Task control with moving RCM

Medical Robotics Project

Summary

Abstract

In this report, we will show the introduction of robot assistant in minimally invasive surgery operation.
Firstly, we will introduce a novel architecture for the control of a moving remote center of motion during surgical robotic operation. This kinematic control strategy allow the insertion of the tool inside the insertion point respecting the structural constraint imposed by the trocar. In the real operation the robot tool is physical constrained by a mechanical part which is the trocar and at the RCM point is not allowed any kind of lateral motion.
In practice, the robot has to consider also the fact that the reference point can be moved by patient's breathing or heart beating. In order to consider also this dynamic effect, as first step we have built up the dynamic model of our operating redundant robot, a KUKA LBR manipulator, using the Newton-Euler method to compute dynamic terms with more efficiency.
Then using this computation, we have implemented a strategy to estimate the external forces using the residual method and comparing different operational conditions varying the intensity, typologies and frequencies of this forces we have evaluated how the system reacts.
Finally, we have reported several significant numerical results obtained executing the whole system in a simulation environment in Matlab.

Introduction

The operating modality of Minimally Invasive Surgery (MIS) consists in introducing the surgical tools inside the patient's body through small incisions and in guiding them from the outside of the body. These operative conditions constrain the manipulation and are less natural and more tiring for the surgeon with respect to open surgery.
The introduction of robots in the operative room is aimed at recovering the conditions of the open surgery while providing an important support in terms of accuracy and comfort for the surgeon during the accomplishment of the surgical task. The special kinematics of these robots is used to control them during the operations. In fact, the robot link which holds the surgical tools is constrained to move through the incision point, only by translating along its axis and rotating around the incision point.
The manipulator motion is constrained with respect to a point on the link axis known as Remote Center of Motion. Many mechanically constrained RCM manipulators have been proposed, but here we deal with a programmable RCM which represents a more flexible and space efficient solution which allows the use of the same manipulator both for open and for minimally invasive surgery.

This project shows the approaches and results obtained by solving the control problem of a specific task during minimally invasive surgery operations assisted by robot.
In particular, in this project we implemented and simulated the process of controlling the motion of a 7-dof robot according to the RCM constraint applied to the end effector: attached to the end-effector tool can be mounted either an endoscopic camera whose job is to help the surgeon with visual data during the operations, or specific surgical tools needed for the operation. The camera, or the surgical tool, is expected to entry inside the patient's body through the trocar, then a medical-mechanical device is used to drive the insertion of the tool inside the work area without causing, with accidental motions, possible damage to the patient.

Firstly, in our work, we decided to use as manipulator the KUKA LBR iiwa R820 which has seven revolute joints distributed in the mechanical design as follows: a spherical shoulder (3-DoF); an elbow (1-DoF); a spherical wrist (3-DoF). For this robot we computed both the kinematic and the dynamic model which allowed us to use this manipulator for achieving our tasks.

After that we have introduced the robot model, we started to define our strategies. The first control model that we implemented was a task control architecture which allows to drive the tool applied to the end effector toward the trocar point. Moreover, this control architecture leads to insert the tool respecting the trocar physical constraint, without damaging the patient's body.
Then we build up the dynamic model of the robot, using the Newton-Euler method, to detect and manage also the dynamic effects that take place during the operation: for example, the breath and the heart beating which cause a periodic movement of the patient's body which must be modelled to avoid injury to the body. In order to achieve these targets, we have implemented a residual method which takes as input the dynamic data from the dynamic model and then it returns the estimated forces which arisen from the contacts.

At the end, we show all the relevant numerical results and plots which allows to better understand how this robot can be used in such a context to obtain the best performance. We concluded our work with a view on a future work which may be applied.

1. KUKA LBR IIWA

The robot model which was chosen to implement all the different stages of the project is the KUKA LBR iiwa R820. This robot is an anthropomorphic arm providing 7-DoF: a spherical shoulder (3-DoF); an elbow (1-DoF); a spherical wrist (3-DoF).

Figure 1. Kuka LBR iiwa R820

The model of this robot is provided by the Matlab repository in the **.urdf** file called "*iiwa14.urdf*". It allows to manipulate the robot using its kinematic and dynamic characteristics, and then it provides a graphical model which can be viewed during the simulations.
We also computed the kinematic model and the dynamic model by hand. In fact, even though the Matlab model is good to be used for many different control tasks, for our purposes it creates some mismatch which doesn't allows to achieve our goal. This is why we decided to combine our computations of the dynamic and kinematic model, and the graphical interface of the Matlab model.
As we know, the dynamic model is not provided by the manufacturer, but its knowledge is indispensable for simulations and control based on the system model. Since we are just working in a simulation environment, we do not really need of the real dynamic parameters. In fact, a simulation on a real robot is not expected, thus we can approximate all the dynamic parameters because we are not constrained to respect the realistic aspect. Anyway, in order to have a satisfactory reference for the choice of the dynamic parameters, we decided to use the ones stored in the urdf file provided by Matlab.
Even though real dynamic coefficients are not required, we need to estimate the whole dynamic model for our seven revolute joints manipulator, in order to complete our tasks.

In the following sections, we are going to show how both the kinematic model and the dynamic model have been computed in order to achieve our tasks.

2. Kinematic Model

The computation of the kinematic model is indispensable for the development of all the stages of this project.
In fact, thanks to this model we are able to elaborate a kinematic control scheme which allows to move the robot according to the RCM constraint.
Successively, it will be used to elaborate a sort of ground truth for the estimation of the external forces applied on the RCM point positioned on the end effector of the robot (details in section 5.).

To compute the kinematic model, we decided to apply the Denavit Hartenberg method on the robot.
In the following figure, the robot is shown in its zero configuration, alongside with the Denavit Hartenberg reference frames:

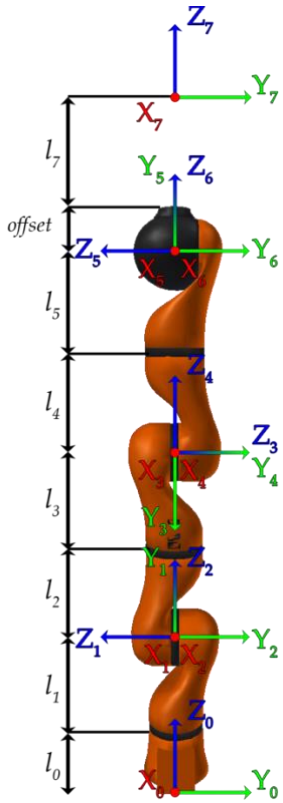


Figure 2. Kuka with DH reference frames

According to this set of frames, a table with all the DH-parameters was estimated as follows:

link	a_i	α_i	d_i	θ_i
1	0	$\pi/2$	$l_0 + l_1$	q_1
2	0	$-\pi/2$	0	q_2
3	0	$-\pi/2$	$l_2 + l_3$	q_3
4	0	$\pi/2$	0	q_4
5	0	$\pi/2$	$l_4 + l_5$	q_5
6	0	$-\pi/2$	0	q_6
7	0	0	$l_7 + offset$	q_7

In this table, we find the lengths of the different links which correspond to the following numerical values:

$$l_0 = 12.6\ cm$$
$$l_1 = l_2 = l_3 = l_4 = 20\ cm$$
$$l_5 = 19\ cm$$
$$l_6 = offset = 7.8\ cm$$
$$l_7 = 23.6\ cm$$

Where the length $l7$ is equal to the length of the tool applied to the end effector.

According to the Denavit Hartenberg formalization of the robot, we computed the kinematic model by estimating the homogenous transformation matrix from the base to the end effector:

$$^0T_7 = \quad ^0A(q_1)_1 \cdot {}^1A(q_2)_2 \cdot {}^2A(q_3)_3 \cdot {}^3A(q_4)_4 \cdot {}^4A(q_5)_5 \cdot {}^5A(q_6)_6 \cdot {}^6A(q_7)_7 =$$

$$\begin{pmatrix} & ^0R_7 & & p_7 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Successively, differentiating the position of the end effector with respect to all the joint coordinates, we computed the complete Jacobian of the robot, needed for our kinematic control system.
The nature of our task, which will be explained in detail in the following sections, required also the computation the Jacobian related to different points of the robot.

During the implementation of the different tasks of the project, we also used a kinematic model which was computed by inspection, in order to better understand if this kind of computation could provide some bigger error during the execution of the kinematic control task. We noted that it does not affect the solution, but since the dynamic model is computed basing on the DH model of the robot, we chose to use this version.

The steps for the computation of this matrix allowed to provide also the positions of each joint over the variation of the joint q-parameters:

$$p_0 = (0\ 0\ 0)^T$$
$$p_1 = (0\ 0\ l_0 + l_1)^T$$
$$p_2 = (0\ 0\ l_0 + l_1)^T$$
$$p_3 = \begin{pmatrix} -\cos(q_1) * \cos(q_2) * (l_2 + l_3) \\ -\sin(q_1) * \sin(q_2) * (l_2 + l_3) \\ \cos(q_2) * (l_2 + l_3) + (l_0 + l_1) * (l_2 + l_3) \end{pmatrix}$$

$$p_4 = \begin{pmatrix} -\cos(q_1) * \cos(q_2) * (l_2 + l_3) \\ -\sin(q_1) * \sin(q_2) * (l_2 + l_3) \\ \cos(q_2) * (l_2 + l_3) + (l_0 + l_1) * (l_2 + l_3) \end{pmatrix}$$

p_5

$$= \begin{pmatrix} (\cos(q_1) * \cos(q_2) * \cos(q_3) * \sin(q_4) - \sin(q_1) * \sin(q_3) * \sin(q_4) - \cos(q_1) * \sin(q_2) * \cos(q_4)) * (l_4 + l_5) - \cos(q_1) * \sin(q_2) * (l_2 + l_3) \\ (\sin(q_1) * \cos(q_2) * \cos(q_3) * \sin(q_4) + \cos(q_1) * \sin(q_3) * \sin(q_4) - \sin(q_1) * \sin(q_2) * \cos(q_4)) * (l_4 + l_5) - \sin(q_1) * \sin(q_2) * (l_2 + l_3) \\ (\sin(q_2) * \cos(q_3) * \sin(q_4) + \cos(q_2) * \cos(q_4)) * (l_4 + l_5) - \cos(q_2) * (l_2 + l_3) + (l_0 + l_1) * (l_2 + l_3) \end{pmatrix}$$

p_6

$$= \begin{pmatrix} (\cos(q_1) * \cos(q_2) * \cos(q_3) * \sin(q_4) - \sin(q_1) * \sin(q_3) * \sin(q_4) - \cos(q_1) * \sin(q_2) * \cos(q_4)) * (l_4 + l_5) - \cos(q_1) * \sin(q_2) * (l_2 + l_3) \\ (\sin(q_1) * \cos(q_2) * \cos(q_3) * \sin(q_4) + \cos(q_1) * \sin(q_3) * \sin(q_4) - \sin(q_1) * \sin(q_2) * \cos(q_4)) * (l_4 + l_5) - \sin(q_1) * \sin(q_2) * (l_2 + l_3) \\ (\sin(q_2) * \cos(q_3) * \sin(q_4) + \cos(q_2) * \cos(q_4)) * (l_4 + l_5) - \cos(q_2) * (l_2 + l_3) + (l_0 + l_1) * (l_2 + l_3) \end{pmatrix}$$

$$p_7 = \begin{pmatrix} (-\cos(q_1) * \cos(q_2) * \cos(q_3) * \cos(q_4) * \cos(q_5) * \sin(q_6) + \sin(q_1) * \sin(q_3) * \cos(q_4) * \cos(q_5) * \sin(q_6) - \cos(q_1) * \sin(q_2) * \sin(q_4) * \cos(q_5) * \sin(q_6) + \cos(q_1) * \cos(q_2) * \sin(q_3) * \sin(q_5) * \sin(q_6) + \sin(q_1) * \cos(q_3) * \sin(q_5) * \sin(q_6) + \cos(q_1) * \cos(q_2) * \cos(q_3) * \sin(q_4) * \cos(q_6) - \sin(q_1) * \sin(q_3) * \sin(q_4) * \cos(q_6) - \cos(q_1) \sin(q_2) * \sin(q_4) * \cos(q_6)) * (l_7 + l_6) + p_6(1) \\ (-\sin(q_1) * \cos(q_2) * \cos(q_3) * \cos(q_4) * \cos(q_5) * \sin(q_6) - \cos(q_1) * \sin(q_3) * \cos(q_4) * \cos(q_5) * \sin(q_6) - \sin(q_1) * \sin(q_2) * \sin(q_4) * \cos(q_5) * \sin(q_6) + \cos(q_1) * \cos(q_2) * \sin(q_3) * \sin(q_5) * \sin(q_6) - \cos(q_1) * \cos(q_3) * \sin(q_5) * \sin(q_6) + \sin(q_1) * \cos(q_2) * \cos(q_3) * \sin(q_4) * \cos(q_6) + \cos(q_1) * \sin(q_3) * \sin(q_4) * \cos(q_6) - \sin(q_1) * \sin(q_2) * \sin(q_4) * \cos(q_6)) * (l_7 + l_6) + p_6(2) \\ (-\sin(q_2) * \cos(q_3) * \cos(q_4) * \cos(q_5) * \cos(q_6) + \cos(q_2) * \sin(q_4) * \cos(q_5) * \sin(q_6) + \sin(q_2) * \sin(q_3) * \sin(q_5) * \sin(q_6) + \sin(q_2) * \cos(q_3) * \sin(q_4) * \cos(q_6) + \cos(q_2) * \cos(q_4) * \cos(q_6)) (l_7 + l_6) + p_6(3) \end{pmatrix}$$

3. Kinematic Control System

This is a task control with Remote Center of Motion constraint for minimally invasive robotic surgery.
Minimally invasive surgery assisted by robots is characterized by the restriction of feasible motions of the manipulator link constrained to move through the entry port to the patient's body. In particular, the link is only allowed to translate along its axis and rotate about the entry point. This requires constraining the manipulator motion with respect to a point known as Remote Center of Motion.
In the literature, many different kinds of mechanically constrained RCM manipulators have been proposed but obtaining a programmable RCM by constraining the motion of a serial robotic arm represents a more flexible and space-efficient solution also allowing the use of the same manipulator both for open and for minimally invasive surgery.

Here, we implemented a control scheme for the RCM constraint which manage the motion of the manipulator such that the RCM point on the end effector can achieve the Trocar point on the patient's body, according to the RCM constraint.

The following image shows a formalization of the robot performing the penetration of the patient's body according to the RCM constraint:

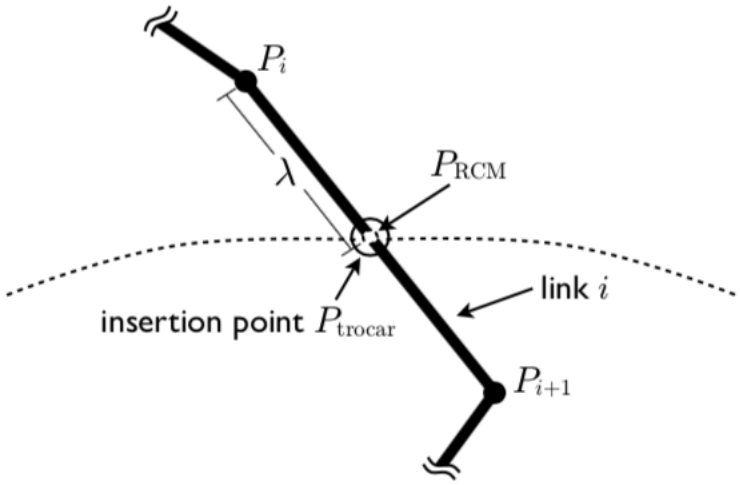


Figure 3. Penetration with RCM Constraint

We assume that the manipulator i-th link is constrained to pass through a fixed point $P_{Trocar} \in R^3$, which is the point where the surgical toll is inserted into the body through a trocar. The point on the link axis which corresponds to the P_{Trocar} is denoted by P_{RCM} and it represents the RCM for the manipulator, and, in principle, it can be located anywhere along the segment with endpoints P_i , P_{i+1} which represents the origin of the Denavit Hartenberg frames associated to the manipulator joints $i \; and \; i+1$ respectively.

In order to define the control design that we need to manipulate the robot to achieve the desired task, we can start by mathematically formalizing the Remote Center of Motion constraint.

Consider a serial manipulator with $n = 7$ joints where all the joints are assumed to be outside of the patient's body, and hence the RCM point can be positioned only on the end effector.
Assuming that:

$\boldsymbol{q} = (q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ q_7\ )^T$	Vector of joint variables
$\boldsymbol{p}_6(\boldsymbol{q}(t)), \boldsymbol{p}_7(\boldsymbol{q}(t))$	Cartesian coordinates of P_6, P_7 in manipulator base frame
$\boldsymbol{p}_{RCM}(\boldsymbol{q}(t))$	Coordinates of RCM point P_{RCM}
$0 \leq \lambda(t) \leq 1$	Portion of the end effector which is outside the body

Now, we can start formalizing the problem. Since the RCM is a variable point on the robot end effector at the entry point of the patient's body, its coordinates can change over time according to the manipulator motion, such that we obtain:

$$\boldsymbol{p}_{RCM}(\boldsymbol{q}(t)) = \boldsymbol{p}_6(\boldsymbol{q}(t)) + \lambda(t)(\boldsymbol{p}_7(\boldsymbol{q}(t)) - \boldsymbol{p}_6(\boldsymbol{q}(t)))$$

Differentiating with respect to time, we obtain the velocity of the RCM point:

$$\dot{p}_{RCM} = (\boldsymbol{J}_6 + \lambda(\boldsymbol{J}_7 - \boldsymbol{J}_6) \quad \boldsymbol{p}_7(\boldsymbol{q}(t)) - \boldsymbol{p}_6(\boldsymbol{q}(t))) \begin{pmatrix} \dot{\boldsymbol{q}} \\ \dot{\lambda} \end{pmatrix} = J_{RCM}(\boldsymbol{q}, \lambda) \begin{pmatrix} \dot{\boldsymbol{q}} \\ \dot{\lambda} \end{pmatrix}$$

The trocar condition requires that the cartesian coordinates of the RCM must be constantly equal to the insertion point (i.e. $\boldsymbol{p}_{RCM} \equiv \boldsymbol{p}_{Trocar}$).
Therefore, their derivative with respect to time must be equal to zero:

$$\dot{p}_{RCM} = J_{RCM}(\boldsymbol{q}, \lambda) \begin{pmatrix} \dot{\boldsymbol{q}} \\ \dot{\lambda} \end{pmatrix} = 0$$

The RCM constraint is the heart of the control system that we are going to develop. In fact, now that we have defined this feature, we can assemble the control design needed to achieve our task.

The control law which allow to control the robot and to achieve the convergence of the error is:

$$\begin{pmatrix} \dot{\boldsymbol{q}} \\ \dot{\lambda} \end{pmatrix} = {J_{RCM}}^{\#} K_{RCM}\, e + (I - {J_{RCM}}^{\#} J_{RCM})\, \omega$$

Where K_{RCM} is a positive constant which represents the gain. Whereas $e = (\boldsymbol{p}_{RCM} - \boldsymbol{p}_{Trocar})$ is the error of the task which should converge to zero.

The term ω reveals its importance only when we decide to insert an additional task for the control system. In this case, we get an "extended" form of the previous control law:

$$\begin{pmatrix} \dot{\boldsymbol{q}} \\ \dot{\lambda} \end{pmatrix} = {J_{ext}}^{\#} \begin{pmatrix} K_t & 0_{n_t\, x\, 3} \\ 0_{3\, x\, n_t} & K_{RCM} \end{pmatrix} e_t + (I - {J_{ext}}^{\#} J_{ext})\, \omega$$

Where K_{RCM} and K_t are two positive definite diagonal matrices of dimension $n_t x\, n_t$ and $3\ x\ 3$ respectively. The error becomes the error of the extended task $e_t = \begin{pmatrix} t_d - t \\ \boldsymbol{p}_{RCM} - \boldsymbol{p}_{Trocar} \end{pmatrix}$, and the Jacobian is no longer only referred to the RCM constraint but it is the extended task jacobian, and it guarantees decoupled exponential convergence of the task to the desired value.
The vector ω is a residual input projected in the null space of the Jacobian: it is useful to satisfy additional tasks like joint limits avoidance and limits on the variable λ modeling the penetration.

3.1.Implementation

Once that we described how to determine the control model which allows to achieve our task, and some additional ones, now we can apply this model to our robot by implementing in Matlab all the control system.

First of all, we have to define, as explained before, all the kinematic model of the robot with all the required parameters. Once that we have properly modeled our robot, we can move on by designing the world where the robot has to move. According to this purpose, we define a fixed three-dimensional point which corresponds to the position of the trocar. Moreover, we insert another three-dimensional point which is the target position of the RCM. Generally, it coincides with the Trocar position, but according to the task it can be chosen differently.

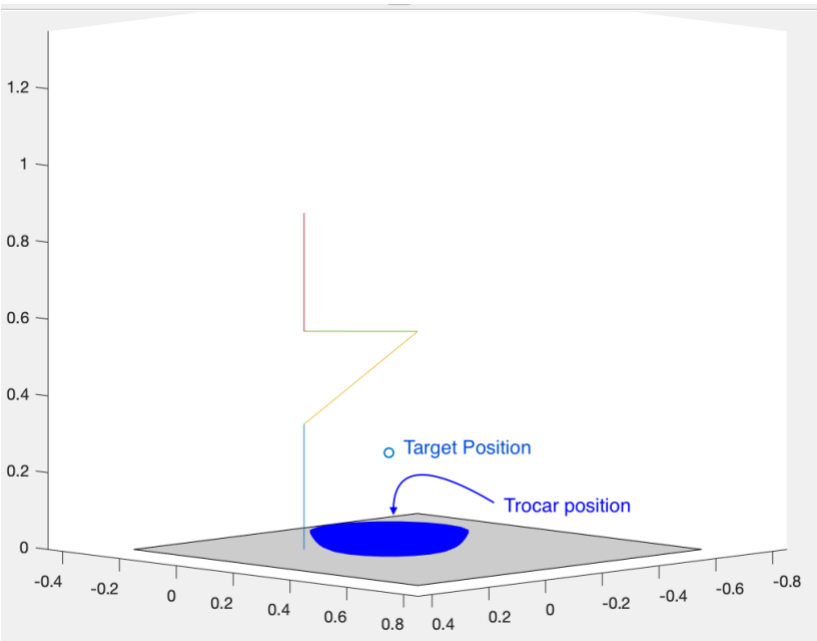


Figure 4. Simulation environment for kinematic control system

After the gains' setup for each different task, we can proceed applying the integration the control law: this integration is managed using the "ode45" Matlab function which allows to handle differential equations in a narrow way.

This function refers to another script where all the elements needed to compute the previously shown control law were properly set. In particular, the vector ω was introduce in the form:

$$\omega = -10 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \lambda - 0.2 \end{pmatrix}$$

Now that thanks to the integration we collected all the robot configuration needed to achieve the goal, we can visualize it using an animation.

There are two versions of the animation:

1. In the first one, the robot is represented as a set of concatenated segments, where each segment corresponds to a link between two joints.

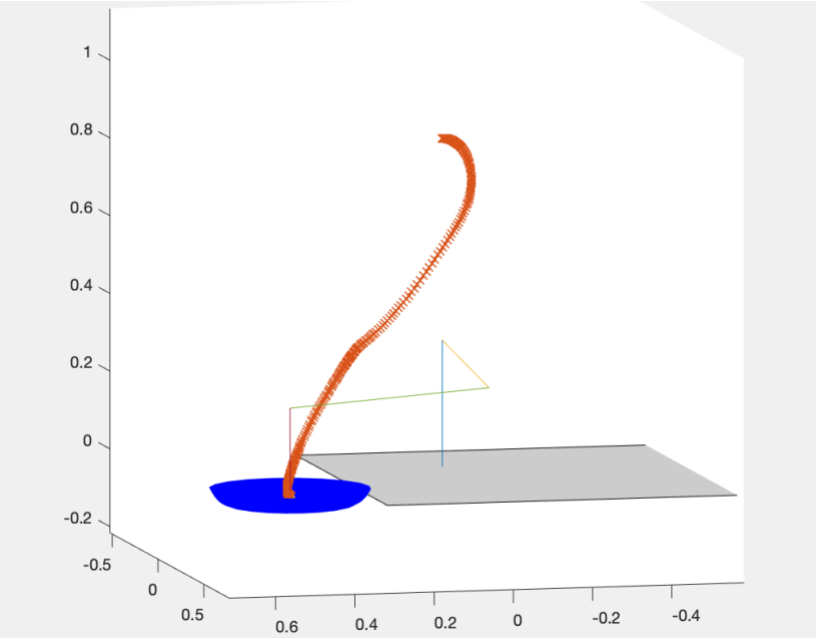


Figure 5. Geometric virtual model of the robot

2. In the second version, the robot is represented by using the graphical model associated to the "*iiwa14.urdf*" and adding to the end effector a segment which represents the needle of an arbitrary length.

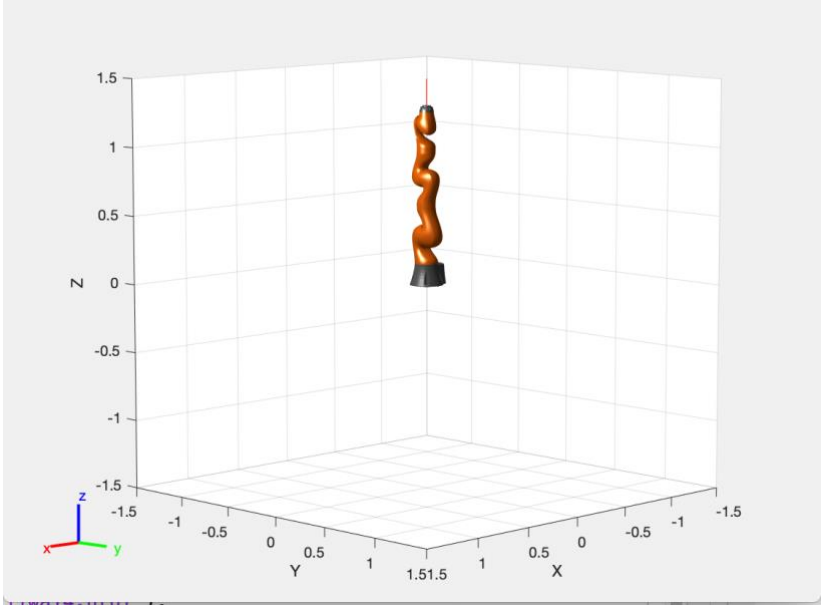


Figure 6. URDF Kuka virtual model

In both the animations, the trocar point is represented as the center of a blue cup on a table. This object was used only to simulate the patient's body next to the robot.

Since the Trocar is just a point, the robot is free reach this point from any direction, but in a real task it could be very dangerous for the patient. This is why we adopted two different strategies to avoid that risk:

1. The target position is different from the trocar position. In particular, the X and Y coordinates are the same, but the Z coordinates differs from each other in order to avoid collisions during the motion between the robot and the patient's body. It implies that after having reached the target position, the robot just has to perform a linear translation to reach the trocar.

2. Since the control law should guide the robot directly on the trocar position, we introduced an additional task. This task implies some constraints on the position of the last joint, such that the elements of the control law become:

$$J_{ext} = \begin{pmatrix} J_{RCM} \\ J_6 \end{pmatrix}$$

$$error = \begin{pmatrix} e_{RCM} \\ e_{ext} \end{pmatrix} \; with \; e_{ext} = \begin{pmatrix} Trocar_x - p_6(1) \\ Trocar_y - p_6(2) \\ (Trocar_z + l_7) - p_6(3) \end{pmatrix}$$

The desired position of the last joint was generally chosen as the target position in the first strategy, but it can also be set differently in order to have the end effector reaching the trocar point from another direction.

4. Dynamic Model

The next step of this work is about estimating external forces which act on the RCM point on the end effector when it is in contact with the patient's body at the trocar point. In order to achieve this goal, due to the absence of sensors on the RCM, we chose as strategy to use the residual method.
This method needs the computation of the dynamic model of the Kuka robot. This model can be computed in two ways:

1. Energy Based approach (Euler-Lagrange method)
2. Balancing of forces/torques approach (Newton_Euler method).

The first is based on the Euler-Lagrange equations which allow to show the relations on the robot among conservatives and non-conservative forces. It is expressed in symbolic/closed form and it's more suitable for robot with fewer joint.
Instead, the second method is more suitable for simulation and control purpose : indeed, equations are evaluated in a numeric/recursive way and so it is faster than the previous approach. Moreover, this method allow us, in spite of everything, to compute still the dynamic terms of the model. For our work we have implemented the Newton-Euler method to compute the dynamic terms of the model. So, we can write the dynamic model of the robot :

$$B(q)q'' + C(q,q') + g(q) = \tau \qquad (4.1)$$

The problem we are facing to, consists in a sensitive, soft interaction between robot and the human. To simulate the correct environment in which the robot works, we need to estimate the interaction forces that arise during the operation. For example, the presence of a fixed tool inside the patient's body (one of the most usage during minimally invasive surgery is for video information extraction during laparoscopic operation) should take into account the external forces coming from the natural body's movements, like breathing and heart beating.
So, in order to get a complete version of the dynamic model which characterizes the robot, we have to introduce the effects caused by the presence of the external forces acting on the robot.

Finally, we can compute the full dynamic model of the robot:

$$B(q)q'' + C(q,q') + g(q) = \tau + J_c{}^T(q)\, f_{\text{ext}} \qquad (4.2)$$

with : $q \in R^n$ joint position vector, $q' \in R^n$ joint velocity vector, $q'' \in R^n$ joint acceleration vector, $B(q) \in R^n \; x R^n$ the inertia matrix, $C(q,q') \in R^n$ the centrifugal and coriolis vector , $g(q) \in R^n$ the gravity vector, $J_c \in R^n \; x R^n$ the jacobian matrix computed at the contact point of the robot and then $f_{ext} \in R^6$ the vector of external contact forces (first three) and external moments (last three).

Knowing the RCM point, we can compute then the Jacobian matrix in this point, but to compute the fully dynamic we need an estimation of external forces. This issue is explained in detail in the next section where, using only proprioceptive sensors we are able to compute an evaluation of these forces.

5. Force Estimation: Residual Method

Physical Human-Robot Interaction (pHRI) is a challenging requirement that comes from the cohabitation and collaboration between the humans and the robot. In particular, this interaction involves forces exchanges, that needs to happen without any issue or concern. Within pHRI we can distinguish two phases:

- Collision: the system has to understand if the contact with the environment, or the human user, is by chance or not.

- Reaction strategy: consequence of the first case.

In this project, we present a strategy which can be applied to achieve the collision check during the robot motion. With regards to the Reaction strategy, we will just discuss about some theoretical case that could be considered to complete the solution of the problem.

A novel framework for collision detection, and consequent control strategy was presented in [3,4]; the main idea is to see a collision as a fault in the actuation system of the robot. In this scenario authors applied fault detection and isolation (FDI) techniques based on "generalized momenta", that only requires proprioceptive measurements. We resume these techniques and apply them to our subject of study.
For systematizing the contact handling problem, we introduce a unified framework entitled the collision event pipeline, which aims at embracing all relevant phases collisions may undergo. In our implementation there are three main phases:

1. **Detection Phase:**
   It has to control if a collision occurred between robot and environment : in our simulations, the movements of the patient are assumed to always be present since the end-effector of the robot come inside the trocar.

2. **Isolation Phase**:
   Based on the task of identify the collision link and point. In our problem both of them are known: they are respectively the last link and the RCM point.

3. **Identification Phase**:
   Starting from a reconstructed signal based on robot dynamic we are able to reconstruct the external forces acting on the RCM point. To do this we use the momentum observer method.

The monitoring method based on the generalized momentum observer was motivated by the desire of avoiding the inversion of the robot inertia matrix, decoupling the estimation result, and also eliminating the need of an estimate of joint accelerations. First we define the generalized momentum associated to eq.(4.2) defined as $p = M(q)q'$ and follows the first-order equation :

$$p' = \tau + {J_c}^T F_c + C^T(q, q')q' - g(q)$$

Then we define our residual signal, the "residual vector" $r \in R^7$, which is our monitoring signal able to reconstruct the additional external torques acting on the robot structure and we refer to it as τ_{ext}:

$$\boldsymbol{r = K_0(p(t) - \int_0^t (\tau_m - \beta(q, q') + r) ds + p(0))}$$

$$\text{where } \beta(q, q') = g(q) - C^T(q, q')q'$$

Here $K_0$is a diagonal gain matrix. The dynamic relation between the external joint torque τ_{ext} and r is:

$$r' = K_0(\tau_{ext} - r)$$

In other words, the filter equation is a stable, linear, decoupled, first-order estimation of the external collision joint torque τ_{ext} .

In fact, for sufficiently large gains, we can assume that :

$$r \cong \tau_{ext} = J_c{}^T(q)\, f_{\text{ext}}$$

Starting from the residual signal r, we can reconstruct the external forces :

$$f_{ext} = (J_c{}^T(q))^{\#} r$$

where $(J_c{}^T q))^{\#} = (J_c J_c{}^T)^{-1} J_c$

In the code the residual identification is compute in this part :

```
sumDyn = sumDyn + (tau - beta)*deltaTime;
sumRes = sumRes + (output*deltaTime);
output = K*(B*qd - sumDyn -sumRes - fv_vec )/(1+K*deltaTime);
```

Here, given the dynamic information from the robot, we compute the integral in a discrete way. This section of code is located inside the main loop of the program, so that the residual can be updated at each iteration.

6. Simulation Results

In this section, we are going to show the numerical and graphical results that our system produced over the variation of some parameters.
We divided our experiments into two sections according to how we developed our work:

1. The first section illustrates the results obtained by the kinematic control system that we implemented and explained in section 3. This section shows how the robot can change its behaviors by varying the main parameters.
   We identified these main parameters as the gain, and how the additional task is set. In fact, we will see that in order to evaluate the correct behavior of the robot, we are going to change the additional task of the robot such that the contact with the patient's body would occur from different poses/orientations.

2. The second section will show the results that we have obtained by applying the residual method to the robot when the end effector is in contact with the patient's body.
   This part will reveal the robustness of the residual method during force estimation over the variation of different parameters such as: the gain, the variation of the typology, intensity and frequency of the forces and moments, and critical situation (empirically identified) acting on the system.

We will show the validity of these results and in the last section of this work we will suggest how they can be used in order to sketch a reaction strategy of the robot.

6.1 Results: Kinematic Control System

In section 3, we illustrated how the kinematic control system was implemented in order to allow the robot's end effector to reach the Trocar position avoiding harming the patient's body.
Now, we examine how this approach is affected by the variation of some important parameters which characterize it:

- **Gain (K)**: it is the gain used during the control system. It is a positive matrix, computed as:

$$\begin{pmatrix} K_t & 0_{n_t x\, 3} \\ 0_{3\, x\, n_t} & K_{RCM} \end{pmatrix} where\ n_t = 3\ and\ then\ both\ gains\ are\ 3x3\ matrices$$

  The values that we have chosen to use for the analysis of the system are K = 0.1, 1, 10, 100.

- **Additional task**: as we have seen in section 3, the additional task is used in order to avoid that the robot motion could cause some collision with the patient's body before of inserting the needle into the Trocar. In order to achieve this goal, the additional task constrains the sixth joint position, but this approach allow to make different choices.
  These different choices lead the end effector to be inserted into the patient's body using different robot's poses ad orientations and thus also the orientation of the end effector will change with respect to the insertion point.
  These variations of the task have been encoded as follows:

  A. The (x,y) coordinates of the joints are equal to the Trocar ones but the z coordinates is the trocar one plus the length of the end effector (**Contact task**).

  B. The (x,y) coordinates of the joints are equal to the Trocar ones but the z coordinates is the trocar one plus the length of the end effector divided by ten (**Deep insertion task**).

  C. The joint coordinates are set such that we don't get perpendicular insertion, but it is sloped of a certain angle (**Diagonal insertion task**).

  According to these different settings, all the result are going to be reported.

6.1.1 Results: Kinematic Control System – Gain

The analysis of the kinematic control system starts by varying the value of the gain applied to the control scheme.
The standard additional task which will be used during all these simulations is the "Contact task".
The first result is produced by setting the gain to the lowest value that we have chosen: K = 0.1.

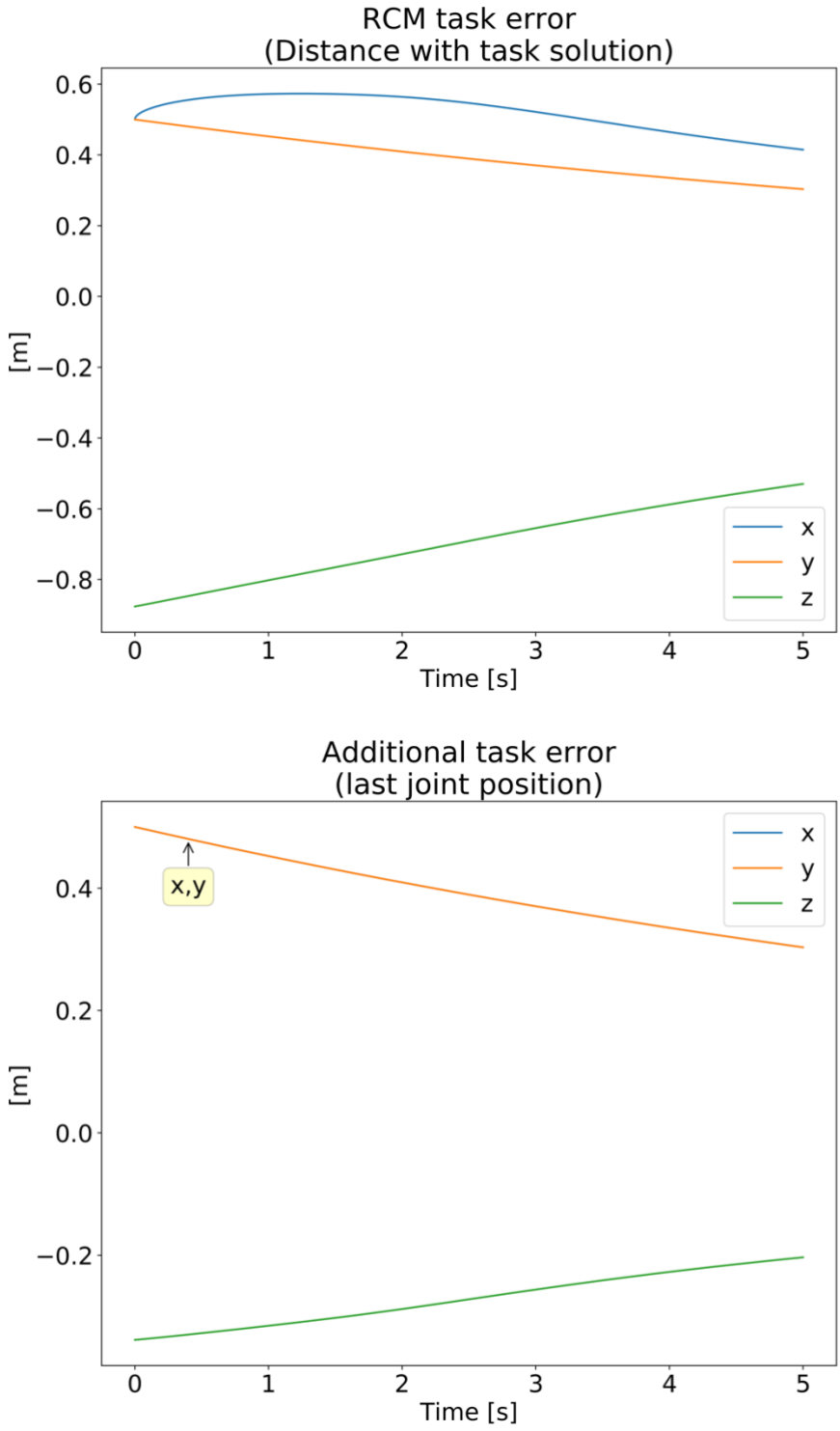


Figure 7. Task results with K = 0.1:(a) RCM Task components; (b) Additional Task Components

In these graphs, we have seen the evolution of the error component both for the RCM task and for the additional task. They all show that using this control scheme with such a low gain is not a good strategy. In fact, the system is too weak, and the error is not able to converge to zero. It means that the task is not achieved.

Setting K = 1, we can evaluate the second leg of results. As before, the following graphs are going to show the evolution of the error for the achievement of the two tasks.

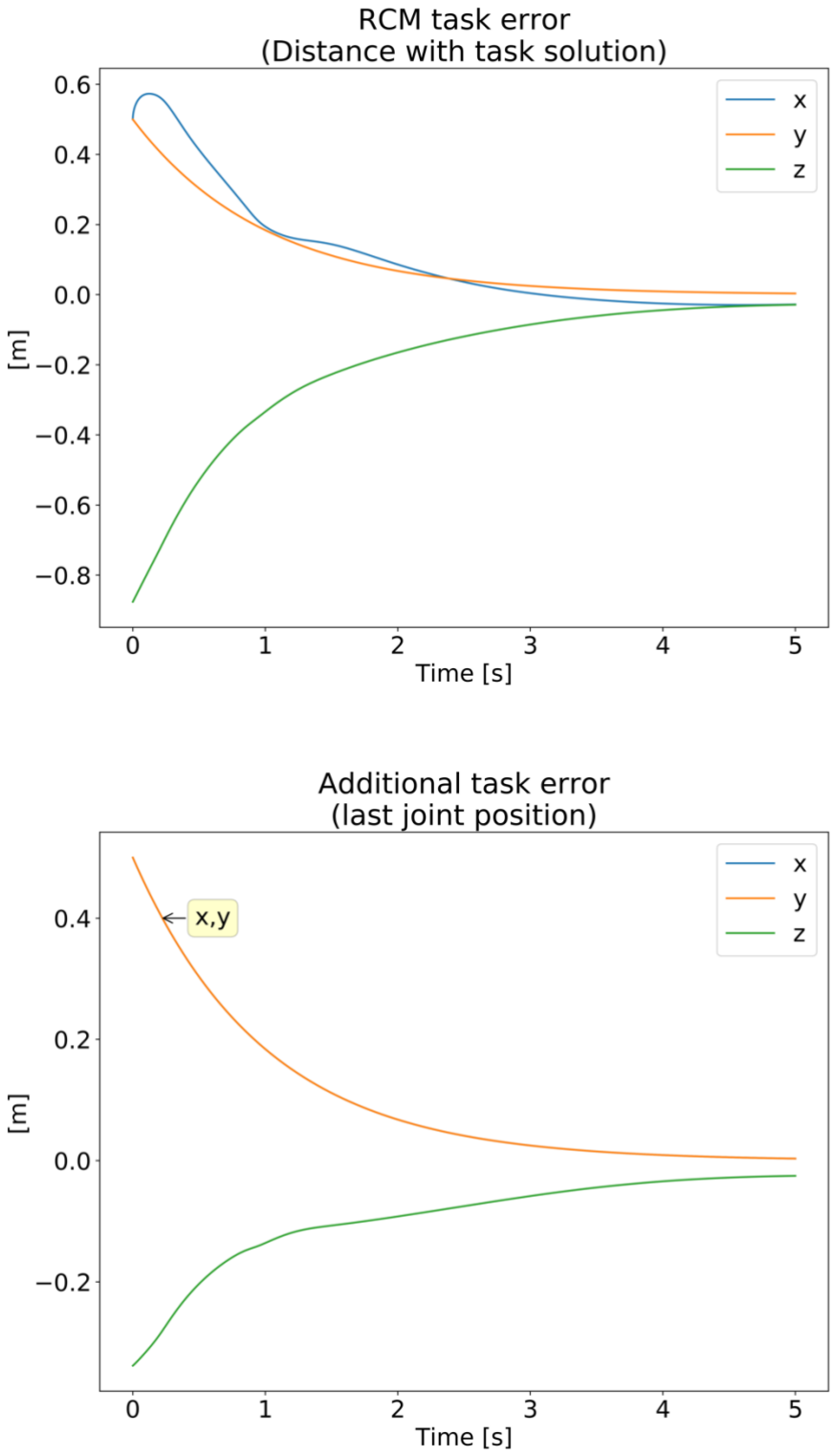
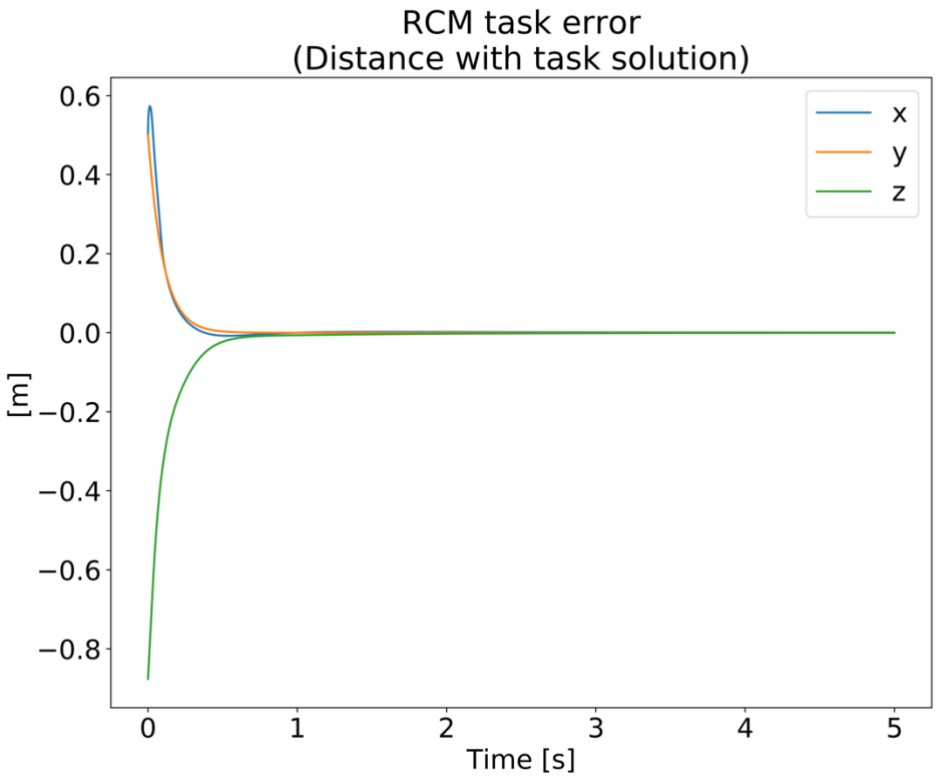


Figure 8. Task results with K = 1:(a) RCM Task components; (b) Additional Task Components

It is immediately evident that increasing the gain from 0.1 to 1 produced very better results with respect to previous ones. In fact, most of the component can converge before of the end of the time interval. Although convergence is achieved for some components, for some other we get different results. For example, the z component of the additional task is not able to stabilize to get to convergence. So do the z component of the RCM error. It is possible to see also that the x component of the RCM error is very unstable, and it never converges. The difference of the behavior between the additional task error component and the RCM task error components highlights the fact that using different values of the gain the two tasks could be an advantageous approach.

If we increment the gain up to K = 10, we can note that it works much better.



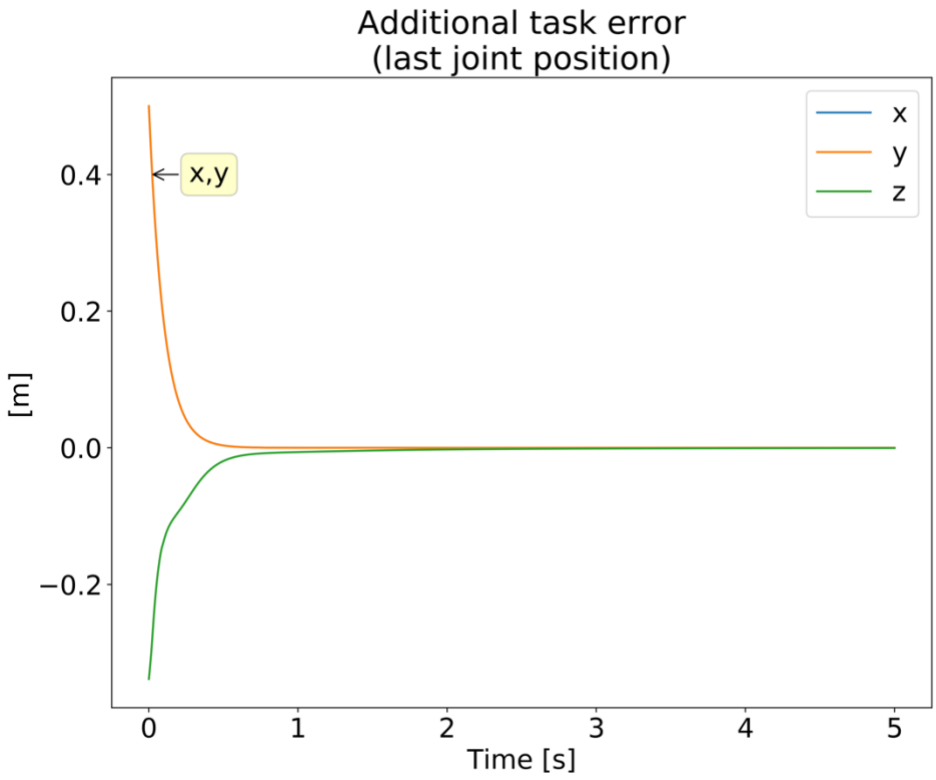
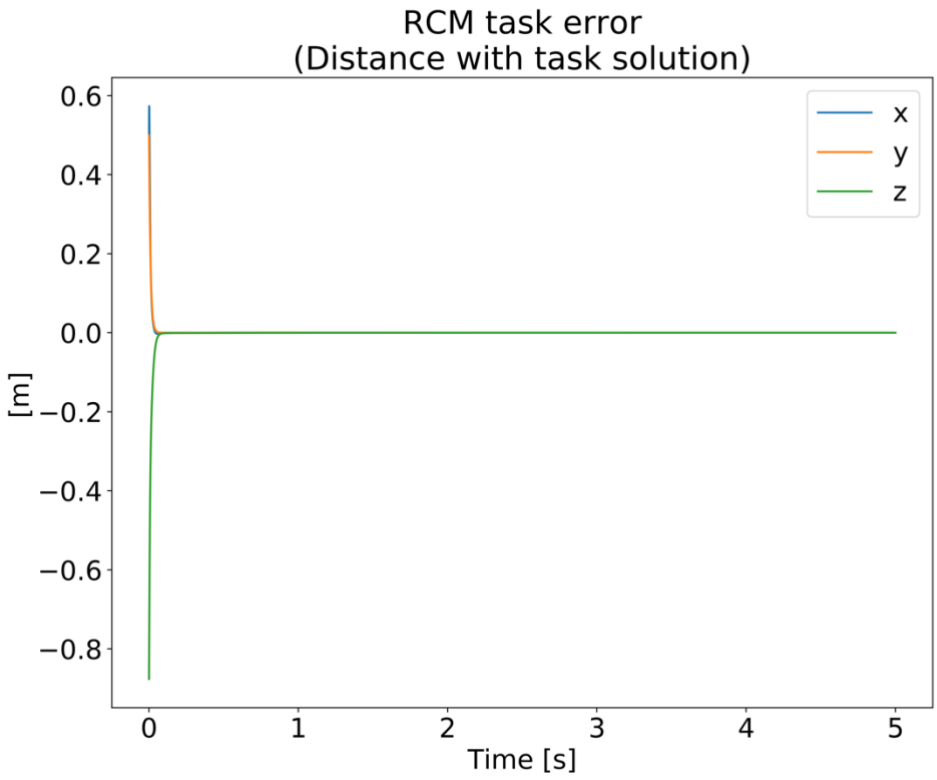


Figure 9. Task results with K = 10:(a) RCM Task components; (b) Additional Task Components

These plots shows how all the previous problem are solved by using this value for the gain. The convergence is fastly achieved for all the errors components.

As expected, these results are respected also when we set K = 100.
The graphs show that here the convergence is faster with respect to the previous case, but in our opinion it is not a good solution because for the elaboration of the control design it requires much more time, and it could cause some important problem in real time applications.



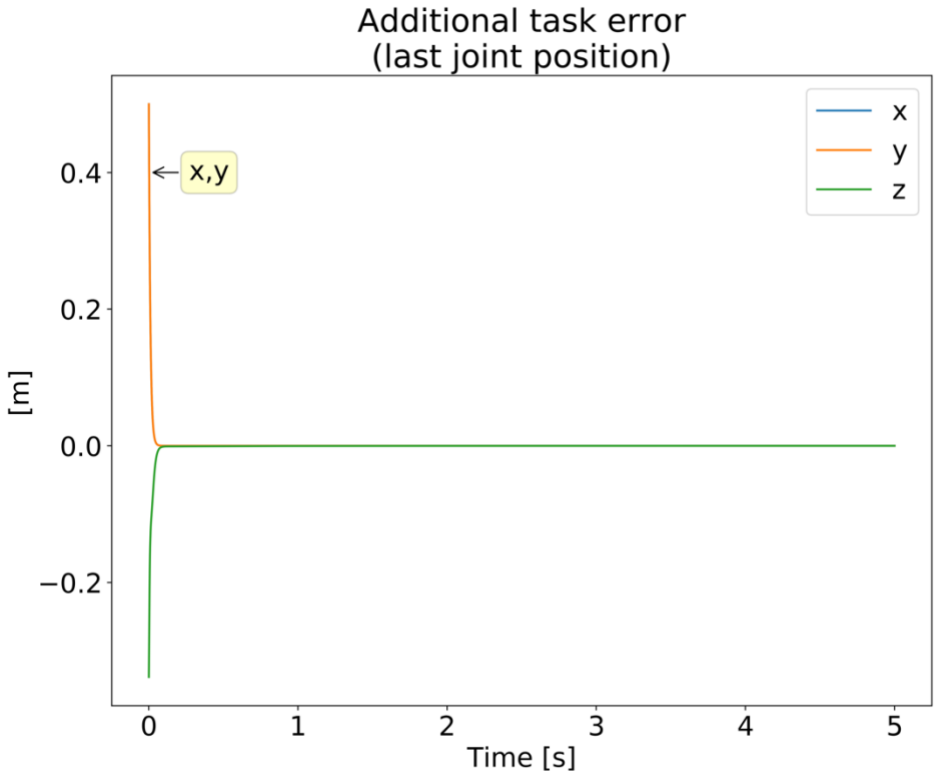


Figure 10. Task results with K = 100:(a) RCM Task components; (b) Additional Task Components

6.1.2 Results: Kinematic Control System – Additional Task

As said in the previous section, the additional task was varied in order to analyze how the control system works with different final configurations. Before of examine the results we should make two main assumptions:

- All the additional task are supposed to be executable according to the main task. In fact, if they would not be, the control system would not work, and it would get stuck on loop.

- In order to check the quality of these results, we set the gain to K = 10, because, as it was shown, this is the best choice for the gain in this control system.

Now, we can proceed with the presentation of these results. The first one was already presented in the previous section. In fact, all the previous simulations was run using the "Contact Task" as additional task. From the figure 11, we can see the final configuration of the robot to perform this task in addition to the main task. The robot achieves the final pose without problems, and the error properly converges to zero for both tasks.

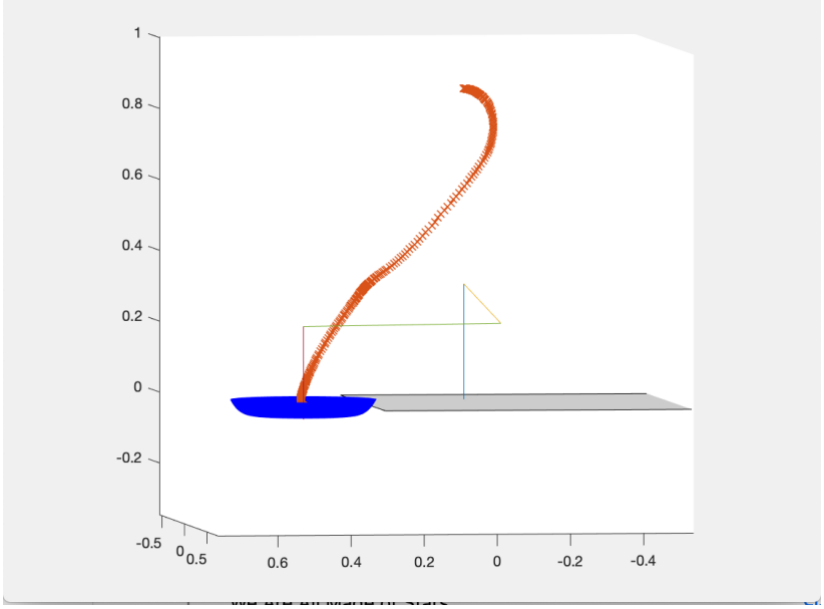


Figure 11. Task achievement with "Contact Task"

The second simulation was run using the "Deep insertion task" as additional task. It could be useful when the robot has to insert all the surgical tool inside the patient's body in order to reach a deep point to execute its task. Figure 12 shows the almost total insertion of the needle inside the trocar point.

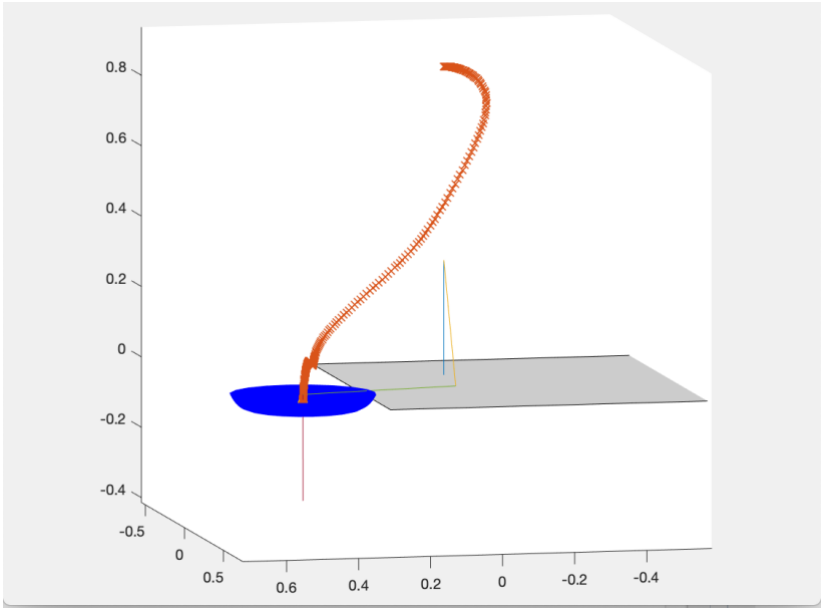
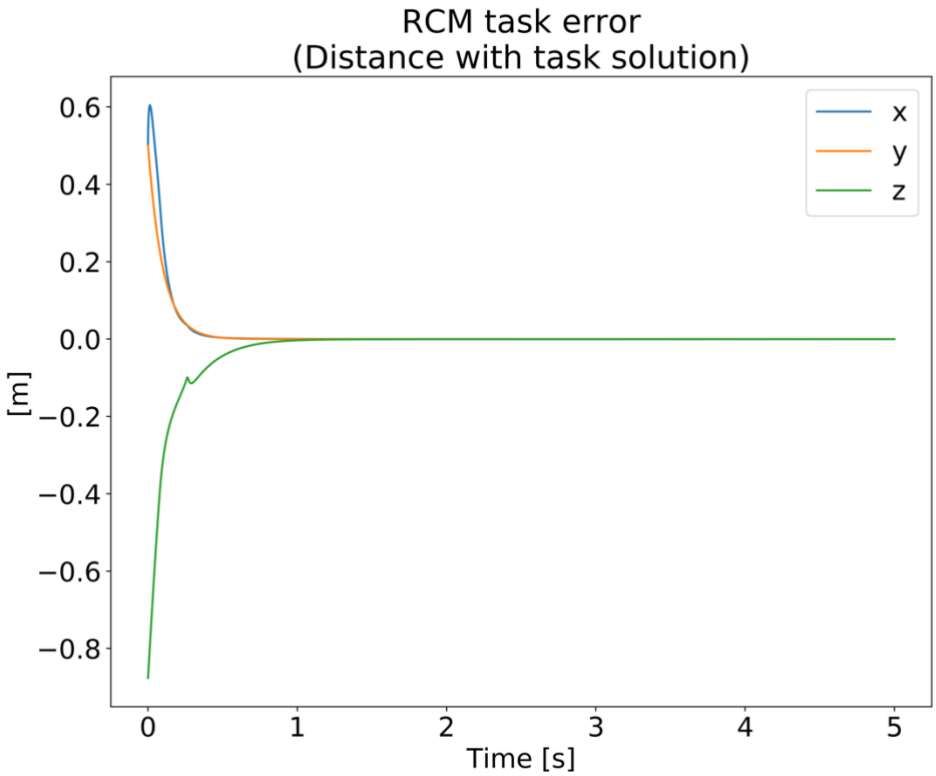


Figure 12. Task achievement with "Deep Insertion Task"

In the following graphs we can see how the robot behaves when it has to achieve this task in addition to the primary one.



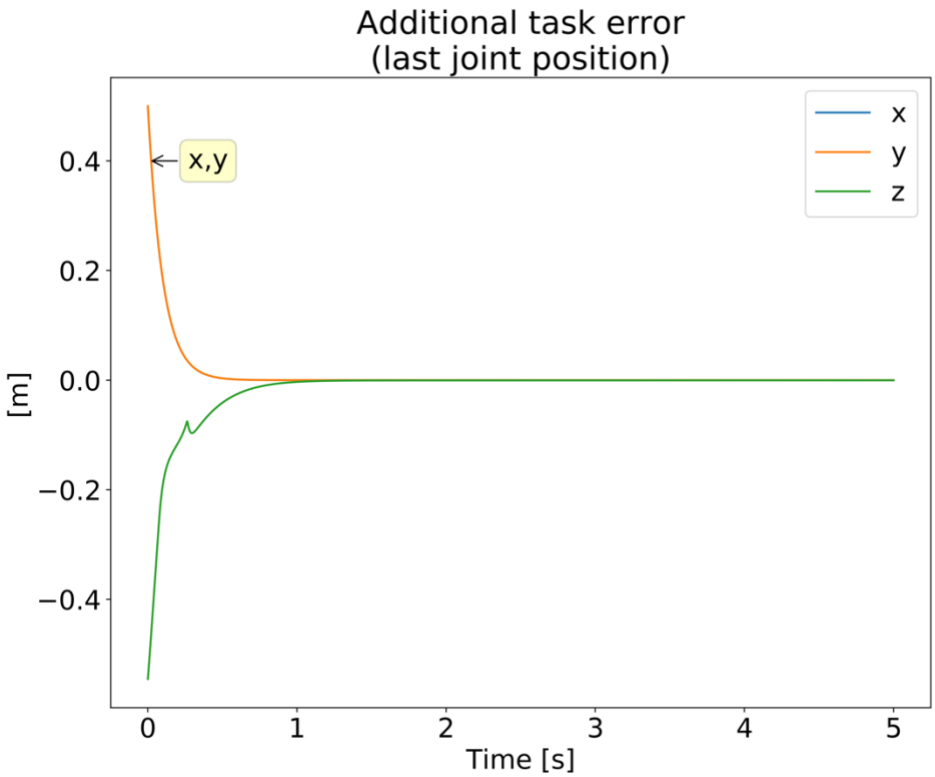
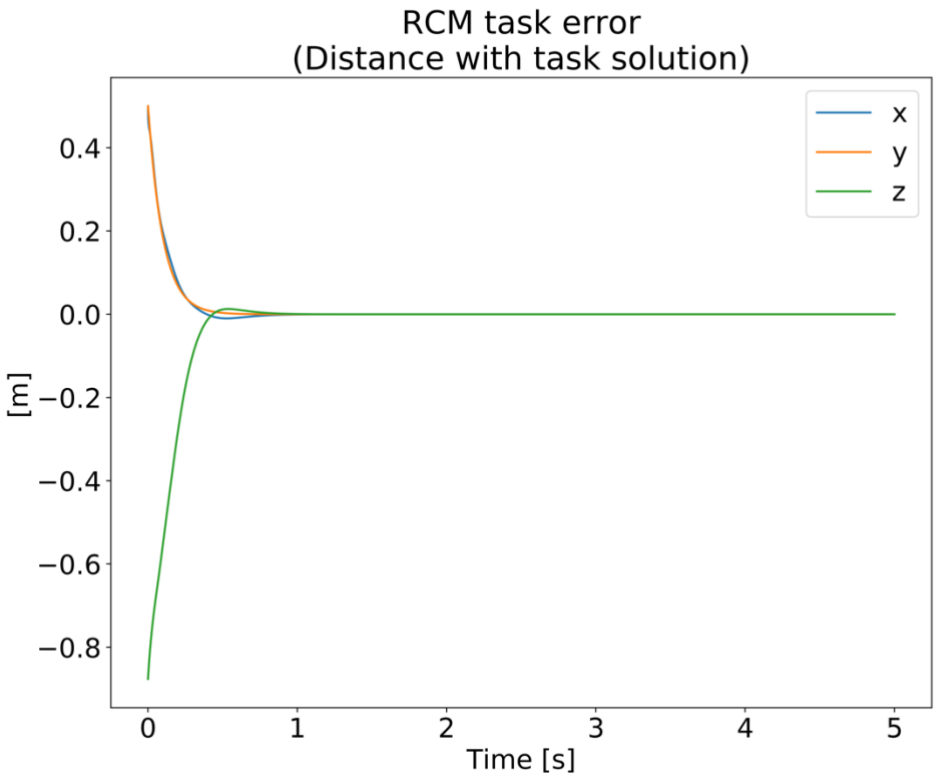


Figure 13. Task Results with Deep Insertion Task

We can see that, except for some adjustments, during the motion the task are correctly achieved and their errors converge to zero.
The important thing to highlight is that these adjustments happen before that the needle gets in contact with the body. If these movements would occur when the needle is inside the body, it could harm the patient. In order to avoid these adjustments, it could also be introduced some correction strategy (e.g. virtual fixtures).

The last graphs show the behavior of the system when the additional task is the "Diagonal insertion task". In many surgical applications the insertion is not needed to be orthogonal to the body, but it is required a sloped insertion to achieve the task. This is why we introduce this last additional task, and as before we take care that the robot does not harm the patient.



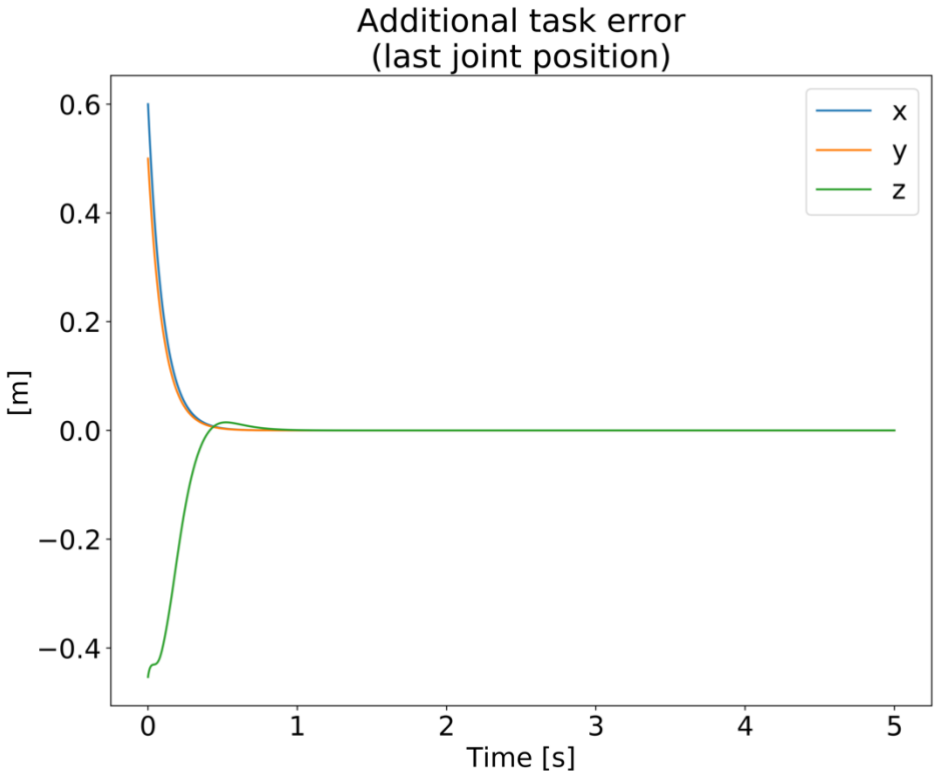


Figure 14. Task Results with Diagonal Insertion Task

It works better than the previous task, even though there is still some small adjustments which can be solved as before, or by increasing the gain. This last figure shows the insertion of the robot using this additional task.

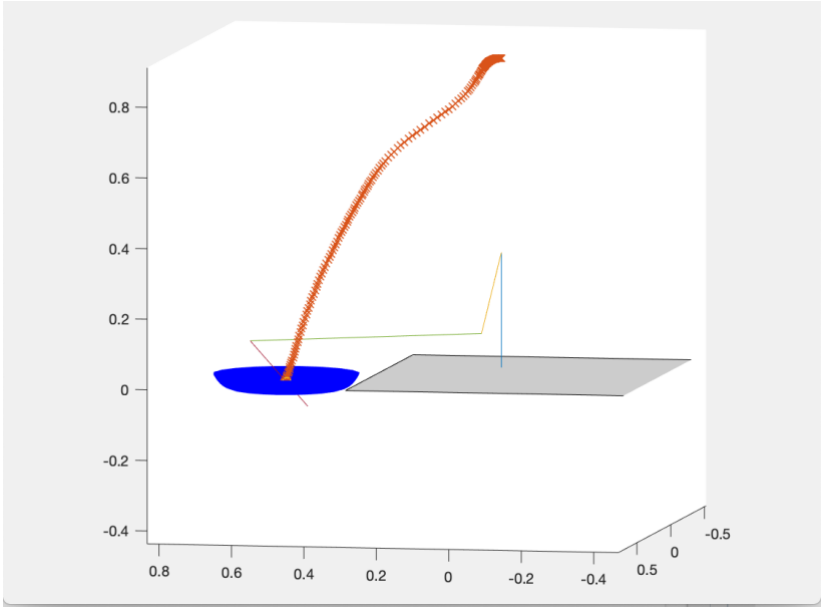


Figure 15. Task achievement with "Diagonal Insertion Task"

6.2 Results: Force Estimation

This section provides all the results which have been examined for the force estimation when the robot[1] is in contact with the patient's body.
As explained before, this estimation is computed according to the residual method. Since we already know that the contact point on the robot is the RCM point, then we do not need to define a strategy which returns the point of application of the forces. It means that, using this method, we just have to estimate the external torque applied on the robot, and then, through inverse differential kinematics computation, we will report the estimation of the external force.
The practical aim of these experiments is to be able to identify the intensity and the dynamic of a force applied to the robot during the surgical operation, such that it can react and avoid harming the patient. These forces which could be applied during operations can be produced from the heartbeat and the breath of the patient, and it highlights the fact that we are not dealing with a special case, but we are interested to solve a common practical problem.

In this section we demonstrate how the dynamic system, which computes the force estimation, can change when we vary some important parameters. Indeed, in order to better understand how the behavior of the system can be affected, we run several simulations which describes the following scenarios:

1. Gain variation
2. Different types of external forces
3. Different frequencies to apply to variable external forces
4. Combination of forces and moments
5. Introduction of noise

In all these situations we chose to evaluate the norm error among estimated forces and real forces as the main parameter of quality. In fact, this parameter will return the accuracy of the estimation method in all the scenarios[2].

[1] The correct operation of the system was tested both with dynamic parameters shown in the previously cited .urdf file and with the dynamic parameters estimated in [6].
[2] In this section only the most significative results will be shown. The whole set of results is kept inside the Google Drive Folder

6.2.2 Results: Force Estimation – Gain

During the simulations we experimented eight different values for the gain **K**. According to the theory illustrated in the previous section, the residual signal must converge to a value similar to the external torque if the gain K is sufficiently large.

Proceeding in increasing order, we set these values for the gain :

$$K = [0.1, 1, 10, 100, 500, 1000]$$

For this simulation we are assuming that the applied force is a periodic force on the x-y plane of maximum intensity of 5 N, in order to deal with a force which can be encountered also during a surgical operation.

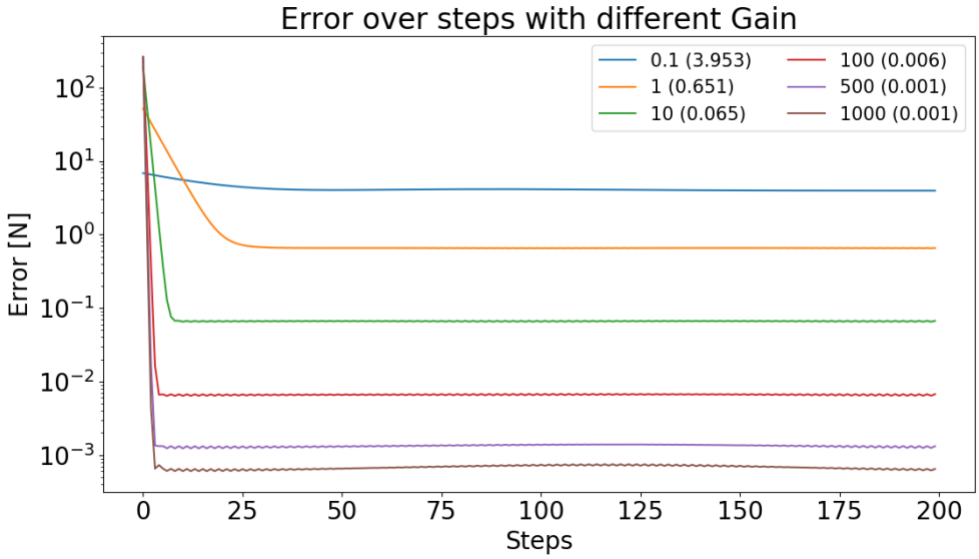


Figure 16. Estimation Forces Error over the gain variation ($F_{ext} = [5 * F_x, 5 * F_y, 0, 0, 0, 0]$)

Starting from the first experiment (K=0.1), we can immediately see that the convergence of the error is too slow and, moreover, it converges to a constant value equal to ≈4 and this value is too big since we want to achieve a better accuracy.
Moreover, both the torque estimation and the force estimation are not good. In fact, even though the dynamics of the estimated torque has a shape that is similar to the real one, the real values, both over the intensity of the applied force and the time correspondence are too different.

Increasing the gain value up to K = 1 implies a clearly better behavior of the system: the error starts to converge faster and it achieves . This is not the only good results which can be achieved by increasing the gain. In fact, with respect to the previous case, after an initial transient interval, both forces and torque estimation reveal a dynamic which is very close to the real one.

Setting the gain to K=100, the initial transient interval is eliminated because using this gain value the system is faster to estimate the external forces applied to the robot.

This result can be highlighted in any plot of the simulation. In fact, now the error converges to values of the order of 10^{-2}.

The maximum value tested for the gain is, as shown before, 1000. These experiments highlight the fact that, assuming there is no noise which affects the system, the more you increase the gain the more the error convergence value decreases. In fact, in this ideal scenario, we can say that if K tends to infinity, then the error tends to zero.

Gain	Min	Max	Mean
0.1	3,952788	6,839799	4,228807
1	0,650577	51,658156	1,883913
10	0,065028	187,743505	1,378218
100	0,006373	253,230065	1,323298
500	0,001230	261,315367	1,318343
1000	0,000602	262,362275	1,317729

6.2.4 Results: Force Estimation – External Forces

In this section, we evaluate how the system reacts when constant or variable forces of various intensity are applied to the robot.
We apply to the RCM point two forces acting on x-y plane. Since they are easier to analyze and learn we consider first the constant case : we use 3 different intensity for the forces [0.5 N, 5 N, 50 N] to show what extent the robot is able to reconstruct the correctly dynamics and estimation forces.

The intensity in surgery situation must be very lower and controlled, so we start from what could be a typical force induced by breathing, heart beating of small movements of the patient, and for this we set an intensity equal to 0.5 N.
We also considered more particular situations, when during a surgical operation a force equal to 5 N is applied to the robot due to the use of particular equipment, and we evaluated the behavior of the system.
For the sake of clarity and safety of the patient during the operation, we have also tested the system under more unexpected, unusual and hard situations which could be provoked either by external accidental impact that caused a movement in the patient or humans mistakes. This is represented by the example where the force intensity is set to 50 N.

In all these simulations we choose a gain value equal to 1: even though this is not the best gain value to use, it offers the possibility to evaluate smooth changes among the different scenarios of these simulations, and then it highlights how the intensity of the force can affect the system.

a) Constant forces

Setting constant forces intensity equal 0.5 N, the estimation results are almost identical to the real ones. We can see that the error starts immediately to converge to a value very close to zero.

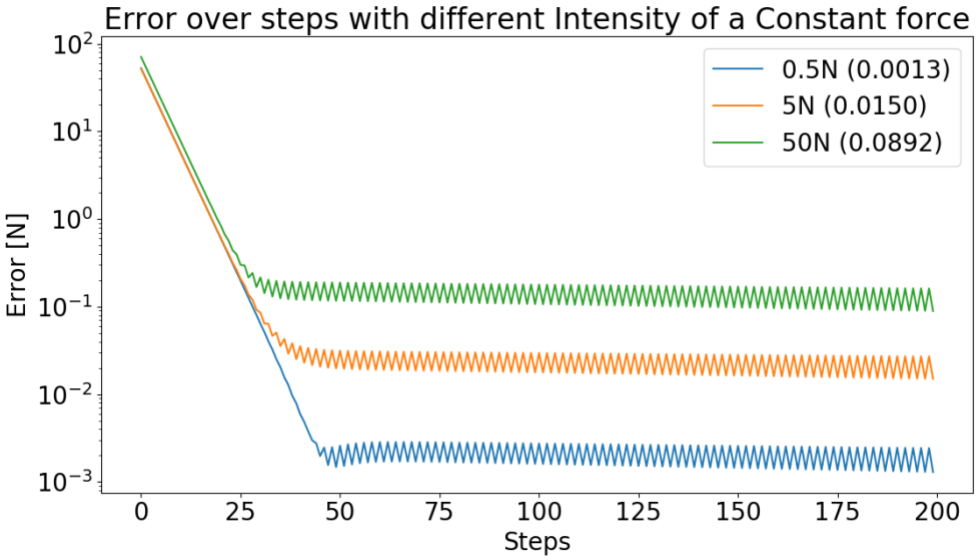


Figure 17. Estimation Forces Error $(F_{ext} = [\alpha * F_x, \alpha * F_y, 0, 0, 0, 0])$

In fact, the estimation torque and the estimation forces results are very reliable.
After that we evaluated the first case, we simulated the constant forces using the other two intensity values.
It is immediately evident that there are no big differences with respect to the first case. In fact, the error still converges to values very close to zero and it means that the estimations are always very similar to the original data.
We analyzed the results even when we face a unusual case, which is represented a force with an intensity equal to 50 N. We can show that the error is higher with respect to the previous cases, but anyway the error is still acceptable and it means that also in this case this system provides good results.

Intensity (Constant Force)	Min	Max	Mean
0.5	0,001296	52,594961	1,316441
5	0,014978	52,068486	1,322424
50	0,089184	70,790542	1,890309

b) Variable forces

Since the forces that we want to model during a realistic scenario are periodic forces, for the simulation test with variable forces we used trigonometric functions with argument the current time step. Moreover, we will change the frequency of these periodic movements to observe how the force estimation process works in more critical situations.

Firstly, we evaluate how the system behaves when the intensity of a variable force increases over the experiment.
We use as a reference sin e cosine functions acting on a x-y plane, and we set a period of (x/30).

Assuming these settings, our first simulation is run with an intensity equal to 0.5 N, as in the constant case.
Even these experiments show a very good estimation that can also be used to implement a robust reaction strategy. These results are very similar to the one that we encountered when we analyzed the constant case. It means that, up to now, estimating a variable force is not different with respect to estimating a constant force.

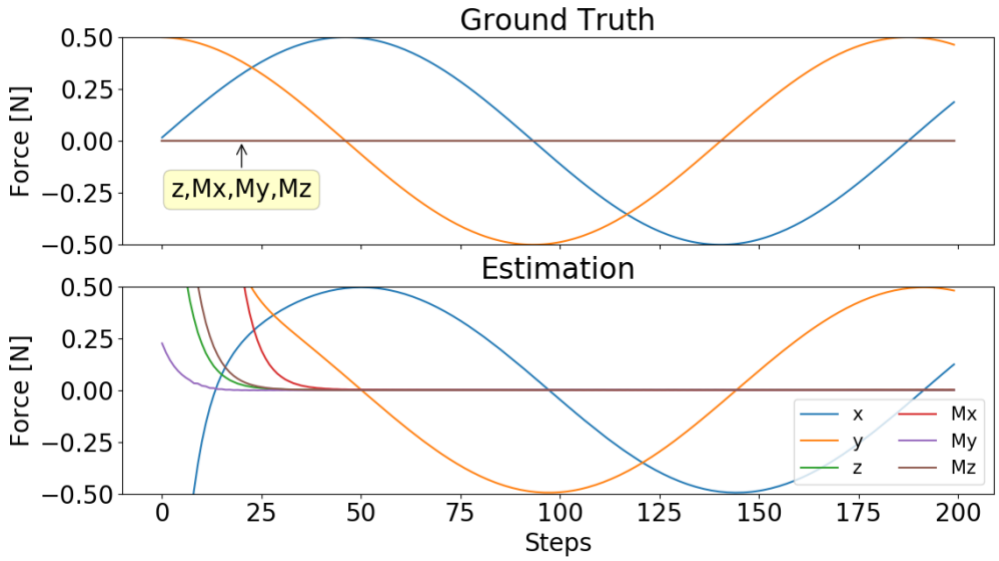


Figure 18. External Variable Forces Estimation: $\alpha = 0.5$ ($F_{ext} = [\alpha * F_x, \alpha * F_y, 0, 0, 0, 0]$)

After this first test, we set the intensity of the force equal to 5 N. This is the case we already faced to during the evaluation of the gain. This experiment highlights immediately the fact that the behavior of the system changes when we increase the intensity of a variable force. In fact, we still get very good results but the error convergence value increased with respect to the previous case.

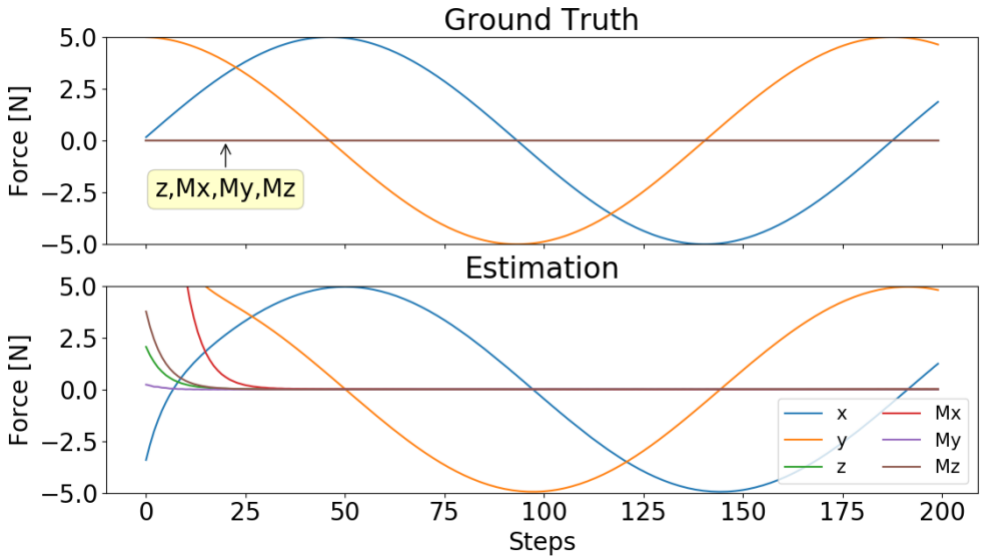


Figure 19. External Variable Forces Estimation: $\alpha = 5$ ($F_{ext} = [\alpha * F_x, \alpha * F_y, 0, 0, 0, 0]$)

This last graph confirms these considerations. It shows the dynamics of the error when we apply a variable force with an intensity equal to 50 N, and even in this situation we can note that the error convergence value is still increased.

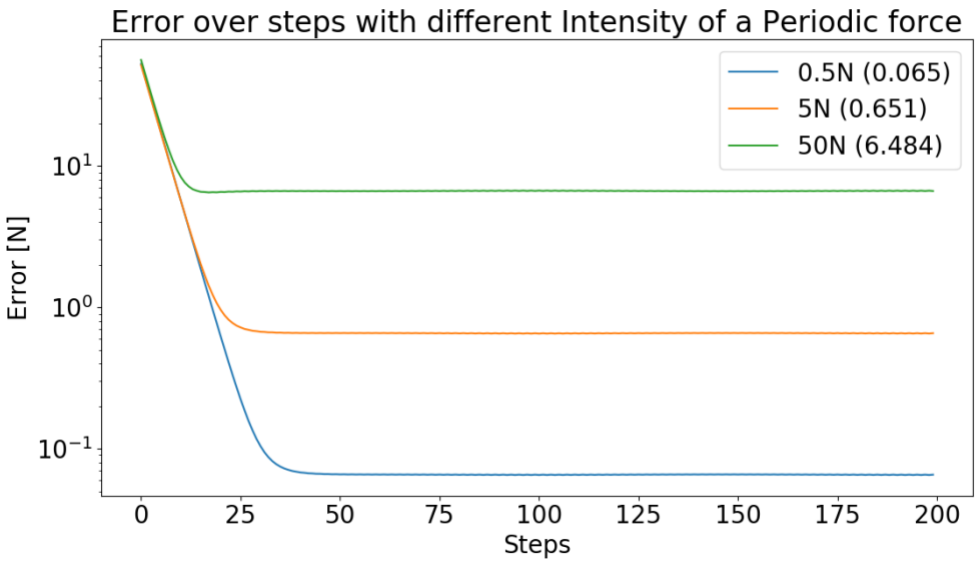


Figure 20. Estimation Forces Error
($F_{ext} = [\alpha * F_x, \alpha * F_y, 0, 0, 0, 0]$)

Intensity (Variable Force)	Min	Max	Mean
0.5	0,065093	52,568181	1,371074
5	0,650577	51,658156	1,883913
50	6,484325	56,086349	7,606115

6.2.6 Results: Force Estimation – Forces and Moments

Finally, we want to demonstrate the accuracy of this method and how it is able to deal with forces and moments acting on the robot RCM.
To achieve this goal, we have tested our system with 4 different kind of forces/moments acting in different arrangement : only forces; only moments; one force and one moment; and all forces and moments components. All these experiments have been tested for constant and variable forces.

$$F1 = [Fx\ , Fy, Fz, 0, 0, 0]$$
$$F2 = [0, 0, 0, Mx, My, Mz]$$
$$F3 = [Fx, 0, 0, Mx, 0, 0]$$
$$F4 = [Fx, Fy, Fz, Mx, My, Mz]$$

Testing the only forces case with constant forces, we obtained good results as in previous experiments, and thus we considered that it is not relevant to show the results that we obtained using constant forces.
Contrarily, we want to highlight how different compositions of variable forces can affect the system.

For all these simulations we used the gain value set to K = 1, for the same reasons that we discussed in the previous section. Moreover, the intensity of the external forces that we used for these tests is 5 N.

Starting from the case where we apply only a three components force to the tool linked to the end effector of the robot, we can note that it is not much different with respect to two components case.
In fact, the error achieves about the same convergence value even though it assumes a slightly more oscillating behavior. Also the results of the estimation of the external torque are very good.

The case where we use an external force composed only by three moments highlights more or less the same features that we have encountered in the previous example, where we only applied forces components.
We can note the same oscillating behavior on the forces estimation error but it is able to achieve a smaller convergence value.
Successively, we have tested the system under the application of an external force characterized by one force component and one moment component. Even this case is not affected by any relevant problem. Moreover, in this case, the estimation error loses the oscillating feature that we have seen before. It may highlight the fact that a force applied along the z direction can cause a little oscillation on the error estimation.

At the end, we have tested the full external input vector composed by both forces and moments.
This last case resembled all the previous examples and it confirms all the considerations that we have done until now.

This section of results helped to understand that this estimation system which uses the residual method, is able to properly estimate different typologies of forces, along different directions.

The following table resembles all the data that we used to analyze these cases:

Force Typology	Min	Max	Mean
F1 Constant	0,016143	52,064777	1,323422
F2 Constant	0,006194	48,900932	1,230417
F3 Constant	0,004410	49,342536	1,238705
F4 Constant	0,019585	48,233698	1,230758

Force Typology	Min	Max	Mean
F1 Variable	0,016143	52,064777	1,323422
F2 Variable	0,006194	48,900932	1,230417
F3 Variable	0,004410	49,342536	1,238705
F4 Variable	0,019585	48,233698	1,230758

6.2.7 Results: Force Estimation – External Forces Frequency

For the sake of realism, we decided to analyze how the behavior of the system when we apply different frequencies to the variable external forces.
We decided to implement this case in order to simulate the dynamic of the heartbeat during surgical operation. The approximation is based on the assumption that the heartbeat follows the dynamic of a sinusoidal function: the systole of the heart corresponds to the positive part of the sinusoid function, whereas the diastole is the negative part such that the whole sinusoid represent an entire heartbeat.

In order to evaluate this aspect of the system, we introduced two likely cases which can be faced during surgical operations:

1. Total anesthesia was not given to the patient and then its heart rate is normal. We considered an average value equal to 70 bpm.

2. Total anesthesia is given to the patient and then its effect depends on the kind of drug that he assumed.
   In fact, according to the following table [5], the effect of the heart rate depends on the drug which is used during the operation:

Drug	**Heart Rate**
Propofol	No significant change
Pentothal	Increases 10-40%
Etomidate	Minimal effect
Ketamine	Increases
Fentanyl	Decreases
Morphine	Decreases
Succinylcholine	Variable (usually decreases)
Halothane	No change
Isoflurane	Decreases
Sevoflurane	Decreases

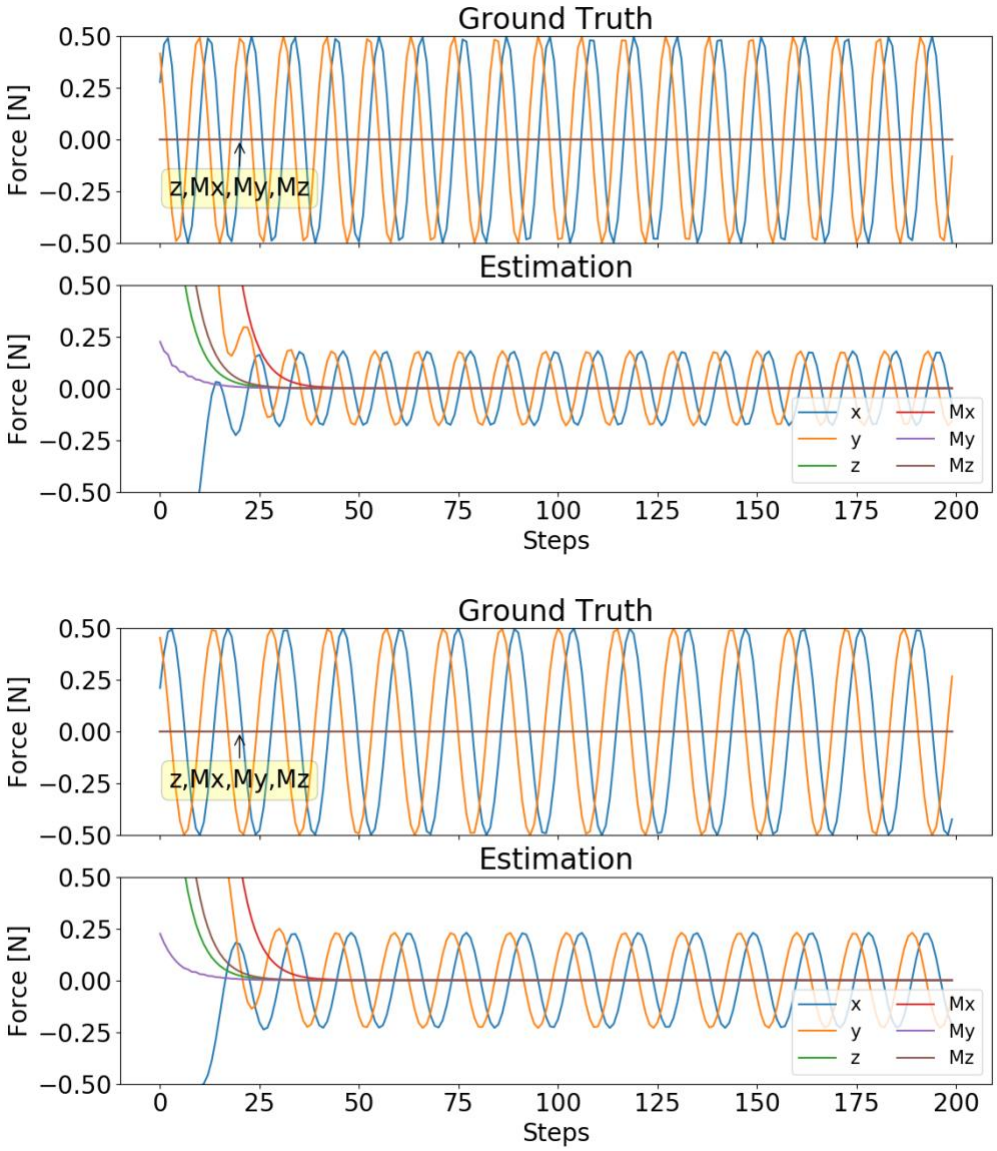
   As we can see, most of the drugs do not produce any relevant effect or they decrease the heart rate. Whereas, if you decide to use the Pentothal, then the heart rate is increased of a valued which goes from the 10% to the 40% of the normal heart rate (assumed as 70 bpm).

In order to reproduce in the simulation environment a frequency which corresponds to the assigned value, we computed the period of the sinusoidal function by inspection according to the time of the system.

The first case, as said before, represent a patient in normal condition with a stable value of 70 bpm.

The second case represent the patient when assumes the Pentothal for anesthetic aims. We assumed that the heart rate is increased of 40% with respect to the normal case, and then it reaches a value equal to 98 bpm.

The third case was introduced in order to analyze in detail the case when the total anesthesia is distributed to the patient. In fact, we have considered the assumption of other common drugs which reduce the heart rate up to 50 bpm. Even in this case, all the previous considerations are confirmed.



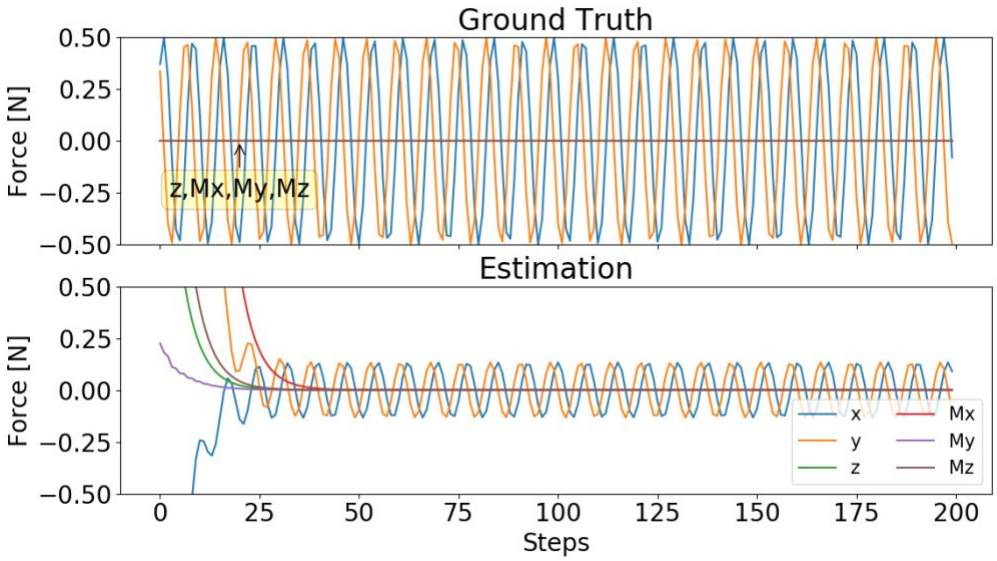


Figure 21. Forces estimation: (a) 50 bpm; (b) 70 bpm; (c) 100 bpm

We can immediately note from the graphs, that the error on the estimation of the forces is increased with respect to the previous cases. Despite this fact, we can see the system we implemented can still return very good results both in the estimation of the forces and in the estimation of the external torque.

The following graph highlights the considerations that we have made for the previous case: indeed, the more you increase the frequency and the worse the system behaves, but also in this situation the residual method is able to return very reliable results.

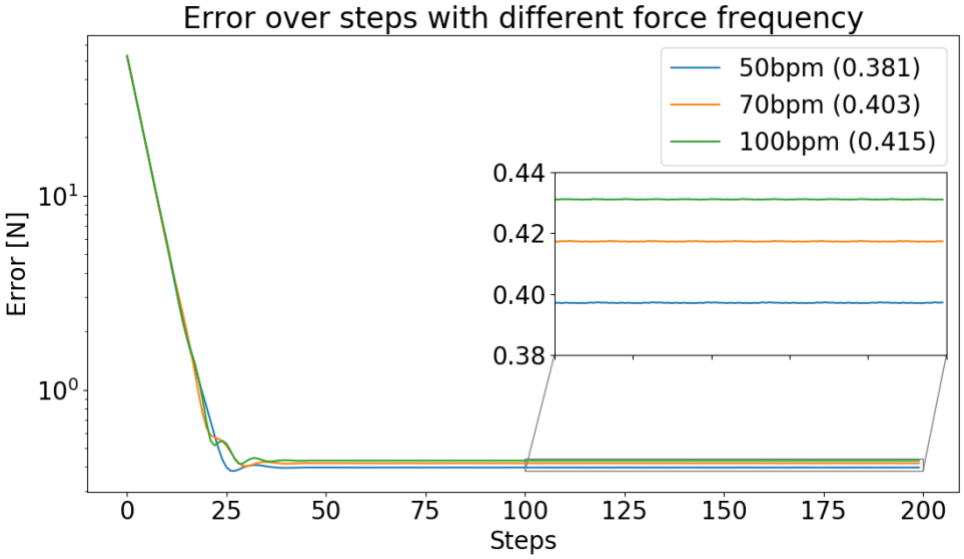


Figure 22. Estimation error in all the three cases

Assuming that the heart wave is not a sinusoidal but it has a more complex shape, then we can deduce that also small increments of the error due to the delay of the estimation system can cause important problems.

These plots show interesting reactions of the system to different levels of frequency: even though the error remains always low, we can note that increasing the value of the frequency, the error increases too and the system is not yet able to reconstruct perfectly the signal due to a delay of the system generated during the estimation of the forces.

6.2.5 Results: Force Estimation – Estimation with Noise

In the previous case, we analyzed the application of variable external forces using different frequencies in order to better simulate a surgical operation environment introducing some realistic feature.
Now, we decided to analyze the response of the system when we add a new realistic element which generally affects a real robot. Using Matlab Communication Systems Toolbox features we applied a white gaussian noise with power equal to 1 to the each of the simulated measurements of the control torque of the robot.

We decided to evaluate all the gain cases that we analyzed before with the aim of finding a good value which improves the behavior of the system but does not amplify the noise.

The following graphs gives an overview of these results in order to better understand how the noise affects the measurements of the residual method.

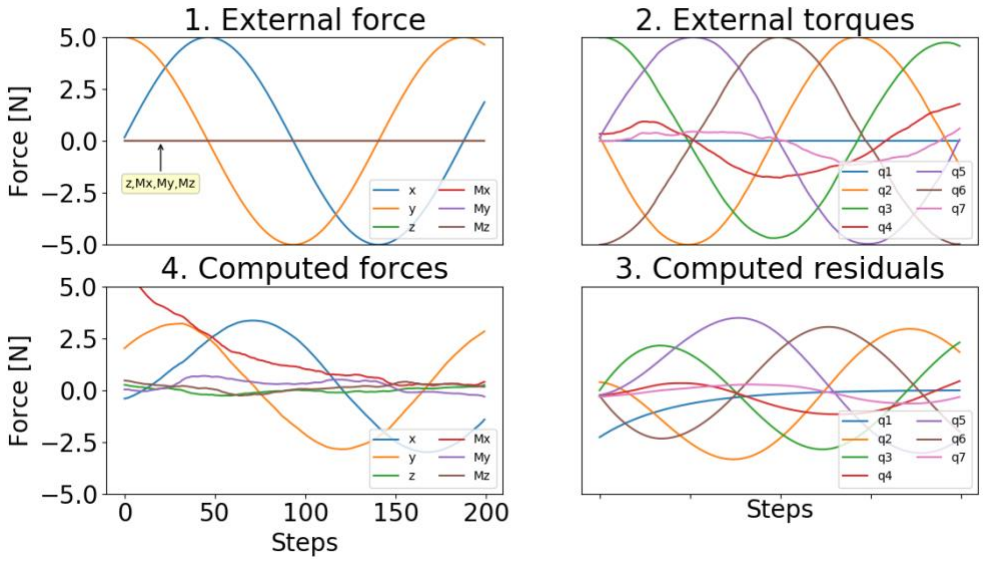


Figure 23. Estimation results with noise: K = 0.1 ($F_{ext} = [5 * F_x, 5 * F_y, 0, 0, 0, 0]$)

In these first graphs we can see that the estimation is really worse than in the case with no noise. In fact, now the system is not able to return results which can be used in order to understand the dynamic of the external forces.
The system tries to follow the right motion in the joint space, but the low values of K make it too slow and it is not able to reach the right amplitude of the signals.

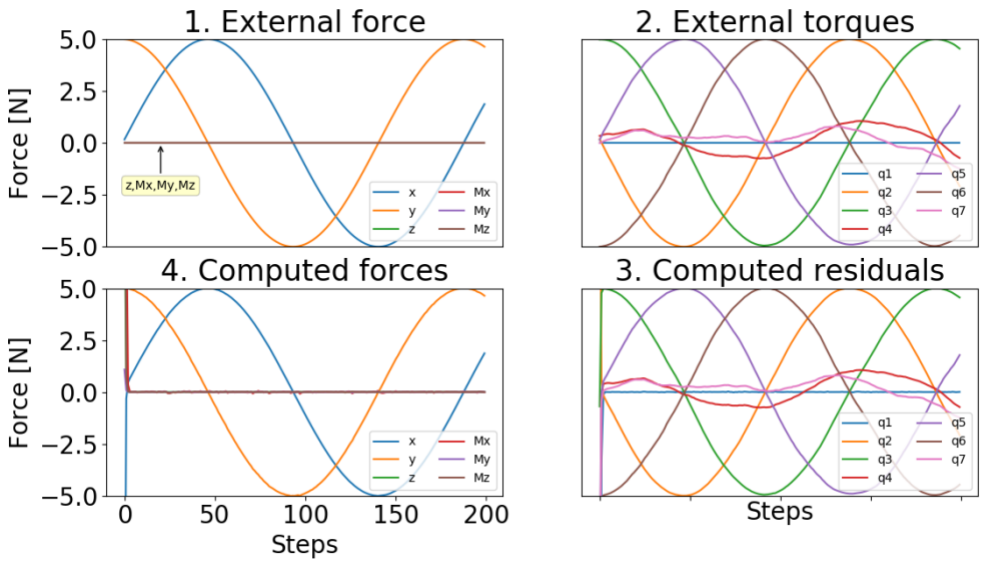


Figure 24. Estimation results with noise: K = 100 ($F_{ext} = [5 * F_x, 5 * F_y, 0, 0, 0, 0]$)

The more we increase the gain and the better the residual method is able to return the good results that we have seen also in the previous sections. But, now, we can see that the noise error now affects a small interval of the signal: its disturb is more focused on a local portion of the signal with respect to respect previous experiments.

This final graph show the estimation error over the variation of the gain. It is possible to see that for lower values of the gain, the error is too high.
When we increase the value of the gain, the error starts to decrease as desired. Differently from the previous cases, when we have the noise which affects the system, it is no longer an ideal scenario and then we have to set the gain to a perfect value. In fact, as we can see from the graph increasing too much the gain value leads to get an unstable estimation. It could cause many practical problems during a surgical operation, and this is why we prefer to set the gain value to a medium value like K = 10.

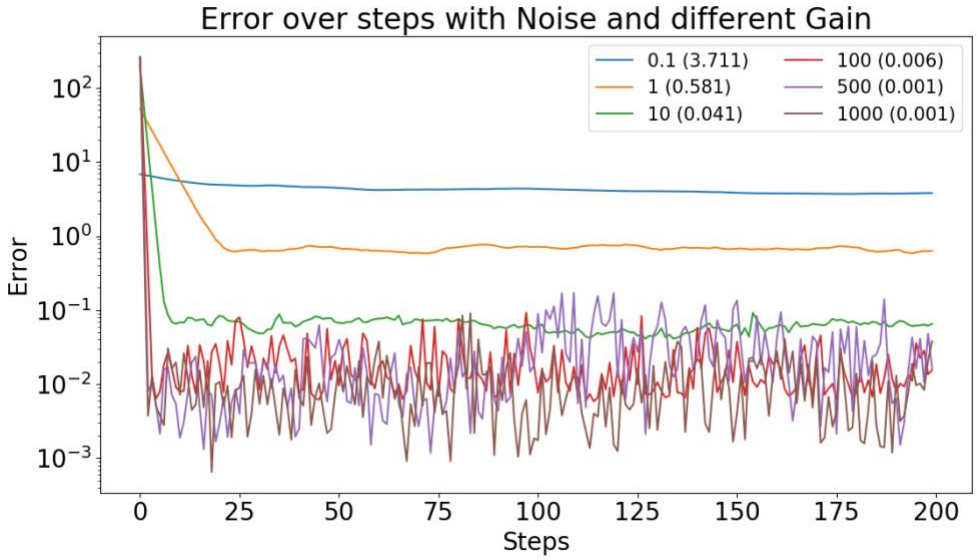


Figure 25. Estimation error with noise

7. Conclusions

In this project, we have created and tested a kinematic control strategy for a KUKA LBR iiwa robot manipulator which can be adopted during minimally invasive surgery operation.
We have evaluated the accuracy of this approach by analyzing a set of numerical results by setting different scenarios in order to better understand if the system properly achieves the assigned tasks.
These results that we have obtained are very good and legitimize the introduction of a robot which can be kinematically controlled into a surgical environment in order to appreciate its benefits. Moreover, this part of the project allowed to create a solid base to implement successive important phases whose the aim is to improve the work of a robot manipulator in a Minimally Invasive Surgery context.

Once that we are able to manipulate kinematically the robot, we have to focus on how to manage the contact with the patient in order to avoid harming the patient's body during surgical operations.
We introduced the residual method as a strategy to estimate the external forces which affects the robot during the operation with the aim to reproduce these forces such that, in a future phase of the project, the robot will be programmed in order to react to these forces in a proper way.
Indeed, this is the first step to implement a good reaction strategy which allows the robot to work avoiding undesired effects on the patient.

After that we have checked the correct functioning of the residual method to estimate external forces which acts on the robot end effector, in order to simulate a more realistic scenario we have considered the dynamic features which can derive from a lying patient: in particular we have considered breathing and in heart beating. We approximated these events and, indeed, they have been modeled as external sinusoidal forces acting on the robot tool.

The experimental results have shown the accuracy on how to recognize and reconstruct these external forces using residual method. Since we decided to introduce some more realistic feature in our simulation environment, we have tested the system assuming multiple scenarios: indeed, we set a white gaussian noise to the control torque measure of the robot in order to simulate a real robot proprioperceptive sensor measurement and, moreover, we tested this method approximating the frequency of the external force to a realistic heart rate.

In future works, the main idea is to implement a dynamic control scheme that reproduces a motion which can adapt and react to the external forces application in order to avoid negative consequences. Surely, the introduction of dynamic controlled features allows the problem to jump forward to future operation performed by both robot and human: this approach will bring many benefits both to surgeon and patient.

Firstly, this system reduces the human component, reducing the percentage of error due by human mistakes. The minimally invasive surgery procedure is allowed for a growing number of typologies of surgical operation and so it will bring benefits to a greater number of patients. Secondly, a faster and easier way to operate a person will bring private and public medical care center to operate a huger number of people respect to the nowadays statistics. The introduction of systems like the one proposed in this work will help mostly public hospital center in countries like Italy with public health care and a saturated health system, to increase the number of allowed operations per day without reducing the accuracy of these operations.

The main negative aspect of this approach is the high initial cost for setting the whole structure and training the medical staff.

The introduction of this kind of technology in operating room is taking hold more in recent years, (mostly in urology operations in Italy and Europe and gynecological operations in USA) confirming the growing need of the man to be supported during increasingly difficult operations by the robots, leading the way for resolution of problems once considered impossible.

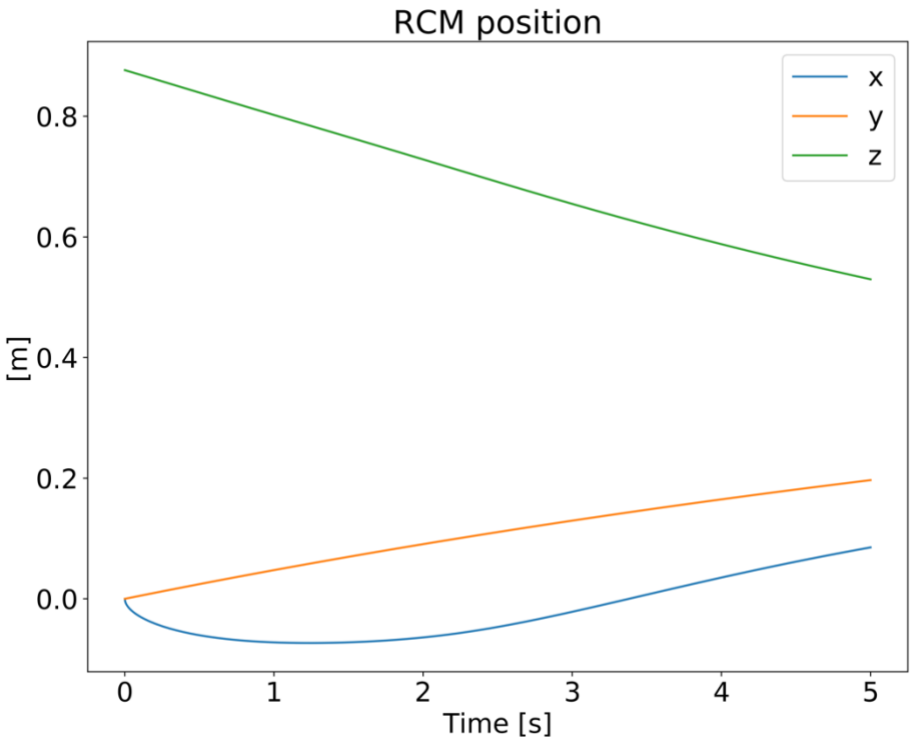
Appendix

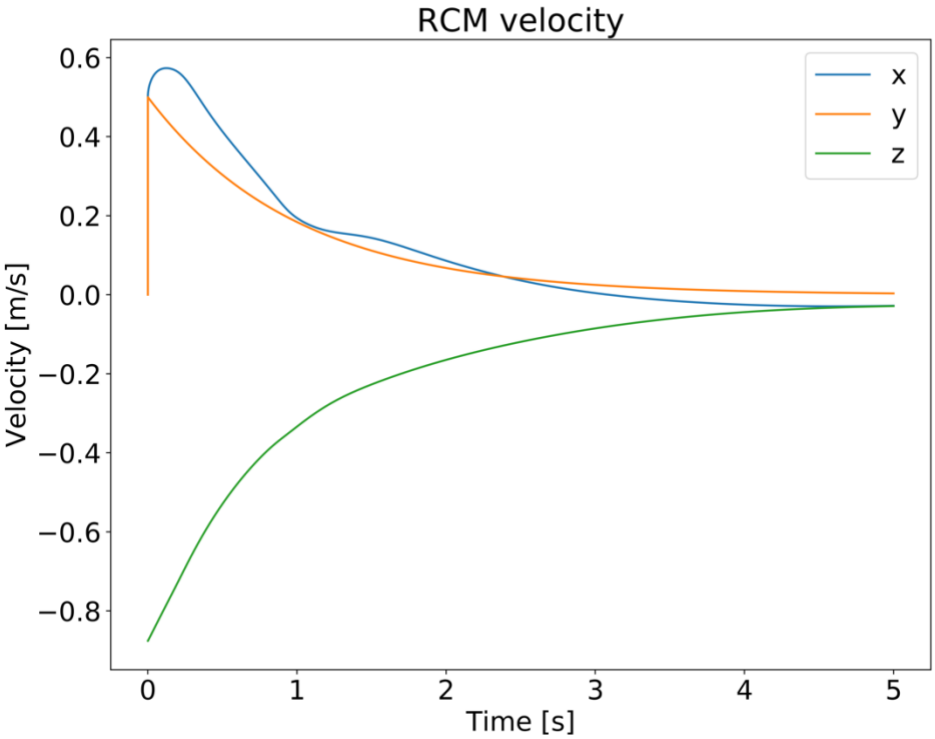
This appendix reports several plots which were elaborated during the simulations. The A part shows the joints positions and velocities, and the RCM position and velocity during the kinematic control system simulations. The B part contains the control torques that the robot assumes during the application of an external force. Moreover, we show the joints positions, velocities and accelerations in three different plots which resembles approximately all the cases.

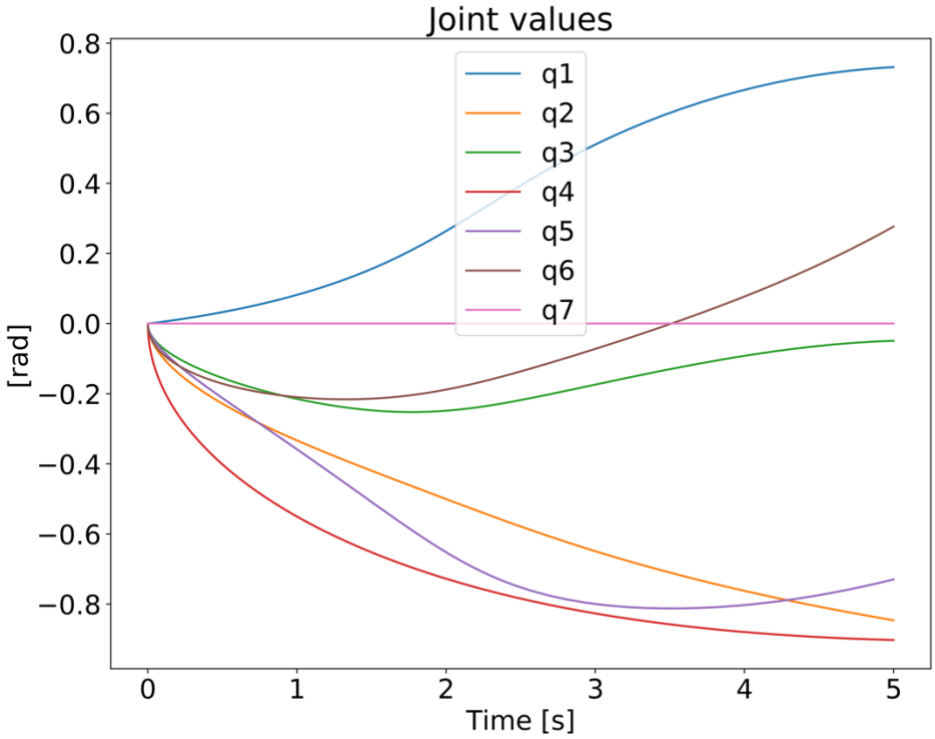
A Appendix

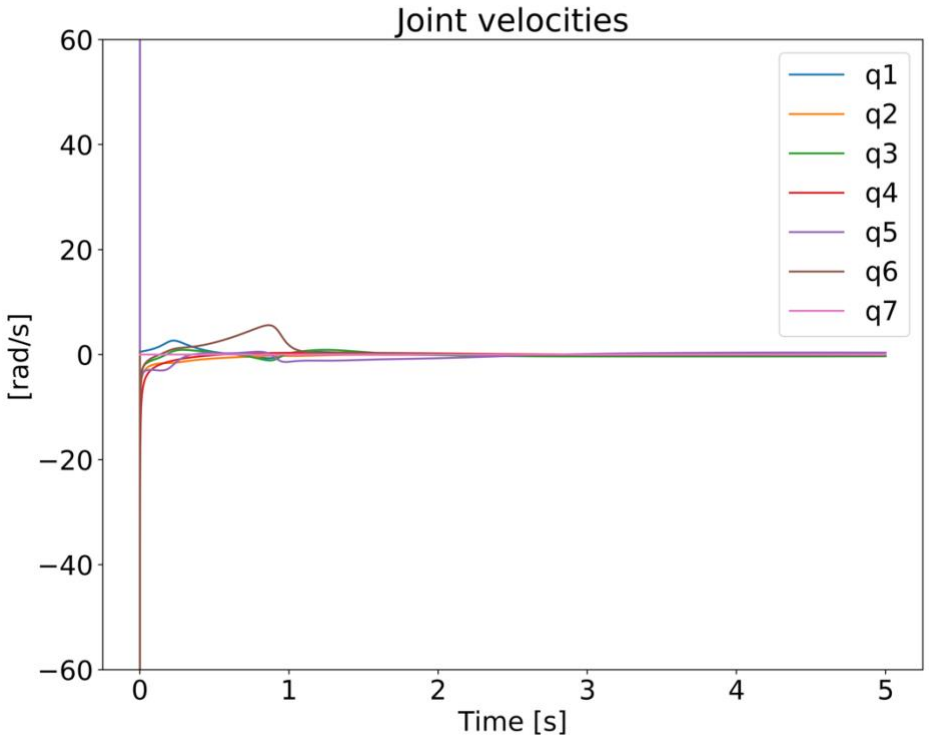
The overview of these results follow the same order which was chosen during the presentation of the experimental results in section 6.1:

- K = 0.1

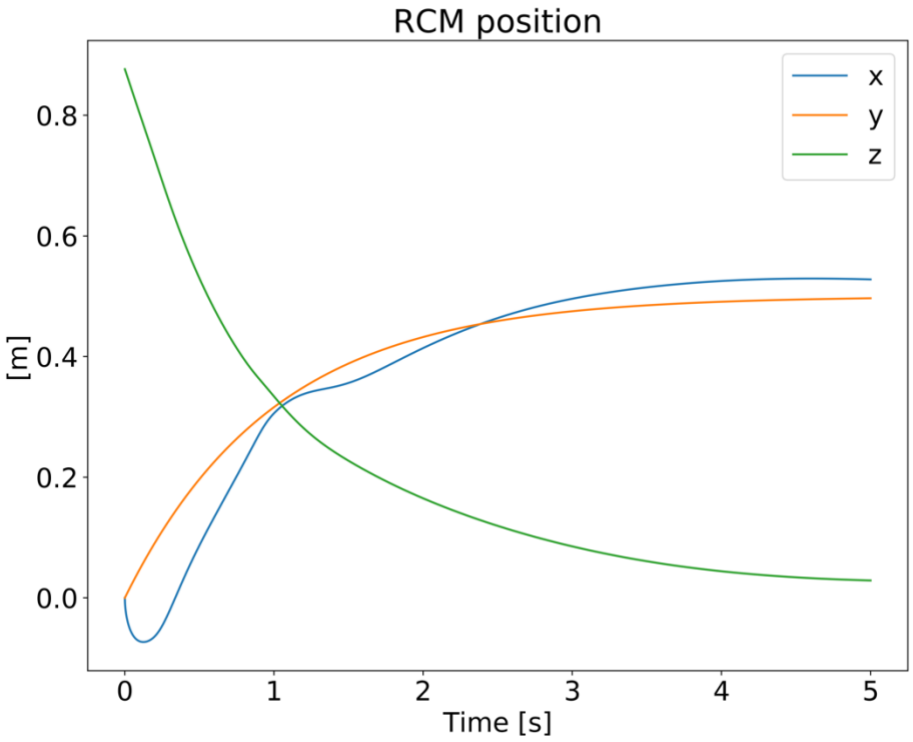


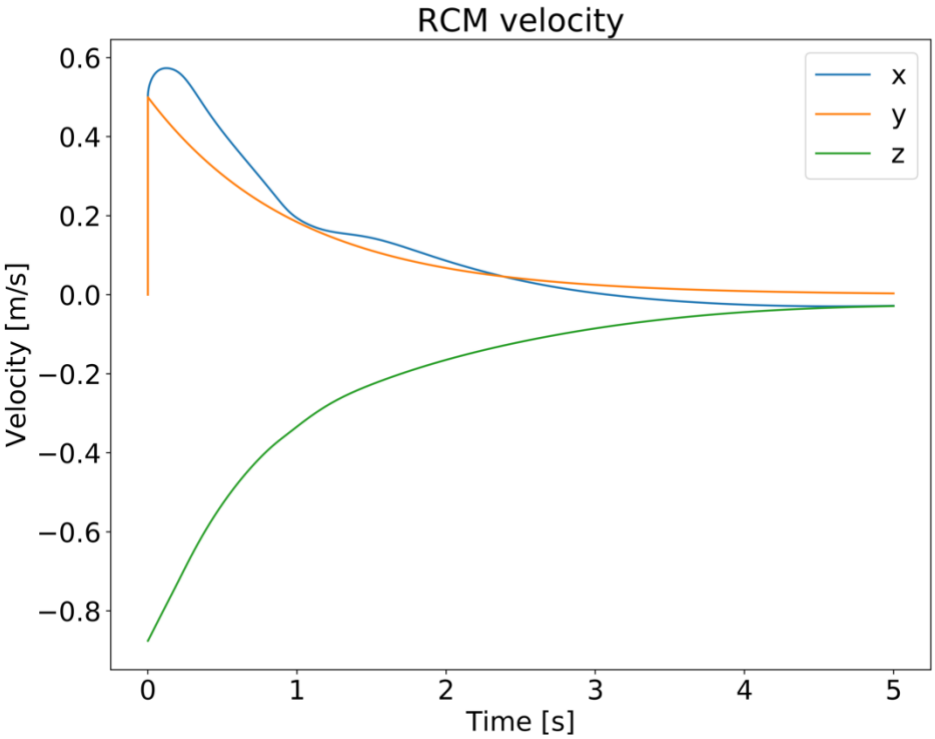

RCM velocity
x
y
z
Velocity [m/s]
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
1
2
3
4
5
Time [s]

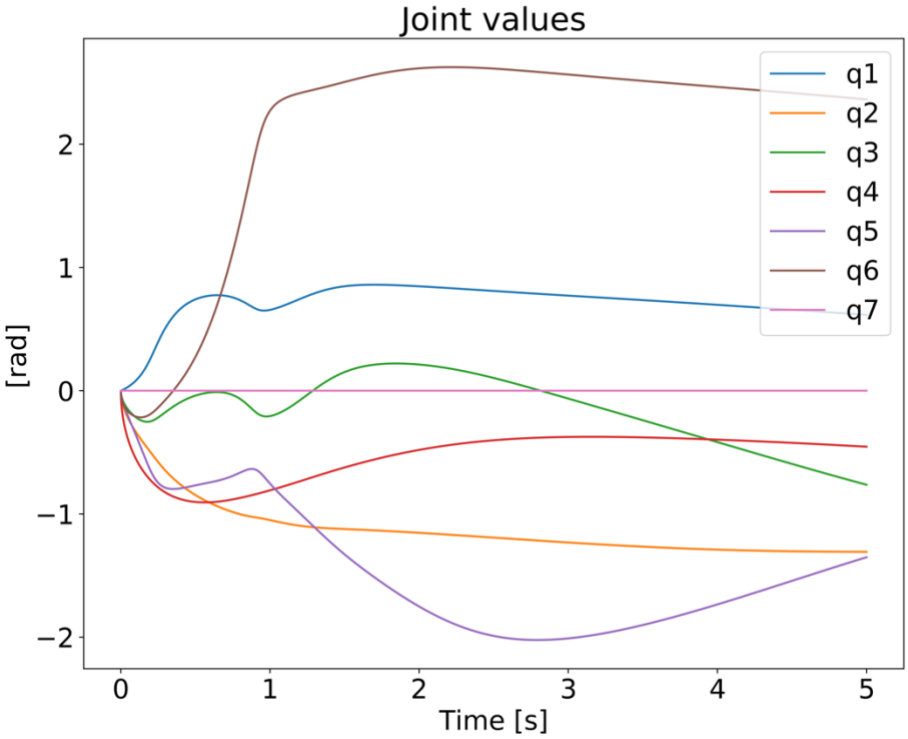

Joint values
q1
q2
q3
q4
q5
q6
q7
[rad]
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
1
2
3
4
5
Time [s]

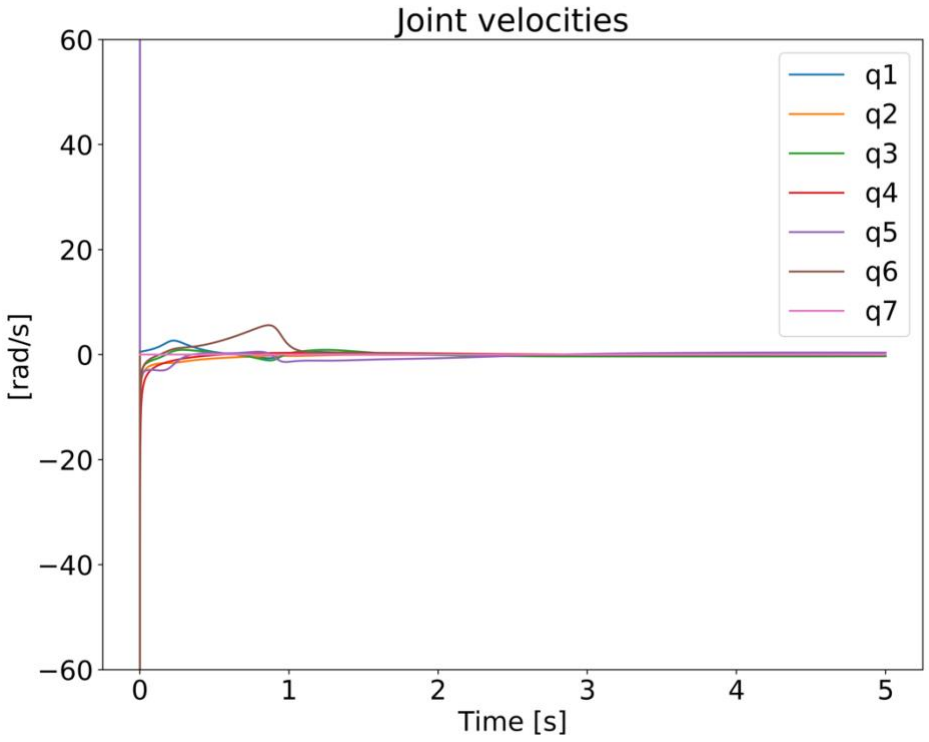

Joint velocities
60
40
20
0
−20
−40
−60
[rad/s]
0
1
2
3
4
5
Time [s]
q1
q2
q3
q4
q5
q6
q7

- K = 1.0

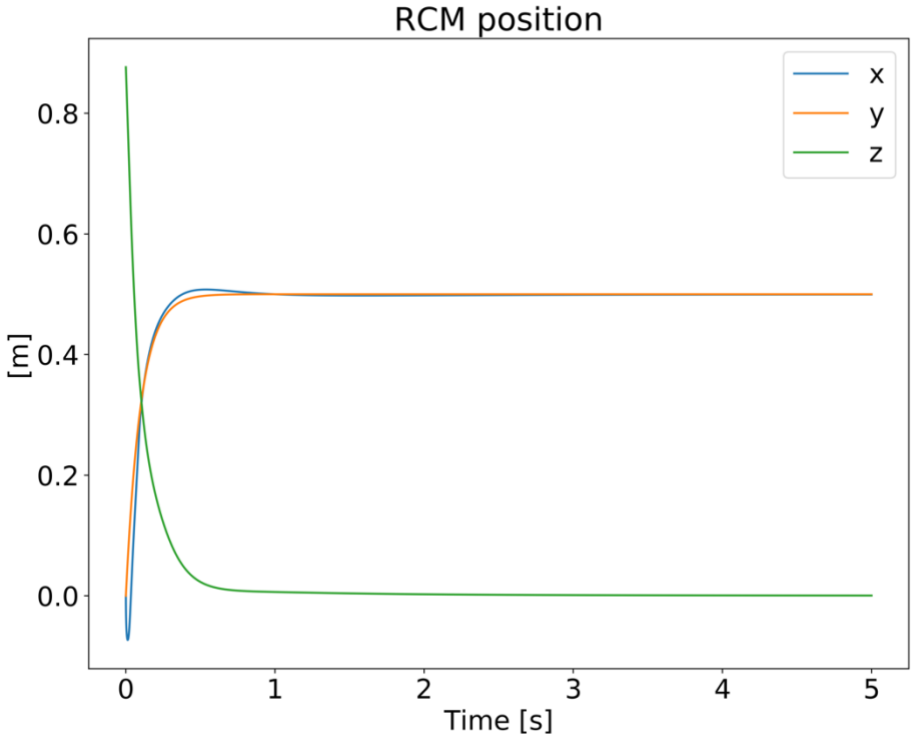

RCM position
0.8
0.6
0.4
0.2
0.0
[m]
0
1
2
3
4
5
Time [s]
x
y
z

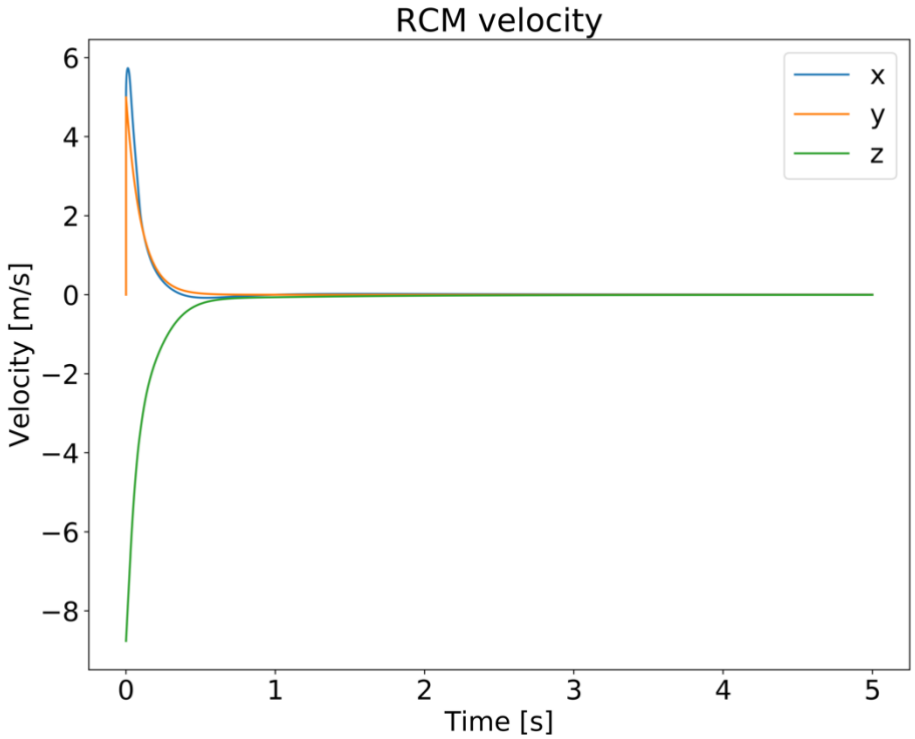

RCM velocity
x
y
z
Velocity [m/s]
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
1
2
3
4
5
Time [s]

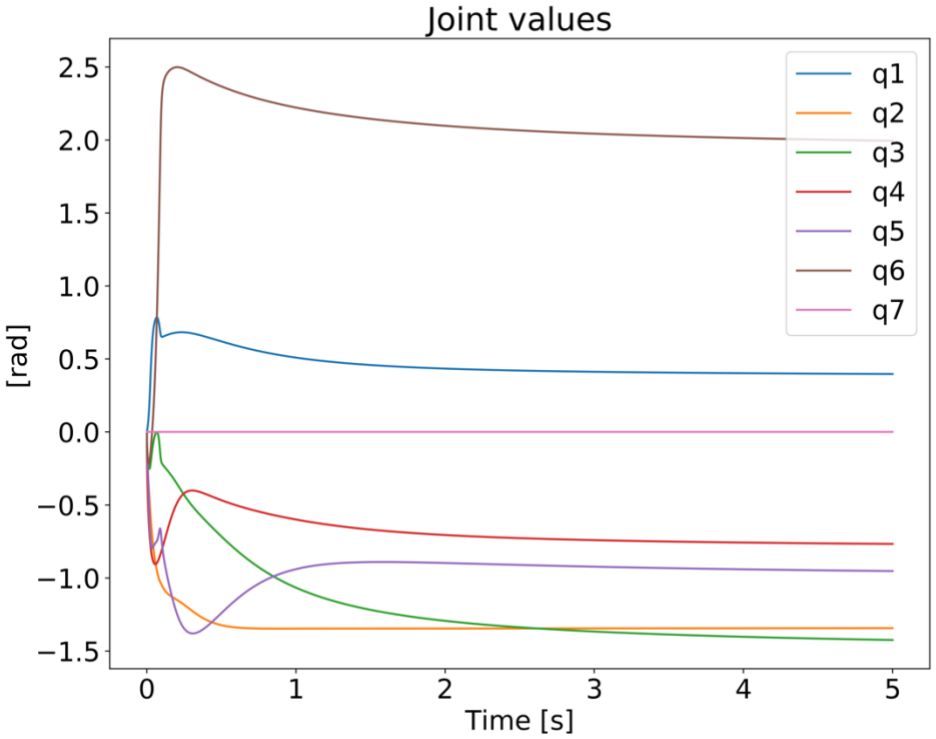

Joint values
q1
q2
q3
q4
q5
q6
q7
[rad]
2
1
0
−1
−2
0
1
2
3
4
5
Time [s]

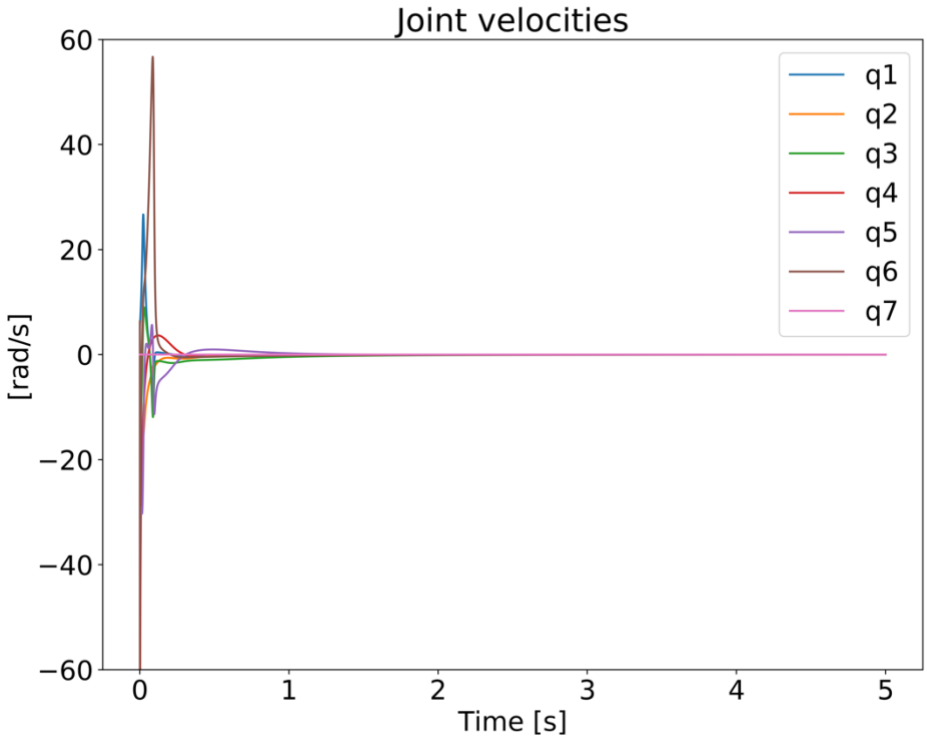

Joint velocities
60
40
20
0
−20
−40
−60
[rad/s]
0
1
2
3
4
5
Time [s]
q1
q2
q3
q4
q5
q6
q7

- K = 10.0:

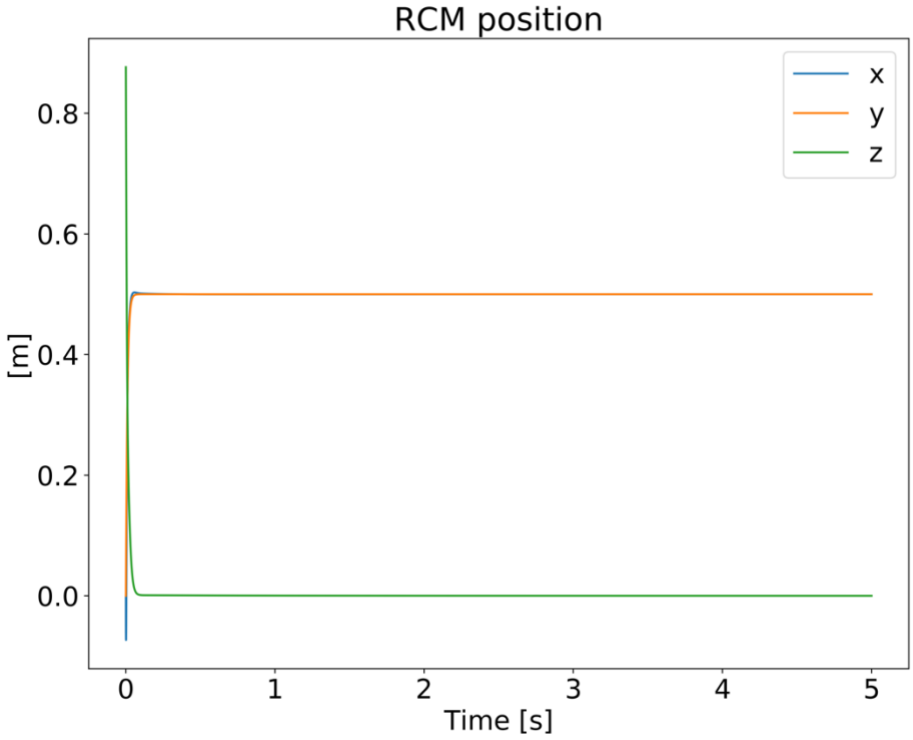

RCM position
0.8
0.6
0.4
0.2
0.0
[m]
0
1
2
3
4
5
Time [s]
x
y
z

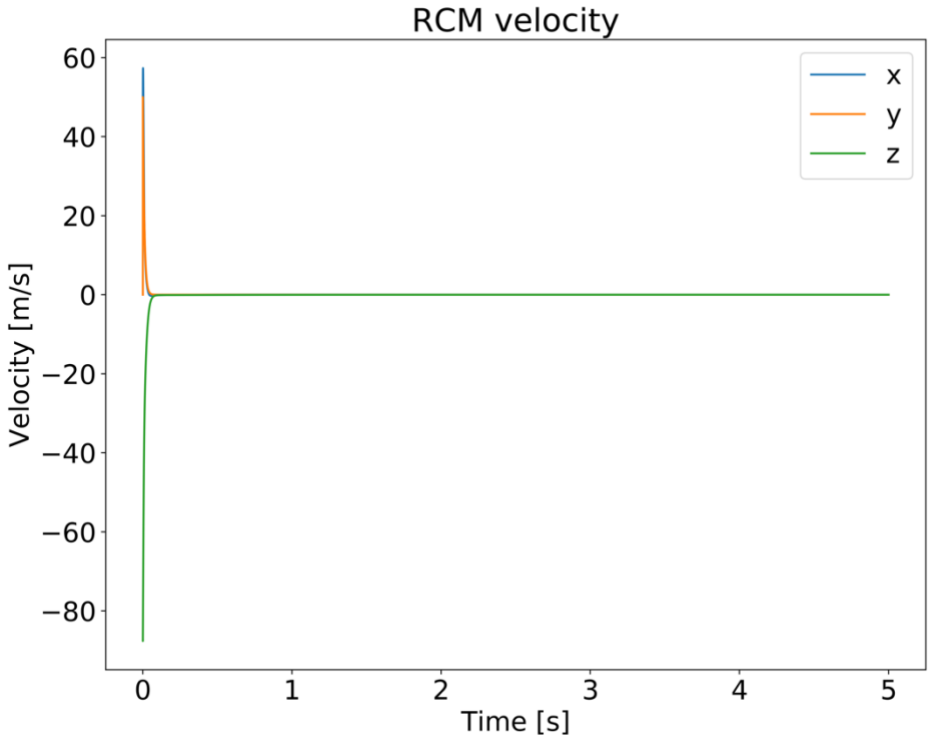

RCM velocity
x
y
z
Velocity [m/s]
6
4
2
0
−2
−4
−6
−8
0
1
2
3
4
5
Time [s]

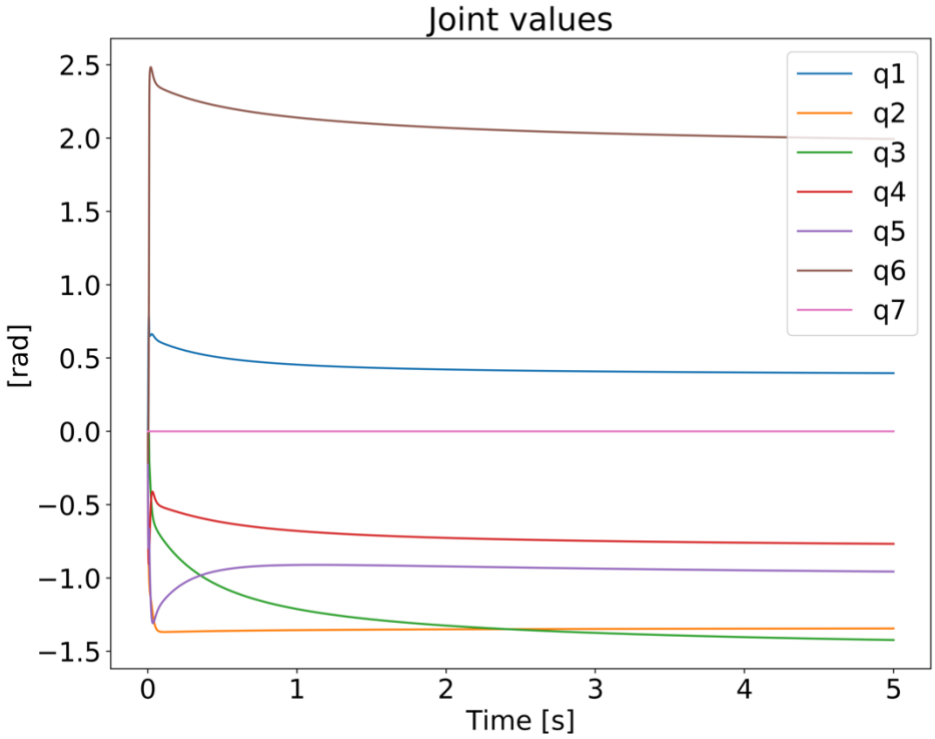

Joint values
q1
q2
q3
q4
q5
q6
q7
[rad]
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
1
2
3
4
5
Time [s]

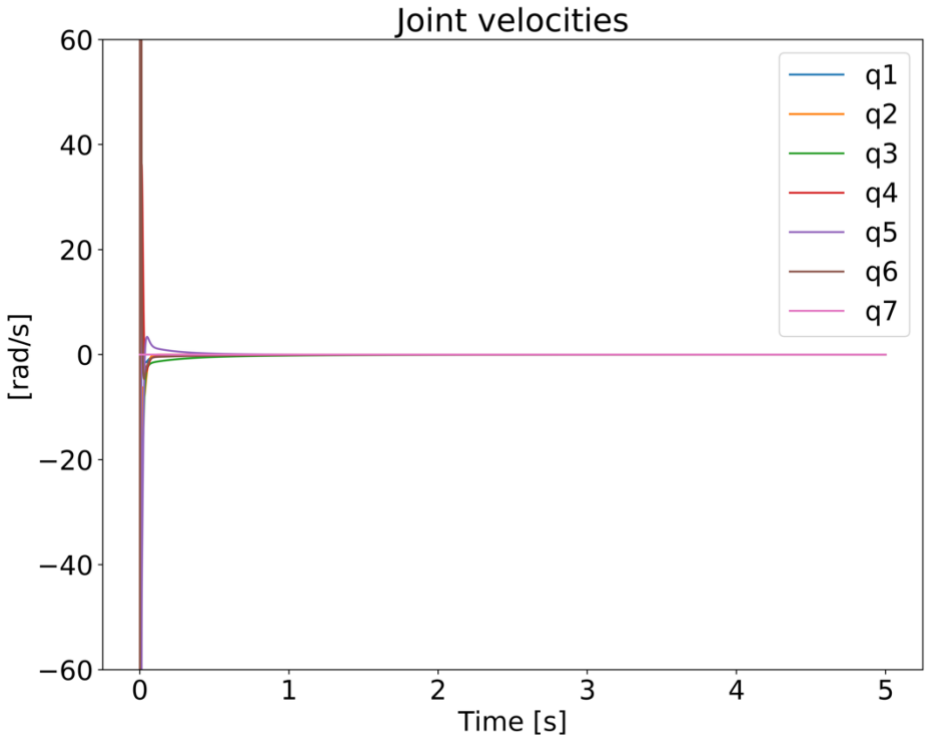

Joint velocities
60
40
20
0
−20
−40
−60
[rad/s]
0
1
2
3
4
5
Time [s]
q1
q2
q3
q4
q5
q6
q7

- K = 100.0:


RCM position
0.8
0.6
0.4
0.2
0.0
[m]
0
1
2
3
4
5
Time [s]
x
y
z

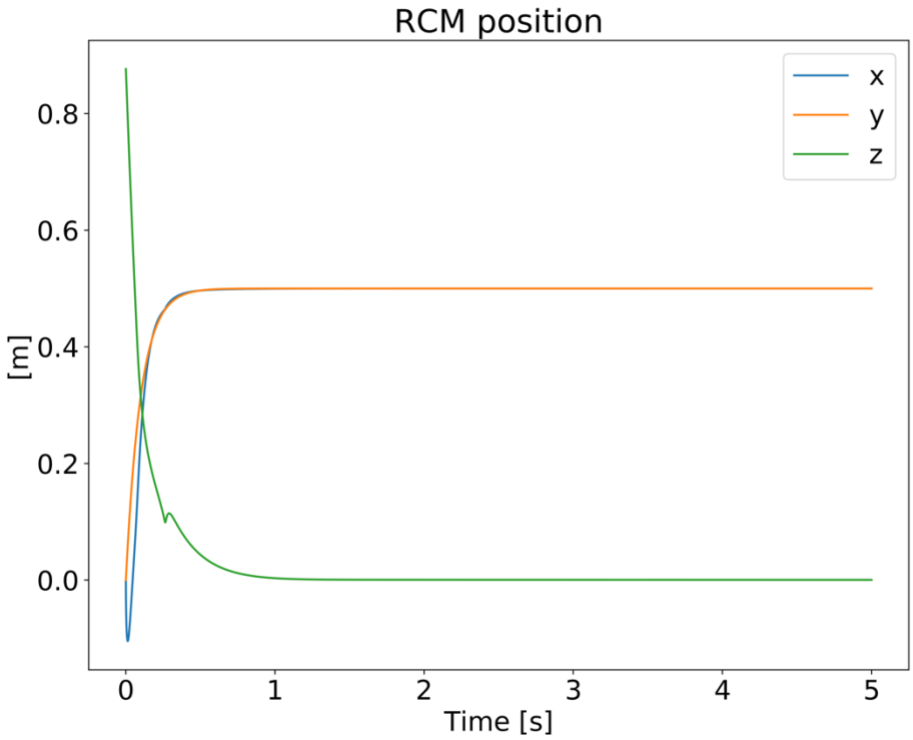

RCM velocity
x
y
z
Velocity [m/s]
60
40
20
0
−20
−40
−60
−80
0
1
2
3
4
5
Time [s]

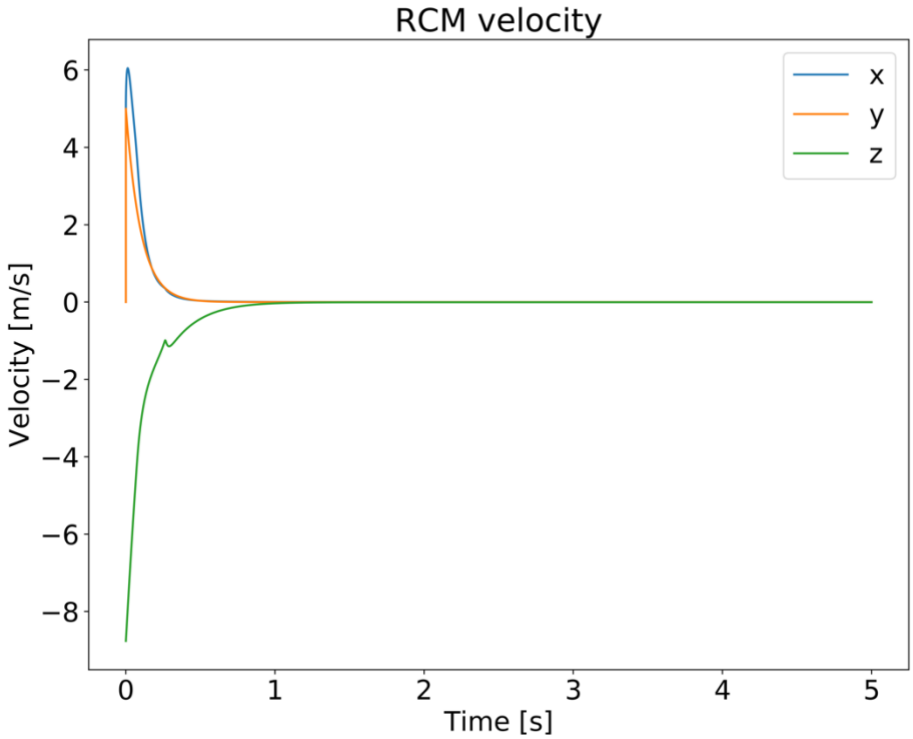

Joint values
q1
q2
q3
q4
q5
q6
q7
[rad]
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
1
2
3
4
5
Time [s]

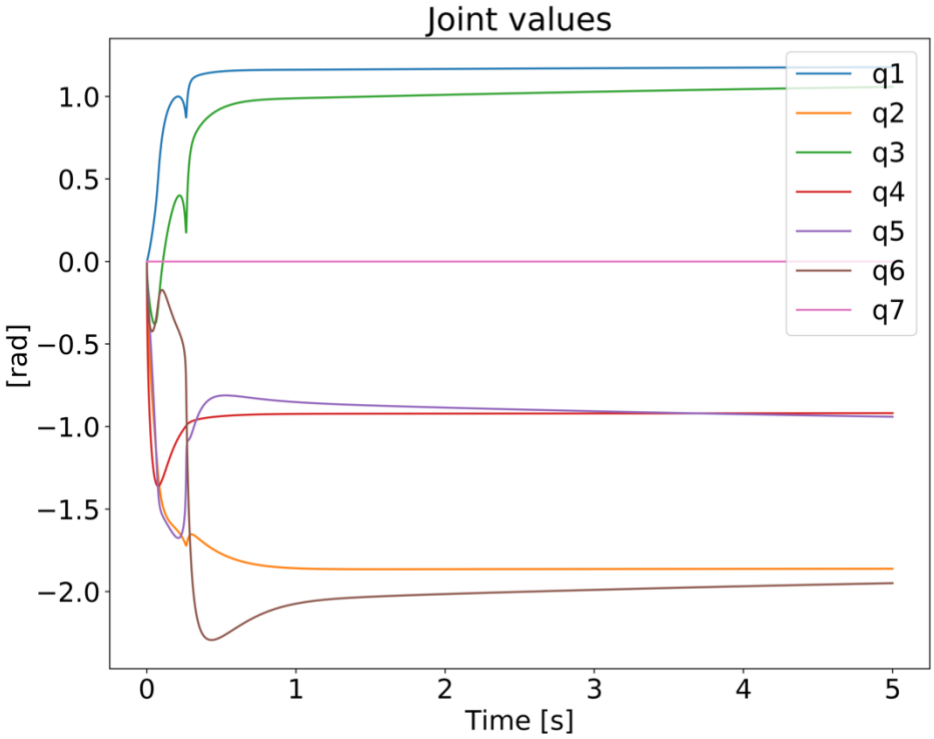


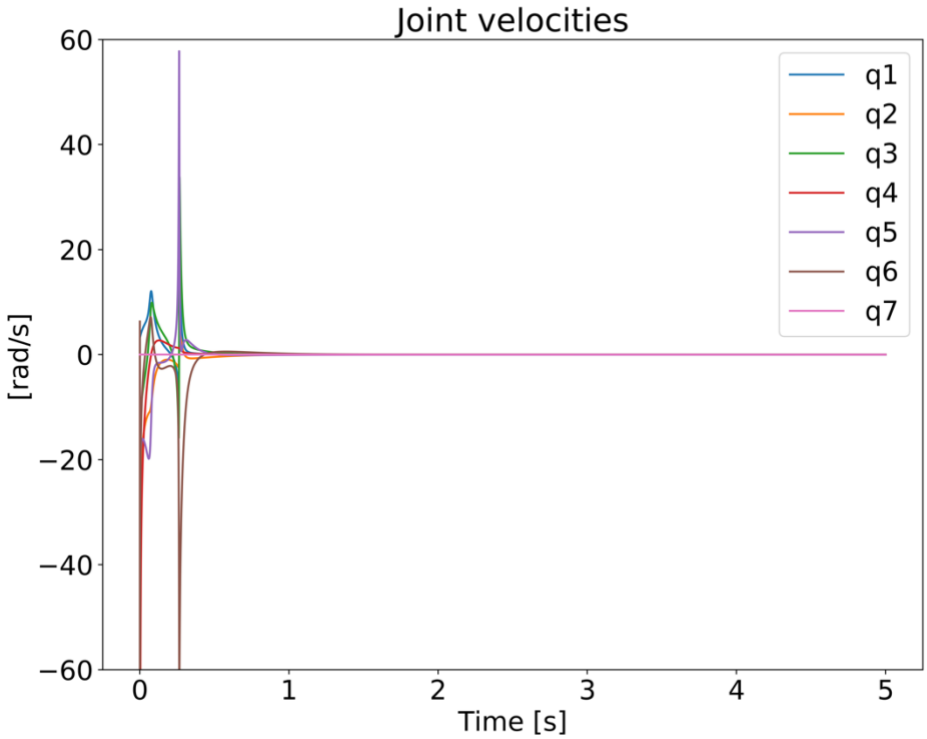
Now, we see the results obtained when we use different additional task, exactly as we did in the experiments sections:

- Deep Insertion Task:

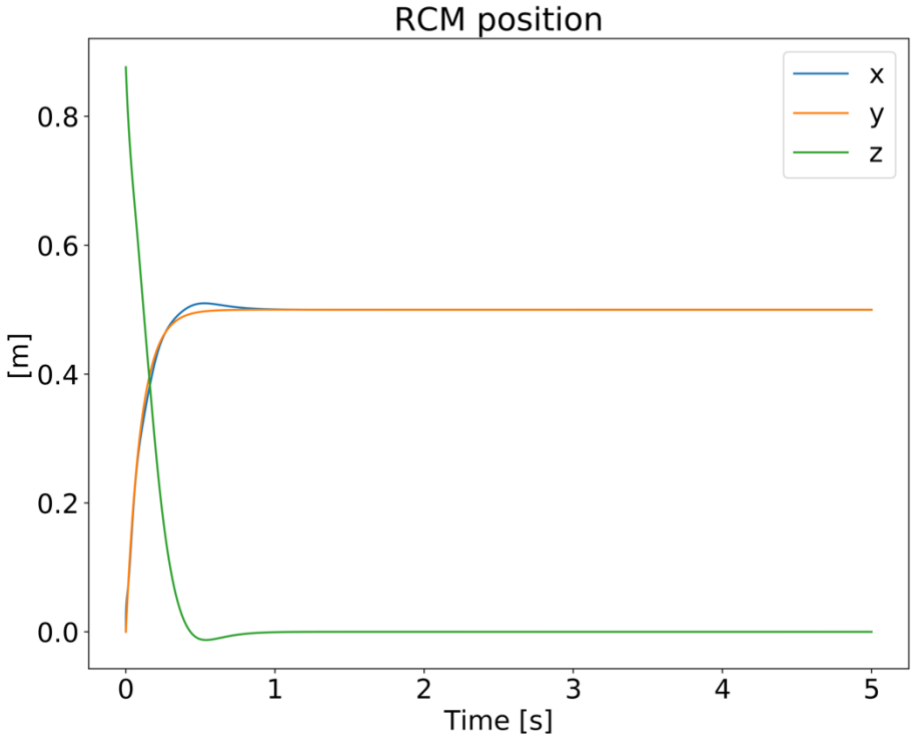


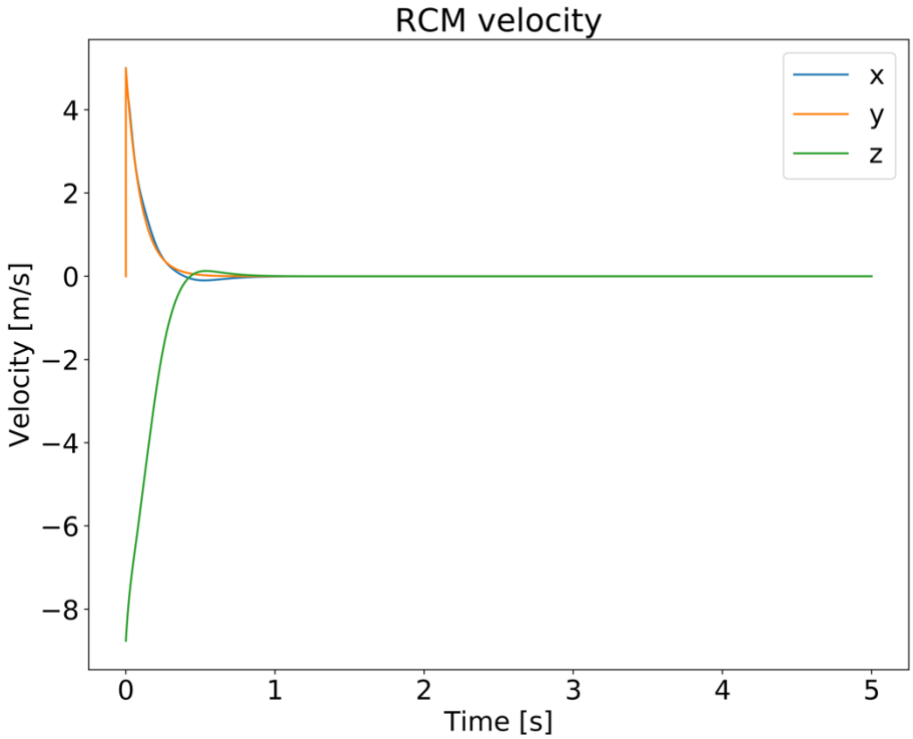

RCM velocity
x
y
z
Velocity [m/s]
6
4
2
0
−2
−4
−6
−8
0
1
2
3
4
5
Time [s]

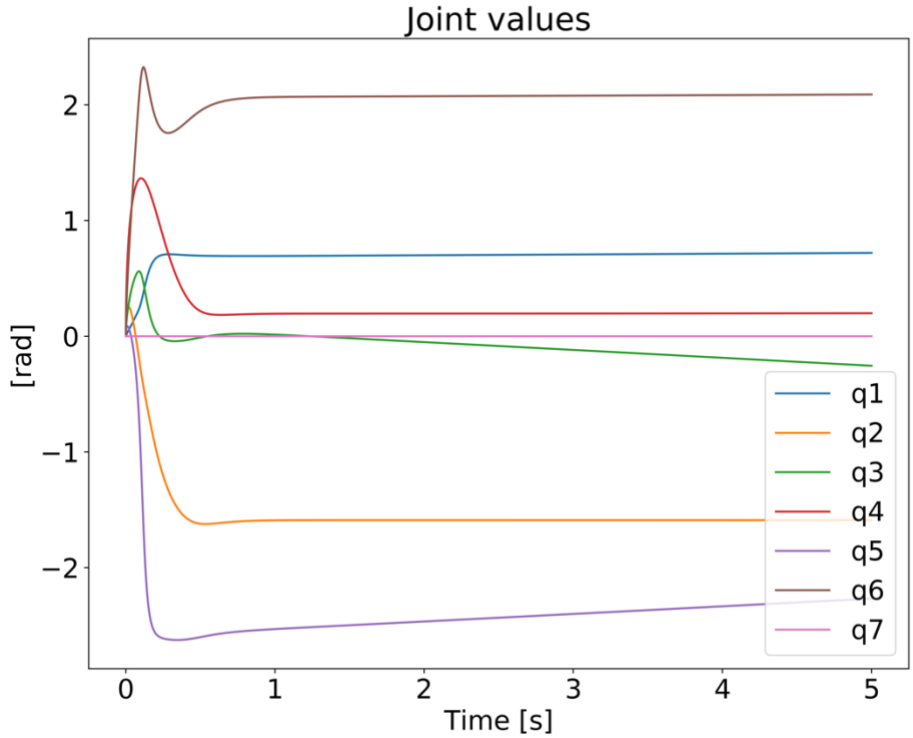

Joint values
q1
q2
q3
q4
q5
q6
q7
[rad]
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
1
2
3
4
5
Time [s]

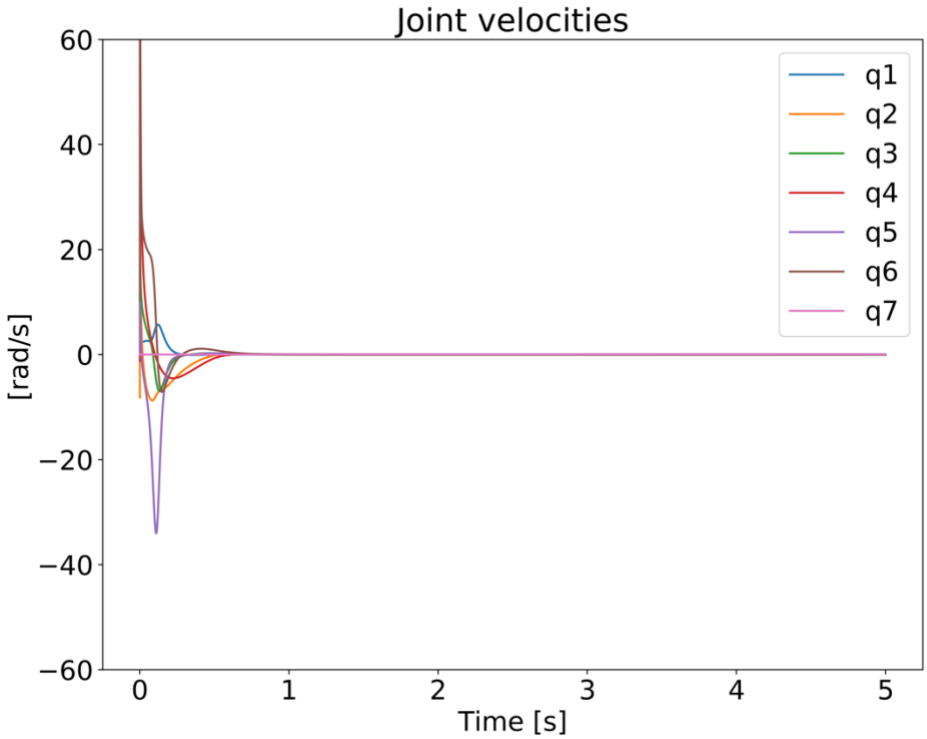


- Diagonal Insertion Task:




RCM velocity
x
y
z
Velocity [m/s]
4
2
0
−2
−4
−6
−8
0
1
2
3
4
5
Time [s]


Joint values
[rad]
2
1
0
−1
−2
q1
q2
q3
q4
q5
q6
q7
0
1
2
3
4
5
Time [s]



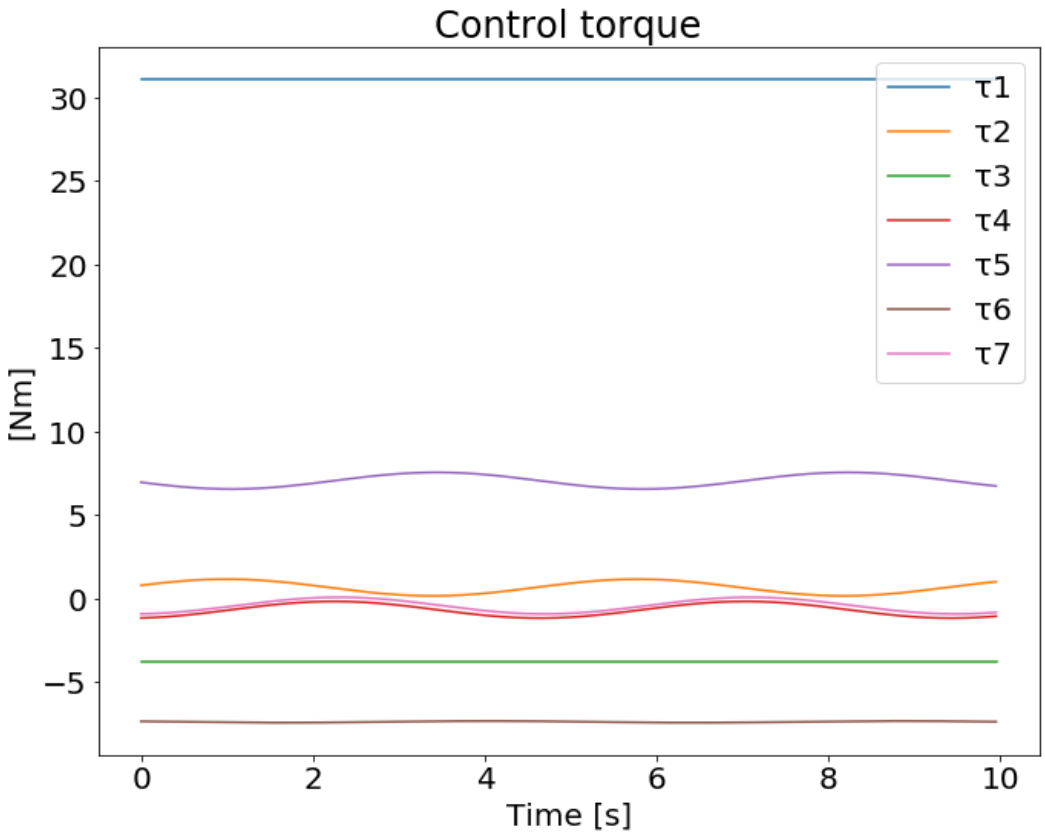
B Appendix

The following plots represents the control torques produced by the robot's actuators: these signals compensates two main components, and these are the gravity term and the the external forces effects on robot joints. This way, the robot is able to stay still during the application of the external forces.
In order to maintain the robot still we apply these torques and we look what happens changing the intensity of external forces: as expected, the torques follow a periodic behavior due to the periodic nature of external forces. In these plots we can see that the more you increase the intensity of the external forces and the more it increases also the amplitude of these signals.

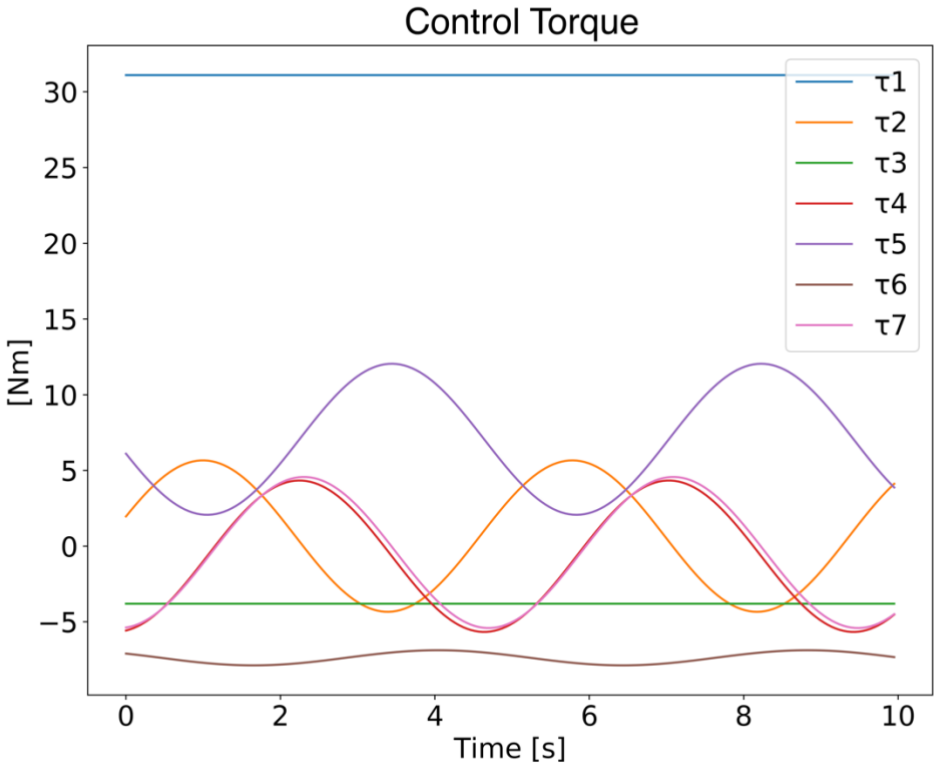
The joints which are closer to the base undergo a smaller effect of the external forces applied to the RCM point on the end-effector and, indeed, in the plots we can note that these joints are affected only by the constant compensation of the gravity term.
Naturally, If you increase the intensity of the external forces, then the effect will be also more evident on the joints closer to the base of the robot, and it will be highlighted by the periodic compensation.

Here is the overview of these results:

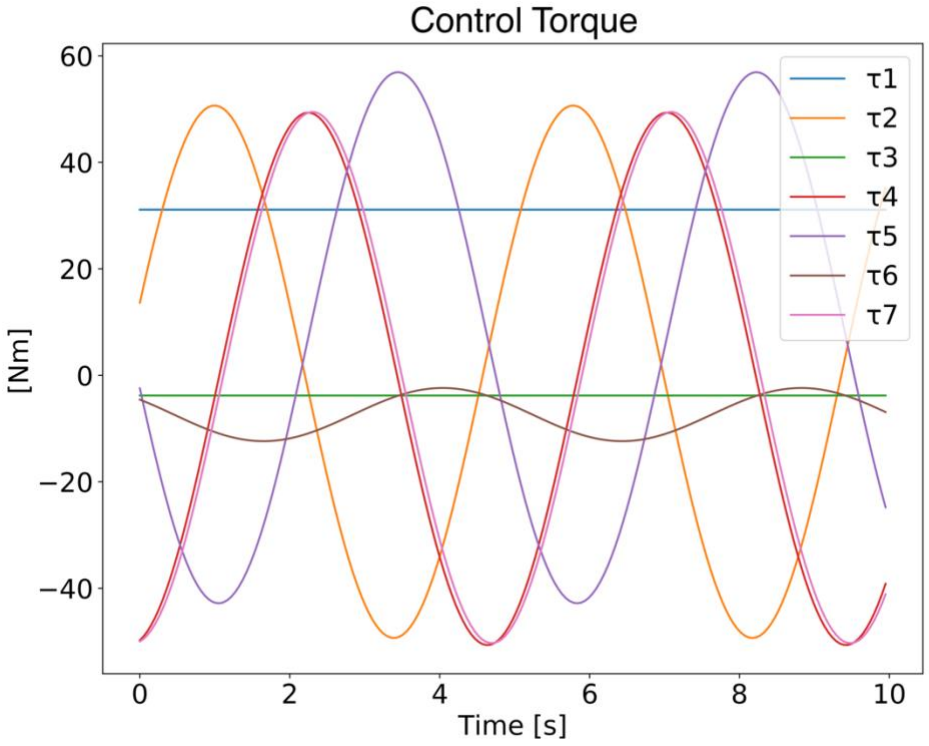
- $F_{ext} = [0.5\ N * F_x, 0.5\ N * F_y, 0, 0, 0]$



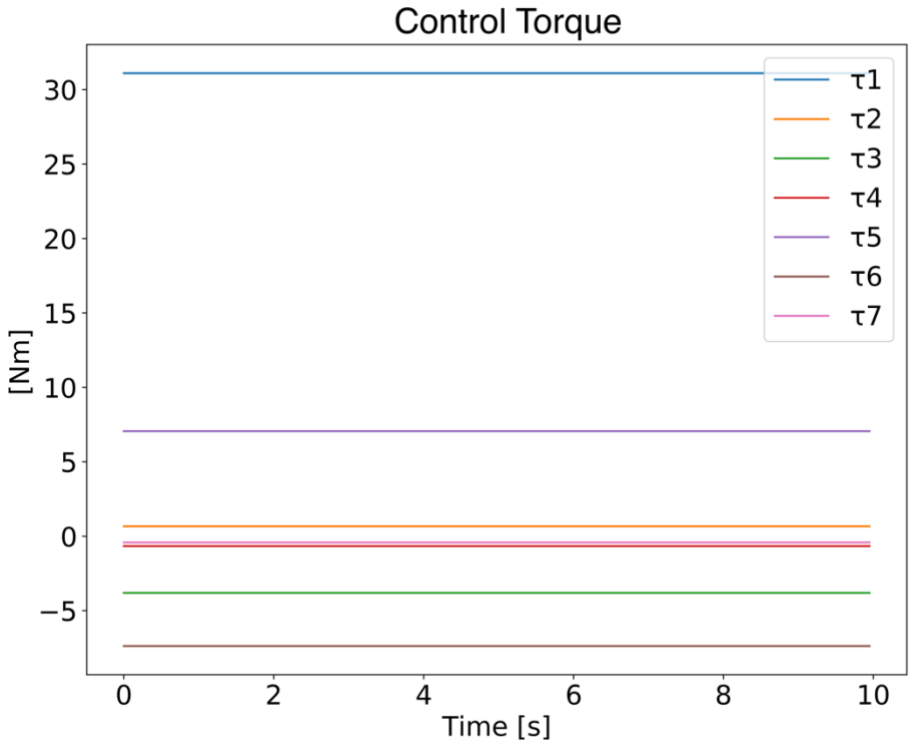
- $F_{ext} = [5\ N * F_x, 5\ N * F_y, 0, 0, 0]$



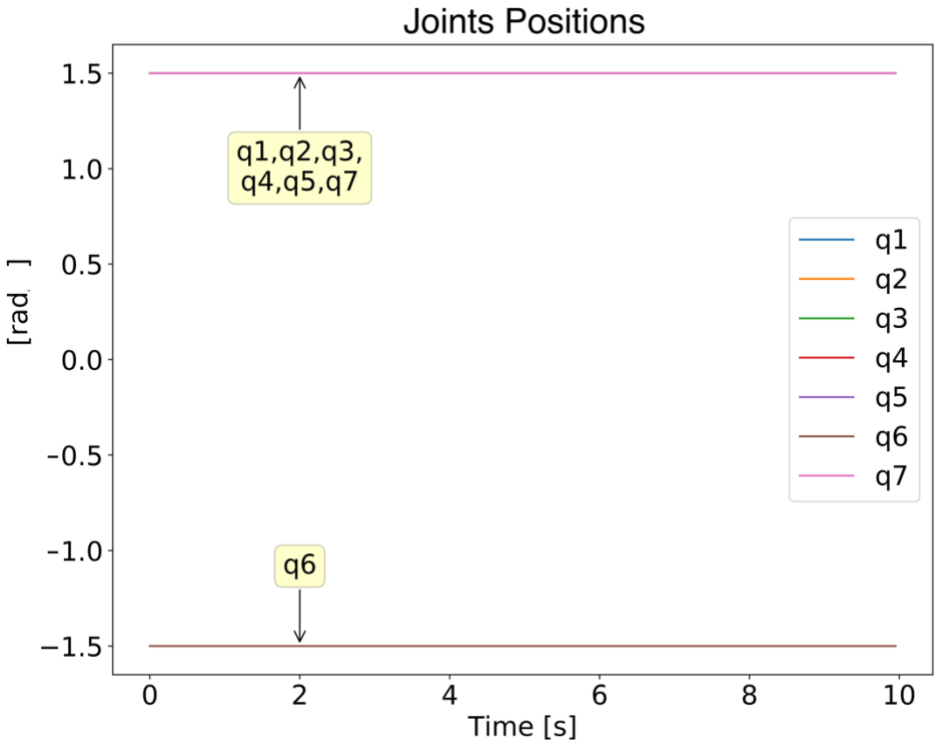
- $F_{ext} = [50\ N * F_x, 50\ N * F_y, 0, 0, 0]$

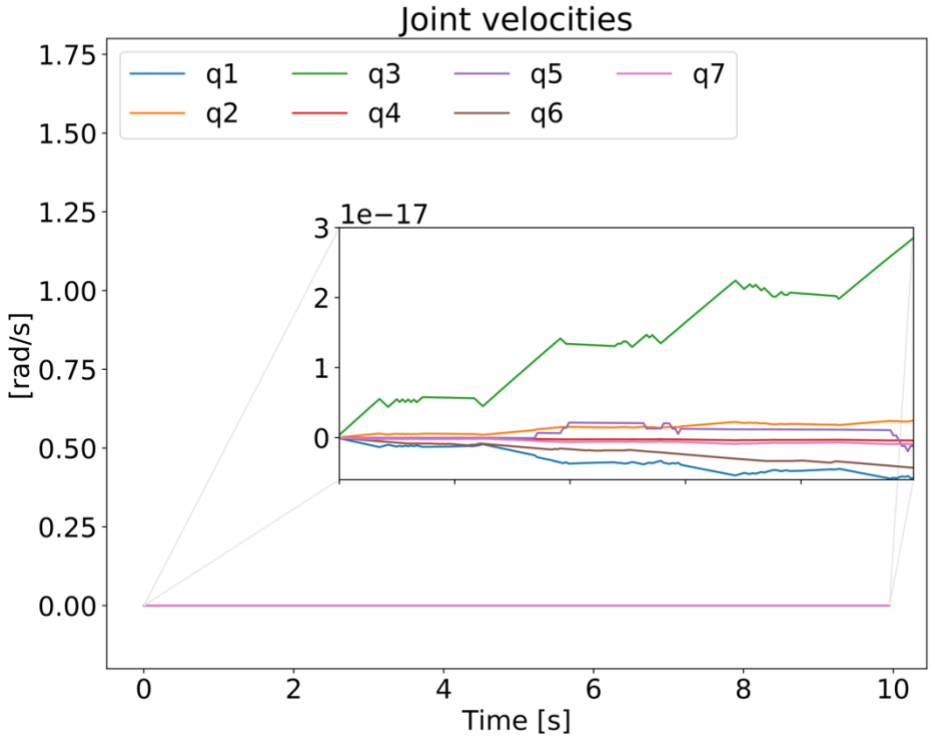


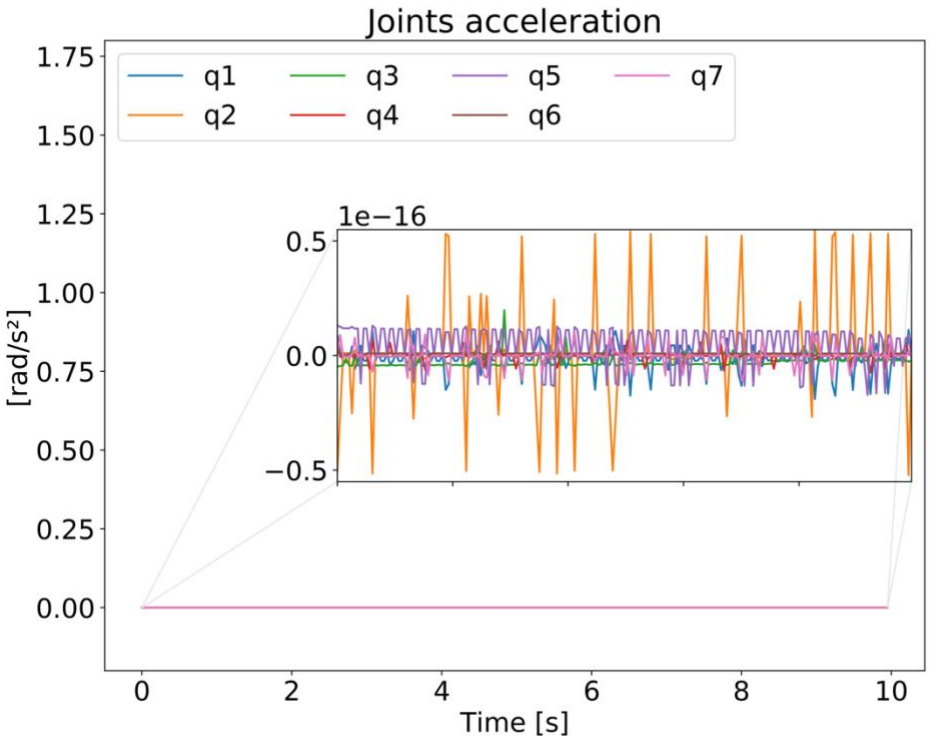
- $F_{ext} = 0\ (Gravity\ Compensation)$



Considering the following results valid for all the simulations, we show the joints positions, velocities and accelerations:






Joints acceleration
q1
q2
q3
q4
q5
q6
q7
1e−16
0.5
0.0
−0.5
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
[rad/s²]
0
2
4
6
8
10
Time [s]

References

[1] N. Aghakhani, M. Geravand, N. Shahriari, M. Vendittelli, G. Oriolo, “Task Control with Remote Center of Motion Constraint for Minimally Invasive Robotic Surgery,” 2013 IEEE International Conference on Robotics and Automation, 2013

[2] Cong D. Pham, Fernando Coutinho, Antonio C. Leite, Fernando Lizarralde, Pal J. From, Rolf Johansson, “Analysis of a Moving Remote Center of Motion for Robotics-Assisted Minimally Invasive Surgery,” IROS 2015, pp. 1440–1446, 2015

[3] S.Haddadin, A.De Luca, and A.Albu-Schaffer,”Robot Collisions: A Survey on Detection,Isolation, and Identification”, IEEE transaction robotics, vol. 33, no. 6, december 2017

[4] . De Luca and R. Mattone, “Actuator fault detection and isolation using generalized momenta,” inProc. IEEE Int. Conf. Robot. Autom., 2003,pp. 634–639.

[5] . https://healthproadvice.com/procedures/Effects-of-Anesthesia-on-the-Heart

[6] . C.Gaz, F.Flacco, A. De Luca “Extracting Feasible Robot Parameters from Dynamic Coefficients using Non Linear Optimization Methods”.