o}\begin{pmatrix}\mathbf{0}\\ m_2\,{}^{o}\boldsymbol{g}_y^{(ext)}\end{pmatrix}$$

code:

```
tau{1,5}(:,1) = − (J{1,5}(:,:,1)' ∗S_p3' ∗ W{1,5}(:,1) + J{1,5}(:,:,2)' ∗ S_p2' ∗W{1,5}(:,2)) −
(J{1,5}(:,:,1)' ∗ W{1,5}(:,3)+ J{1,5}(:,:,2)'∗W{1,5}(:,4));
```

The $\boldsymbol{\tau}_{eq}$ vector will contain the value of the forces or moments the arm must apply to its joint to hold the equilibrium configuration.

$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

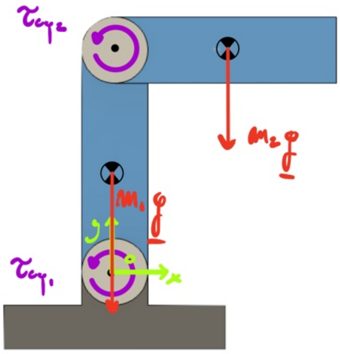
Depending on the joint-type the unit of measure of the $\boldsymbol{\tau}_{eq}$ components will change.

- $[Nmm]$→ rotational joint
- $[N]$→ prismatic joint

3. Solution of the given problems

3.1 Exercise 1

problem 1



Angles of the configuration:

$$ang_{link_1} = [\frac{\pi}{2}]$$
$$ang_{link_2} = \left[-\frac{\pi}{2}\right]$$

External forces:

$$m_1\boldsymbol{g} = [0, 0, 0, 0, -1097.8, 0\ ]\ [Nmm \mid N]$$
$$m_2\boldsymbol{g} = [0,0,0,0, -1201.46, 0]\ [Nmm \mid N]$$

Tau equivalent:

$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} 579770\ [Nmm] \\ 579770\ [Nmm] \end{bmatrix}$$

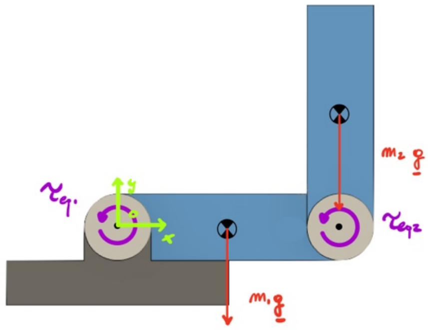
Comments:

As we can notice, only the gravity force is acting on the body. The two moments coming out are the same. This is because the following two reasons:

- The weight of the first link doesn't have any moment effect on the configuration since the lever arm and the weight force are parallel.
- The gravity force acting on the second link have the same distance from the motors' rotation fulcrums.

Fabio Conti
Mat.4693053

problem 2



Angles of the configuration:

$$ang_{link_1} = [0]$$
$$ang_{link_2} = \left[\frac{\pi}{2}\right]$$

External forces:

$$m_1\boldsymbol{g} = [0, 0, 0, 0, -1097.8, 0\ ]\ [Nmm \mid N]$$
$$m_2\boldsymbol{g} = [0,0,0,0, -1201.46, 0]\ [Nmm \mid N]$$

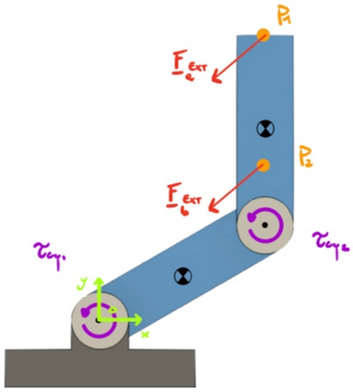
Tau equivalent:

$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} 1694925\ [Nmm] \\ 0\ [Nmm] \end{bmatrix}$$

Comments:

Since the gravity force acting on the second link is parallel to its lever arm, it doesn't apply any moment on the second joint.
The effect of the force of gravity on the second and first link are both counter balanced only by the first joint.

problem 3



Angles of the configuration:

$$ang_{link_1} = \left[\frac{\pi}{6}\right]$$
$$ang_{link_2} = \left[\frac{\pi}{3}\right]$$

External forces:

$$\boldsymbol{F}_{ext\ 1} = [0,0,0,-0.7,-0.5,0]\ [Nmm \mid N]$$
$$\boldsymbol{F}_{ext\ 2} = [0,0,0,-0.7,-0.5,0]\ [Nmm \mid N]$$

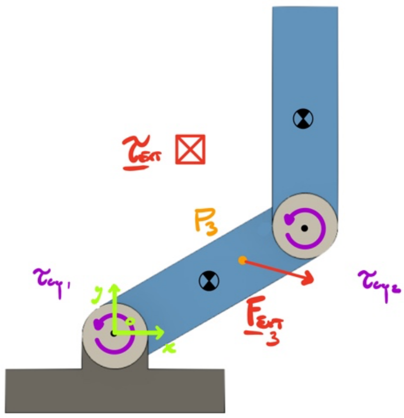
Tau equivalent:

$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} -616\ [Nmm] \\ -700\ [Nmm] \end{bmatrix}\ \boldsymbol{\tau}_{eq} = \begin{bmatrix} -126\ [Nmm] \\ -210\ [Nmm] \end{bmatrix}$$

Comments:

In this case there is no gravity force but only two external forces are acting. Though, the two forces are not applied on the center of mass. The Jacobians of every exercise are calculated taking as reference the center of mass of the links. having a force applied on another point of the link implies that the wrench must be pre-multiplied by the rigid Jacobian matrix. This geometrical application will let us use the same formulation for the calculation of the $\boldsymbol{\tau}_{eq}$ vector. The wrench vectors will try to make the robot turn counterclockwise. The actuation forces will be negative to counterbalance the effect of the forces.
The second formulation of the problems shows of lower levels of $\boldsymbol{\tau}_{eq}$ provided a shorter lever arm.

problem 4



Angles of the configuration:

$$ang_{link_1} = \left[\frac{\pi}{6}\right]$$
$$ang_{link_2} = \left[\frac{\pi}{3}\right]$$

External force and moment:

$$\boldsymbol{F}_{ext\ 3} = [0, 0, 0, 1.5, -0.3, 0]\ [Nmm \mid N]$$
$$\boldsymbol{M}_{ext} = [0,0,1200,0,0,0]\ [Nmm \mid N]$$

Tau equivalent:

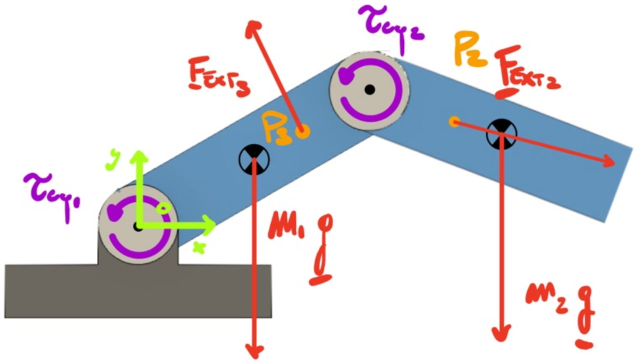
$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} -291\ [Nmm] \\ -1200\ [Nmm] \end{bmatrix}$$

Comments:

On this configuration a momentum and a force are acting on the manipulator. Only the force wrench must be translated on the link's center of mass since the effect of the toque doesn't change depending on where it is applied.

Fabio Conti
Mat.4693053

problem 5



Angles of the configuration:

$$ang_{link_1} = \left[\frac{\pi}{6}\right]$$
$$ang_{link_2} = \left[-\frac{\pi}{3}\right]$$

External forces:

$$m_1\boldsymbol{g} = [0, 0, 0, 0, -1097.8, 0\ ]\ [Nmm \mid N]$$
$$m_2\boldsymbol{g} = [0, 0, 0, 0, -1201.46, 0]\ [Nmm \mid N]$$
$$\boldsymbol{F}_{ext\ 3} = [0, 0, 0, 1.2, -0.2, 0]\ [Nmm \mid N]$$
$$\boldsymbol{F}_{ext\ 2} = [0,0,0, -0.4, -0.2,0]\ [Nmm \mid N]$$

Tau equivalent:

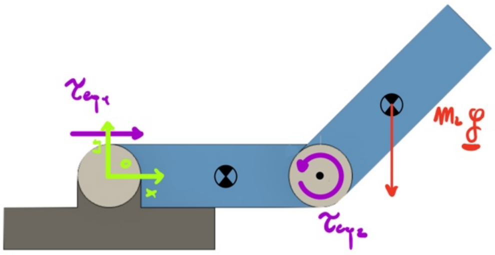
$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} 1969149\ [Nmm] \\ 501844\ [Nmm] \end{bmatrix}$$

Comments:

In this case a combination of gravity forces and external forces are acting on the body.
The rotation caused by the gravity force is higher than the external forces. The $\boldsymbol{\tau}_{eq}$ will counterbalance mainly the gravity forces giving the noticeable on the negative sign.

3.2 Exercise 2

problem 1



Angles of the configuration:

$$ang_{link_1} = [0]$$
$$ang_{link_2} = [\frac{\pi}{4}]$$

External forces:

$$m_2\boldsymbol{g} = [0, 0, 0,0, -1201.4,0]\ [Nmm \mid N]$$

Tau equivalent:

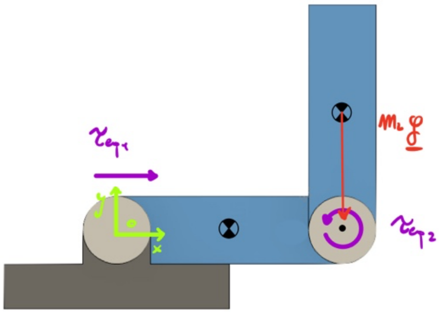
$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} 0\ [N] \\ [Nmm] \end{bmatrix}$$

Comments:

Since there is no gravity acting on the first link, the only momentum to be counterbalanced is taken care by the second joint. All the momentum is counterbalanced by the second link since there is no force parallel to the axis of the prismatic joint which means that no external forces are to be counterbalanced by it.

Fabio Conti
Mat.4693053

problem 2



Angles of the configuration:

$$ang_{link_1} = [0]$$
$$ang_{link_2} = [\frac{\pi}{2}]$$

External forces:

$$m_2\boldsymbol{g} = [0, 0, 0,0, -1201.4,0]\ [Nmm \mid N]$$

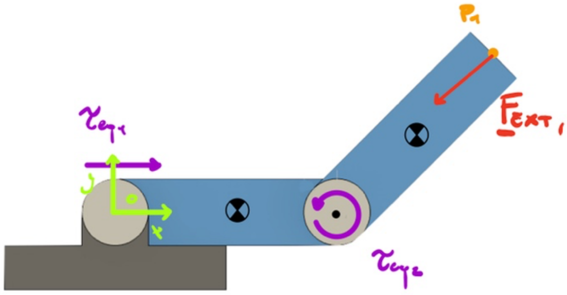
Tau equivalent:

$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} 0\ [N] \\ [Nmm] \end{bmatrix}$$

Comments:

In this configuration only the gravity force on the second link is acting on the sys. Given the geometrical configuration, the gravity force is parallel to the lever arm. Since the gravity force has no horizontal component the first joint will not actuate any kind of force. We can conclude that the system is already in balance. There is no need for external forces.

problem 3



Angles of the configuration:

$$ang_{link_1} = [0]$$
$$ang_{link_2} = [\frac{\pi}{4}]$$

External forces:

$$\boldsymbol{F}_{ext\ 1} = [0,0,0,-0.8,-0.8,0]\ [Nmm \mid N]$$

Tau equivalent:

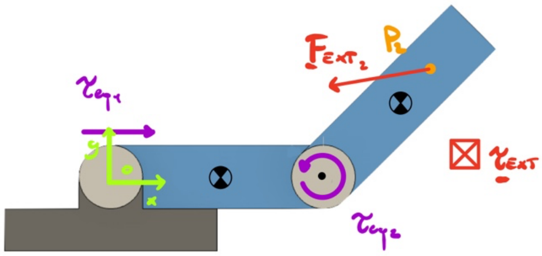
$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} 0.8\ [N] \\ 0\ [Nmm] \end{bmatrix}$$

Comments:

The only force acting in this case is the $\boldsymbol{F}_{ext\ 1}$ external one. The horizontal component will be counterbalanced by the actuation of the first link, while the second one will not cause any actuated torque because the lever arm and the force direction are parallel.

Fabio Conti
Mat.4693053

problem 4



Angles of the configuration:

$$ang_{link_1} = [0]$$
$$ang_{link_2} = [\frac{\pi}{4}]$$

External forces:

$$\boldsymbol{F}_{ext\ 2} = [0,0,0,-0.8,-0.2,0]\ [Nmm \mid N]$$
$$\boldsymbol{M}_{ext} = [0,0,500,0,0,0]\ [Nmm \mid N]$$

Tau equivalent:

$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} 0.8\ [N] \\ -775\ [Nmm] \end{bmatrix}$$

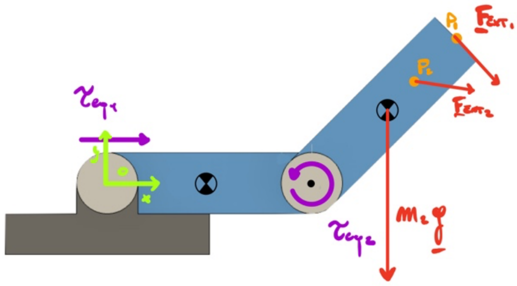
Comments:

This configuration describes the acting of a momentum wrench and a force wrench. The external momentum doesn't have an impact on the actuation of the first joint since it is prismatic. The external force will make both links produce a generalized actuation force/moment.

5, May, 2022

Fabio Conti
Mat.4693053

problem 5



Angles of the configuration:

$$ang_{link_1} = [0]$$
$$ang_{link_2} = [\frac{\pi}{4}]$$

External forces:

$$m_2\boldsymbol{g} = [0,0,0,0,-1201.4,0]\ [Nmm \mid N]$$
$$\boldsymbol{F}_{ext\ 1} = [0,0,0,0.5,-0.6,0]\ [Nmm \mid N]$$
$$\boldsymbol{F}_{ext\ 2} = [0,0,0,1,-0.4,0]\ [Nmm \mid N]$$

Tau equivalent:

$$\boldsymbol{\tau}_{eq} = \begin{bmatrix} -1.5\ [N] \\ 411455\ [Nmm] \end{bmatrix}$$

Comments:

In this case a combination of external wrenches and the gravity force are acting on the structure. Both the signs of $\boldsymbol{\tau}_1$ and 2 are negative because the in both cases the force to be counterbalance is negative as well as the moment.