Mattia Piras n 4524072 31/05/2022

Robotics Engineering AY. 2021/2022

Università di Genova

Robot dynamics and control second assignment's report

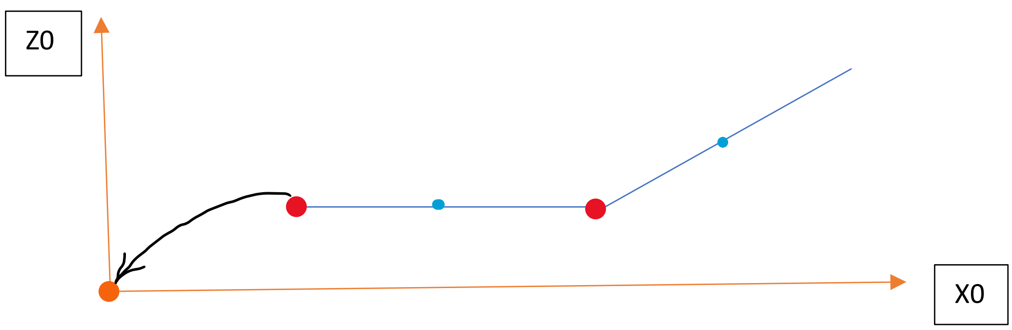
Inverse Dynamic of serial manipulators

INTRODUCTION:

The Newton-Euler inverse dynamic equation is a recursive approach to compute the actuation torques of an open chain robot.

Considering an n-link robot with an end-effector, each link is a rigid body and the center of mass of each link is shown below:

MY ABSOLUTE REFERENCE FRAME <0>

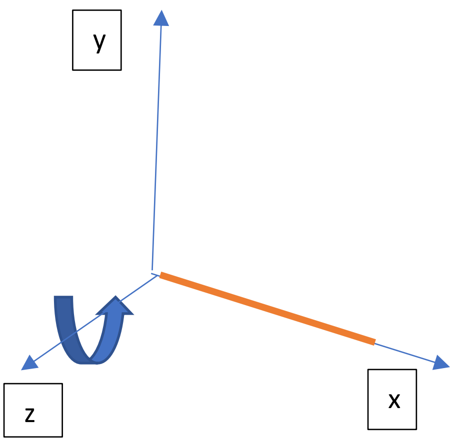


We define a frame at the beginning of each link, and we define z_i to be the twist of link i expressed in frame < i >.

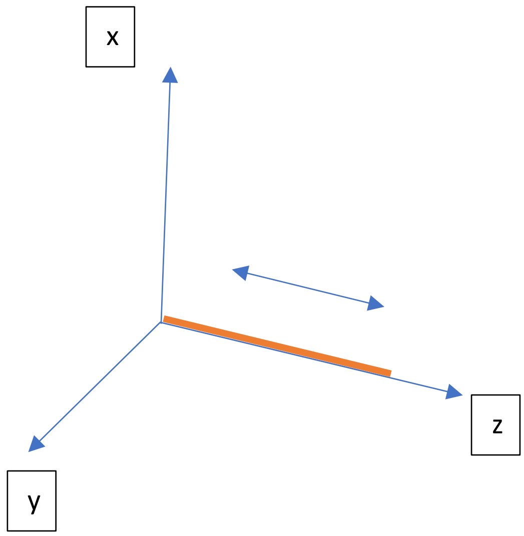
NOTE: my frame directions are different with respect to the ones shown in the exercises.

Moreover, according to the Denavitt-Heartemberg rule each joint z-axis is directed along its motion. (So, exiting from the sheet if rotational or along the link if prismatic).

MY LINK REFERENCE FRAMES <i>



Rotational joint



Prismatic joint

With these definitions I'll quickly summarize the algorithm.

FORWARD ITERATION

This calculates the configuration, twist and acceleration of each link, starting from link 1 and moving outward.
Given the vectors of joint positions (in each reference frame with respect to the previous one), velocities and accelerations and starting from link 1 we calculate the twist $\mathfrak{z}_i$ of the link i as the sum of the twist of link <i-1> but expressed in the i-th frame and the added velocity due the joint velocity $\dot{q}$.
Then we calculate the acceleration of link i as the sum of the acceleration of link <i-1> but expressed in the i-th frame plus the added acceleration due to the joint i acceleration $\ddot{q}$, plus a velocity-product term due to $\dot{q}_i$ and the twist $\mathfrak{z}_i$.

After the forward iterations are completed, we have the configuration, twist, and acceleration of each link.

1.1 POSITION AND ORIENTATION COMPUTATION

- ${}^{i-1}_{i}R = R_i * R_z(q_i)$
- $r_{i/(i-1)} = \begin{cases} Q_{i-1} - P_i & if\ i\ Rotational \\ (Q_{i-1} - P_i) + K_i * q_i & if\ i\ Prismatic \end{cases}$

I have computed this step, in a slightly different way.

In the first part of the MATLAB file, there is a section dedicated to the whole movement of the robot. It starts from the starting point, in which every q_i = 0 and ends up to the desired position, which the real problem starts from.

In this time interval, every transformation matrix from i th frame to the i-1-th is computed, as well the distance from this frame to the base (for the simple sake of graphical representation).

From these transformations matrices I have extracted the movement axis

(Ki = biTei(1:3,3,i)) and the distance of i wrt i-1 (ri = biTei(1:3,4,i)).

1.2 VELOCITY COMPUTATION

- $\omega_{i/o} = \begin{cases} \omega_{i-1/o} + K_i * \dot{q}_i & if\ i\ RJ \\ \omega_{i-1/o} & if\ i\ PJ \end{cases}$
- $v_{i/o} = \begin{cases} v_{i-1/o} + \omega_{i-1/o} \wedge r_{i/(i-1)} & if\ i\ RJ \\ v_{i-1/o} + \omega_{i-1/o} \wedge r_{i/(i-1)} + K_i * q_i & if\ i\ PJ \end{cases}$

Regarding this stage:

Inside the code, you can notice a separate function/script called Velocity_Computation. Its aim is the upper mentioned calculation.

What's inside?

Obviously, I passed every Ki, $\dot{q}_i$, and the previous angular and linear velocities (if the link taken into consideration is not the first).

NB: These previous velocities have to be rotated in the following frame with the transposed of biTei(:,:,i) since this last matrix refers to i wrt i-I and we want to move Vel(1:6,i-1) into the next link reference space.

(NOTE: Vel is a 6 rows, (numberofjoints-1) columns matrix. The first three components of the i-th column refer to the angular velocity of the i-link while the last three refer to the linear velocity component).

I also passed the joint type of the one considered in order to do the correct computation between the two possible choices.

1.3 ACCELERATION COMPUTATION

- $\dot{\omega}_{i/o} = \begin{cases} \dot{\omega}_{i-1/o} + (\omega_{\frac{i-1}{o}} \wedge K_i)\, \dot{q}_i + K_i * \ddot{q}_i & if\ i\ RJ \\ \dot{\omega}_{i-1/o} & if\ i\ PJ \end{cases}$
- $\dot{v}_{i/o} = \begin{cases} v_{i-1/o} + \dot{\omega}_{i-1/o} \wedge r_{i/(i-1)} & if\ i\ RJ \\ v_{i-1/o} + \dot{\omega}_{i-1/o} \wedge r_{i/(i-1)} + K_i * q_i & if\ i\ PJ \end{cases}$

Inside the code:

The same concepts of the Velocity Calculation apply to the acceleration one. A separate function is present, called Acceleration_Computation and does exactly the upper mentioned calculations.

The concepts of bringing the measures of i-1-th link in the i-th link with the premultiplication for transpose(biTei(:,:,i)) are here considered too.

ACCELERATION WITH RESPECT THE CENTER OF MASS, D AND Δ

This is an intermediate step.

In this section we compute:

The linear acceleration with respect to the center of mass of each link

- $\dot{v}_{Ci/o} = \dot{v}_{i/o} + \dot{\omega}_{i-1/o}{}^\wedge r_{Ci/i} + \omega_{i/o}{}^\wedge(\omega_{i/o}{}^\wedge r_{Ci/i})$

The term that considers this acceleration and, multiplying for correspondent link mass, plasma *the gravitational force.*

- Di = $m_i * \dot{v}_{Ci/o}$

The term (Delta_i in the code) that considers the Inertia Matrix of each link for computing its *moment*, since it is a rigid body with a mass and a length.

- Δ_i = $I(\dot{\omega}_{i\,/0}\ ) + \omega_{i\,/0}$ ^ $I(\omega_{i\,/0}))$

NOTE: This given inertia matrices are referring to the Inertia Matrices of the link i, in its reference frame, with respect to its centre of mass.

Indeed, I have noticed that the given values, or the better the only one that goes inside the matrix diagonal spot where the motion should be computed around, is equal to (m l^2) /12.

BACKWARD ITERATIONS

Now we perform the backward iterations, calculating the required joint forces and torques starting from joint n and going back to the joint 1.

First, we calculate the wrench F_i required for link I as the sum of the wrench F_i+1, which is the wrench needed at link i-plus-1 but expressed in the <i> frame, plus the wrench needed to accelerate link <i>, using the inverse dynamics of the rigid body.

Then we calculate τ_i as the component of the wrench F-I along the joint screw axis. Note that only the portion of the wrench must be applied by the joint motor; the rest of the wrench is provided passively by the mechanical structure of the joint, such as the bearings.

At the end of the backward iterations, we have calculated all the joint forces and torques needed to create the desired joint accelerations at the current joint positions and velocities.

FORCES COMPUTATION:

$$F_{i/(i-1)} = F_{(i+1)/i} - m_i g_i - F_{Ci}^{ext} + D_i$$

MOMENTS COMPUTATION

$$M_{i/(i-1)} = M_{(i+1)/i} - M_{Ci}^{ext} + -(r_{i/Ci} \wedge F_{i/(i-1)}) +$$
$$+ (r_{i+1/Ci} \wedge F_{i+1/i}) + \Delta_i$$

TORQUES

$$\tau_i = \begin{cases} M_{i/(i-1)} * Ki \\ F_{i/(i-1)} * Ki \end{cases}$$

That means that the torques are the projection of the moment (if rotational) or the force (if prismatic) on the corresponding link.

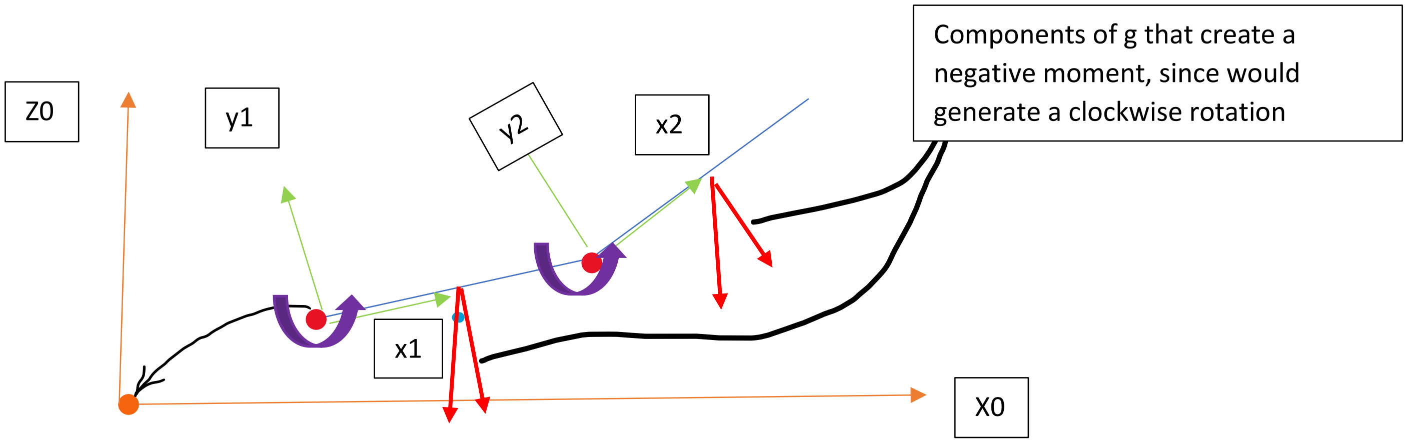
ABOUT THE ALGORITHM

The Newton- Euler Recursive method is efficient because:

- It involves no differentiation
- Computationally efficient due to its recursive nature
- Since I worked only in link frame and not in absolute frame, I avoided lots of matrix multiplications that could result in slow performances.

EXERCISES AND RESULTS

1)



Data set 1.1:

	type	Length (m)	Mass (kg)	q (degrees)	$\dot{q}$ (rad/sec)	$\ddot{q}$ (rad/sec^2)	I_{zz} (Kg m^2)
Joint 1	Rot	1	22	20	0.2	0.1	0.4
Joint 2	Rot	0.8	19	40	0.15	0.085	0.3

Results:

with NO gravitational force

$$\tau = \begin{cases} 4.3641\ Nm \\ 1.3955\ Nm \end{cases}$$

with gravitational force

$$\tau = \begin{cases} 318.1937\ Nm \\ 38.6735\ Nm \end{cases}$$

Since I desire, for each joint, a positive rotation, velocity, and acceleration, it is right to have positive torques as result. They will generate that movement. Each torque has to generate a counter clockwise rotation, so a positive one (since we are speaking about two revolute joints).

The gravity doesn't help at all, since it acts on the opposite "direction" with respect to the general movements and with respect to the position of the links (and therefore the two torques in the gravity activated mode are so relatively big, because they must contrast g). The first torque is greater than the second because the first joint, in addition to forces and Inertia on itself, must move itself but the second one too in a sense. It, indeed, is affected by the forces, the Inertia and, if gravity-on mode, the gravitational force applied on the second link

Data set 1.2:

	type	Length (m)	Mass (kg)	q (degrees)	$\dot{q}$ (rad/sec)	$\ddot{q}$ (rad/sec^2)	I_{zz} (kg m^2)
Joint 1	Rot	1	22	90	-0.8	-0.4	0.4
Joint 2	Rot	0.8	19	45	0.35	-0.1	0.3

with NO gravitational force

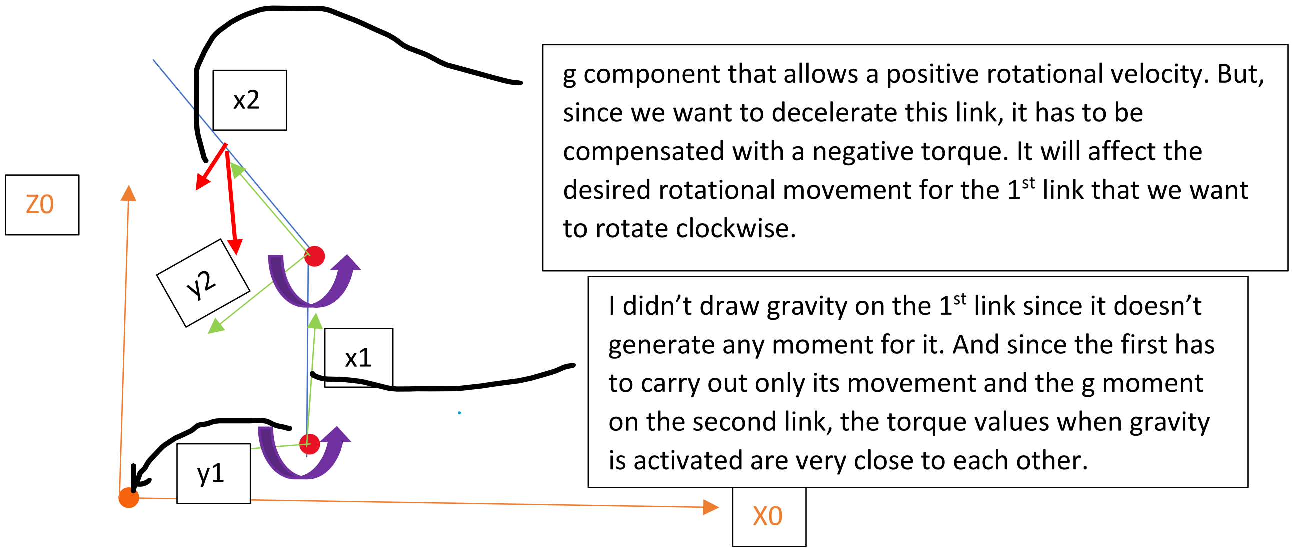
$$\tau = \begin{cases} -12.3727\ Nm \\ 0.2878\ Nm \end{cases}$$

with gravitational force

$$\tau = \begin{cases} -65.0917\ Nm \\ -52.4313\ Nm \end{cases}$$

This time the situation is different from the previous section of exercise 1:

This time the considerable things are:

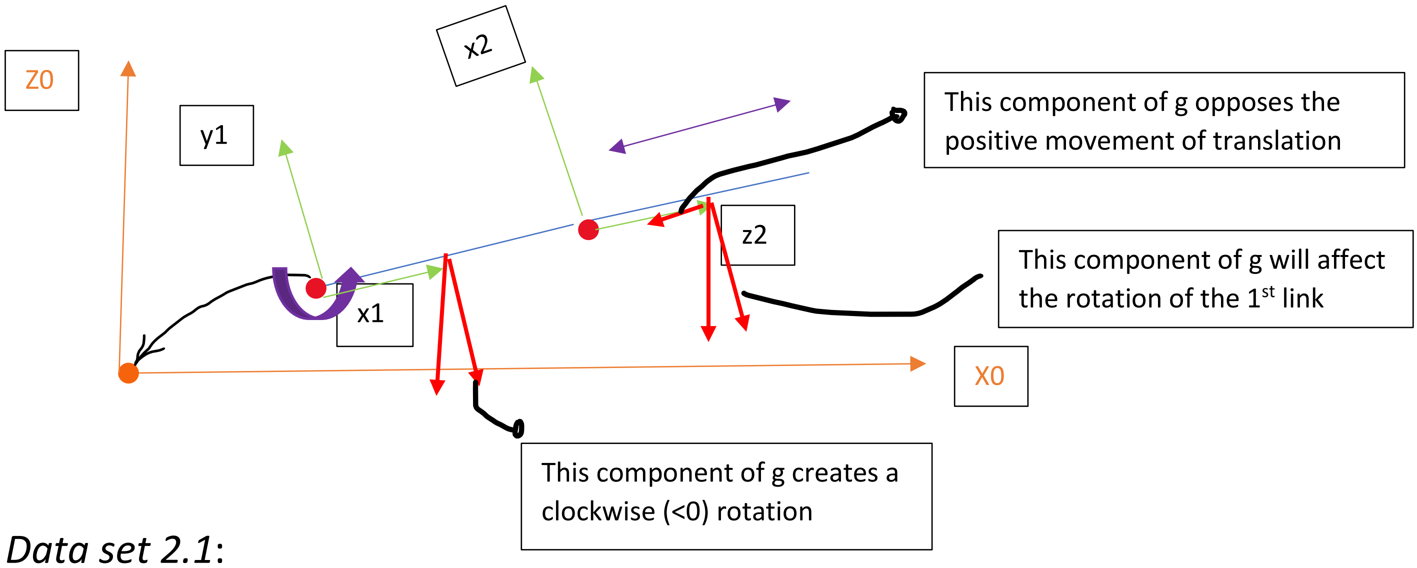


- The first link is rotated of 90 degrees with respect to the absolute x-axis. This means that the gravity will not affect that directly but only through the second link that has a weight and reaction forces applied on.
- The desired velocity and acceleration for the first link are negative and since it is orthogonal to the (x,y) absolute plane its torque has to be negative while for the second joint the velocity is positive and the acceleration is negative, that's why the correspondent torque is positive.
- If gravitational effects are taken into account, the first torque has to counter fight the gravity acting on the second link, as just mentioned.
- The second joint will rotate counter clockwise (q_dot> 0→ So following the falling for better saying) but has to decelerate in some way, and since the gravity will inevitably accelerate it, our torque value, for this joint, has to be negative: some of the work is done by the gravity and the torque on the second joint is limited on the deceleration phase while the velocity phase has been taken care by the gravitational force.

2)

IMPORTANT NOTE:

To run both sub exercises 2 you should enter '1' concerning the translation motion. I had to do like so only for graphical reasons. The exercises will be performed as requested.



Data set 2.1:

	type	Length (m)	Mass (kg)	q	$\dot{q}$	$\ddot{q}$	I_{zz} (kg m^2)
Joint 1	Rot	1	10	20 deg	0.08 (rad/s	-0.4 (rad/s^2)	0.4
Joint 2	Prism	0	6	0.2 m	0.03 (m/s)	-0.1 (m/s^2)	0.3

with NO gravitational force

$$\tau = \begin{cases} 1.2186\ Nm \\ 0.0139\ N \end{cases}$$

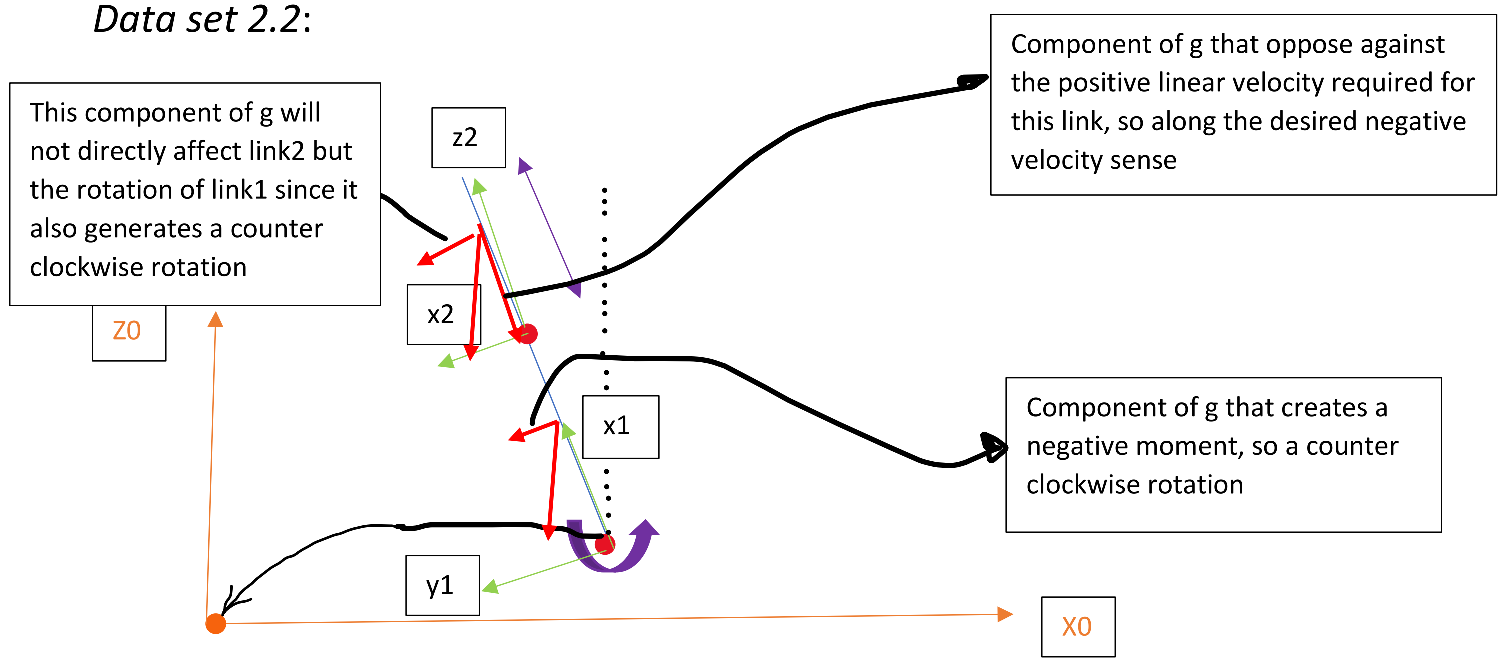
with gravitational force

$$\tau = \begin{cases} 113.6829\ Nm \\ 20.1452\ N \end{cases}$$

In this case we are considering one revolute joint and a prismatic one. The first one, turning in a counter-clock way has given a positive velocity and a negative acceleration, as well as the second that, instead, has to translate decelerating. That's why the torque values with no gravitational effects are so small.

When **g** is activated, it has to be opposed since its sense is along the deceleration, but our velocities are positive. It means that, concerning both joints, the torques are positive since they have to oppose to the deceleration (even if they have to decelerate but not so much as the gravity would do) in order to perform their positive velocities, physically visible by a counter clockwise rotation for the 1st and a translation with a positive (x,z) plane component for the 2nd. Concerning the second joint, it slides along its z-axis (as shown in the previous picture) according to the denavitt rule. As before we want a positive sliding velocity and a negative acceleration. Since the

Data set 2.2:



	type	Length (m)	Mass (kg)	q (degrees)	$\dot{q}$	$\ddot{q}$	$\cancel{I_{zz}}$ (kg m^2)
Joint 1	Rot	1	10	120	-0.8 (rad/s)	-0.1 (rad/s^2)	0.4
Joint 2	Prism	0.8	6	0.6	-0.08 (m/s)	-0.01 (m/s^2)	0.3

with NO gravitational force

$$\tau = \begin{cases} -1.2416\ Nm \\ -1.5960\ N \end{cases}$$

with gravitational force

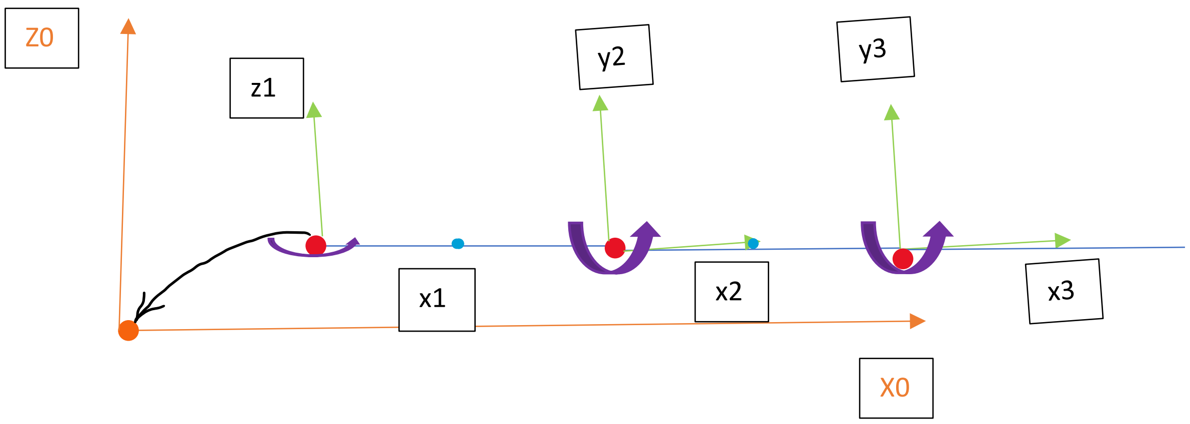
$$\tau = \begin{cases} -72.8546\ Nm \\ 49.3783\ N \end{cases}$$

In this case:

With gravity off: the first joint have to accomplish a negative velocity and acceleration. This means that has to ‘come back’ (means rotation on the clockwise side). The second joint has to accomplish a negative velocity and acceleration. (That’s why the torque value is low and negative)

With gravity on: Since it passed the 90 degrees, the first joint will have the gravity against its desired movement. That’s why a negative and high torque value. The second joint, with a negative acceleration and velocity has the gravity helping it on perform the velocity ‘phase’. But since it must decelerate, the torque value is positive and quite high (considering that it is the last one and so it has not the terms derived from the subsequent link) since it has to oppose against, as said, the gravity.

3)



Data set 3:

	type	Length (m)	Mass (kg)	q (degrees)	$\dot{q}$ (rad/sec)	$\ddot{q}$ (rad/sec^2)
Joint 1	Rot	1	20	20	-0.2	0.1
Joint 2	Rot	0.8	20	40	0.15	0.085
Joint 3	Rot	0.35	6	10	-0.2	0

with NO gravitational force

$$\tau = \begin{cases} 5.1128\ Nm \\ 1.3712\ Nm \\ 0.1532\ Nm \end{cases}$$

with gravitational force

$$\tau = \begin{cases} 5.1128\ Nm \\ 104.1829\ Nm \\ 6.7743\ Nm \end{cases}$$

The situation is that our manipulator will have:

The first z-axis vertical and the other two exiting from the paper. (See the actual design running the code).

Since the first joint remains horizontal each time, with and without gravity will have the same torque allowing only the motion on the absolute plane (x,y).

The second joint must move itself, the third one (or better, is affected by its effects) and then has to compensate the gravity that acts on the oppositive way with respect the desired motion.

The third joint has to carry on only its weight and to perform the desired motion (q_dot = -0.2). A part of this is done by the gravitational force, so the torque will be no so high and positive (for avoiding acceleration. If it would have been negative it would have had an accelerated motion).