POLITECNICO
MILANO 1863

UNIVERSITY
OF ŽILINA

# Analysis and Modelling of Operational Characteristics of Delta Robot FANUC M-1*i*A using MATLAB/Simulink

Thesis for Master of Science Degree in Mechanical Engineering

Author: **Michele Viganò**

Student ID: 971735

Advisor: prof. Vladimír **Bulej**, prof. Hermes **Giberti**

Academic Year: 2021-22

School of Industrial and Information Engineering

# Abstract

This master thesis deals with the simulation of the six degrees of freedom Delta robot Fanuc M-1$i$A/0.5A. A multi-body model was built in *Mathworks Matlab Simulink* from a CAD drawing using *Simscape Multibody* blocks. The robot wrist is considered at constant orientation, thus the model has only three degrees of freedom. The model was used to detect the robot workspace, to solve the kinematics problems and to analyse the motor torques and the forces in the structure. The work contributed with relevant knowledge about the M-1$i$A/0.5A and produced a software framework easily adaptable for further analysis or to other robot models.

**Keywords:** Fanuc M-1iA, Delta robot, Simulink model, operational characteristics

# Abstract in italiano

Questa tesi di laurea magistrale concerne la simulazione del Delta robot Fanuc M-1*i*A/0.5A a sei gradi di libertà. *Mathworks Matlab Simulink* è stato usato per costruire un modello multibody con blocchi *Simscape Multibody* a partire da un disegno CAD del robot. Il polso robotico è considerato rigido, dunque il modello ha solo tre gradi di libertà. Il modello è usato per rilevare lo spazio di lavoro, risolvere i problemi cinematici, e analizzare la coppia fornita dai motori e le forze nella struttura. Il lavoro ha contribuito con conoscenze importanti sul M-1*i*A/0.5A e ha prodotto un complesso di software facilmente adattabile a ulteriori analisi o applicabile ad altri modelli di robot.

**Parole chiave:** Fanuc M-1iA, Delta robot, modello Simulink, caratteristiche operative

# Contents

# Introduction

Robots are very commonly used in production systems, and they have different structures depending on their designed purpose. The Delta robots are a class of parallel kinematics robot capable of high speed motion and exceptional accelerations. Their main limitations are the small workspace and the limited payload. For these reasons they are extensively adopted in assembly lines for pick and place, packaging and labelling of lightweight objects. Thanks to high stiffness and precision they recently found use in 3D printing, haptics and surgery.

The robot manufacturers provide some information about the characteristic of their products, and enable the users to control their robots. In this context, computer simulations are crucial, and it is hard to design or implement a robot in a production system without first simulating it.

The goal of this work is to delve into the Fanuc M-1*i*A/0.5A and discover its operational characteristics. This process is carried out in two steps. Firstly a digital model of the robot must be created. Then some tests are designed to experiment the hypothetical traits and limits. Ideally, the information acquired should be validated by reproducing it on a physical model. Unfortunately this third step was not undergone in this work.

This thesis was developed during an *Erasmus+* exchange period in the University of Žilina in Slovakia, at the Faculty of Mechanical Engineering, Department of Automation and Production Systems.

## Structure of the Document

This document is structured in 8 Chapters. The first one is an introduction to robotics. It presents some elements of robotic theory and describes some core aspects later used in the work. The first Chapter has also specific parts focusing on parallel kinematic structures and on Delta robots. The second Chapter introduces the robot model subject of this study: the Fanuc M-1*i*A/0.5A. The next Chapter is dedicated to the *Simulink* and *Simscape Multibody* software. The basics of these software are discussed in a limited extent, just to give the possibility to understand how the *Simscape Multibody* model of the robot works. Chapter 4 describes how the model was created from a 3D drawing, and how it can be controlled in *Matlab* for the purpose of this study. The remainder of the document presents the results of the analysis undertook. The vast majority of the work focuses on a simpler model of the robot, with the end-effector at constant attitude. Firstly, the extent of translational workspace was measured. Next, Chapter 6 reports the results of kinematic analysis, such as the Jacobian, singularities and manipulability. Chapter 7 tackles how forces are transmitted from the end-effector through the structure and eventually to the motors. The last Chapter contains the conclusions and indications for possible future work on this topic. The two appendixes at the end of this document contain some images reporting the results of Chapter 6 and 7.

# 1 | Introduction to Parallel Kinematic Robots and Delta Robots

This Chapter introduces some important elements of robotics. The theory of this Chapter serves as a basis for the latter part of this document, where some characteristics of the M-1$i$A/0.5A are investigated. After presenting core concepts such as kinematics problems, Jacobian and singularities, the Chapter tackles the parallel kinematics mechanisms and the Delta robots in particular.

## 1.1. Definitions and Basic Theory

A robot is a device composed of many rigid bodies called links, connected to each others with joints. A joint connects exactly two links. Figure 1.1 shows some of the most common joints used in the robotic field. The joints can also be used to connect the robot to the ground. A link may feature an end-effector used to accomplish specific tasks. Some joints can be actuated, typically by electric motors, thereby moving the robot. The motors can be mounted on a link or grounded, and might require a transmission mechanism or have a brake to hold the links in position.

The sequence of links is called kinematic chain. If a chain has a closed loop, it is a closed-chain mechanism, otherwise it is an open-chain mechanism (or serial mechanism). The ground can be considered a link closing a loop, therefore a mechanism with at least two grounded links or joints is closed-chain.

### 1.1.1. Degrees of Freedom

The position of all the points of the robot is called configuration or posture, and can be represented by a few independent parameters. The minimum set of such parameters is called degrees of freedom (dof) of the robot. The dof are real numbers in a continuous

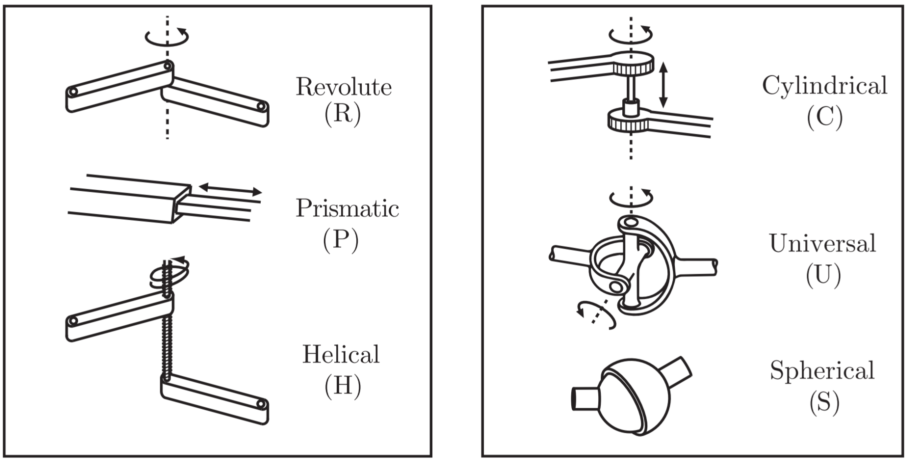


Figure 1.1: Some typical joints. [9, fig. 2.3]

**Number of dof of joints**

| Joint type | Translational dof | Rotational dof | Total dof ($f$) |
|---|---|---|---|
| Revolute (R) | - | 1 | 1 |
| Prismatic (P) | 1 | - | 1 |
| Helical (H) | - | - | 1 |
| Cylindrical (C) | 1 | 1 | 2 |
| Universal (U) | - | 2 | 2 |
| Spherical (S) | - | 3 | 3 |

Table 1.1: Number of dof of joints in Figure 1.1. The number of degrees of constraints is $m - f$, where $m$ is 3 in the plane and 6 in the space. Only the revolute and prismatic joints can be used also in the plane.

range. A rigid body has three dof in a plane (two coordinates to specify the position, and one for the rotation), and six in the space (three for position and three for rotation). When two bodies are connected by a joint, their number of dof is reduced. This is because joints constraint some motions, therefore the description of the configuration requires less parameters. The sum of the number of degrees of freedom and degrees of constraints of a joint is always six (or three for rigid bodies in a plane). The number of degrees of freedom for joints in Figure 1.1 is reported in Table 1.1.

The number of dof of a mechanism with $N$ links (including the ground) and $J$ joints, each with $f_i$ degrees of freedom is given by Grübler's formula [9, eq 2.4] [23, eq. 1.4]:

$$\text{dof} = m\,(N - 1 - J) + \sum_{i=1}^{J} f_i \tag{1.1}$$

Where $m$ is the number of dof of a rigid body (3 or 6).

### 1.1.2. Workspace

The workspace is the set of all the points that the end-effector can reach. It is a volume in an $r$-dimensional space called task space, that represents the end-effector position and orientation (or attitude). The workspace is intrinsically related to the geometry of the links and how and they are connected by joints; it must also consider joint limits and self-collision of links. The number of variables $r$ to describe the end-effector does not depend on the robot, but on its application. In the most generic case $r = 6$ (i.e. position and rotation of a frame in the 3D space), but some tasks may require less coordinates, for instance $r = 3$ if only the position and not the orientation is relevant. If a task space has only the three spatial directions, it coincides with the real world.

The reachable (or maximal) workspace consist in all the points that are reachable by any robot configuration, while the dexterous workspace describes only those where all orientations can be attained. Obviously the latter is a subset of the firmer. Finally, the points reachable with a fixed end-effector attitude, form the constant-orientation (or translational) workspace [23] [24].

### 1.1.3. Direct Kinematics

It is convenient to express the configuration using the coordinates of the actuated joints, namely the angle or position of their degrees of freedom. This choice is sensible considering that the motors can simply feature an encoder to measure their angular position. These values can be grouped in the joint vector $\mathbf{q} \in \mathbb{R}^n$, where $n$ is the number of dof. For every configuration, represented by a specific joint vector, the end-effector has a certain position and orientation, whose coordinates can be collected in the space vector $\mathbf{x} \in \mathbb{R}^r$.

$\mathbf{q}$ and $\mathbf{x}$ represent respectively a configuration in the joint space, and in the task space. Given the physical meaning of $\mathbf{q}$ and $\mathbf{x}$, it must exist a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^r$ that maps a configuration $\mathbf{q}$ into its corresponding task space value $\mathbf{x}$:

$$\mathbf{x} = f(\mathbf{q}) \tag{1.2}$$

It is evident that $f$ disguises the geometry of the links and eventual transmission ratios of the joints. Nonetheless, it might be impossible to obtain a closed-form expression of $f$, so in some cases a more generic function $f(\mathbf{x}, \mathbf{q}) = 0$ must be used.

The computation of $\mathbf{x}$ from a certain known configuration $\mathbf{q}$ is called direct kinematic problem, or forward kinematic problem.

If $n > r$ there are more actuators than strictly needed to move the end-effector in the task space, and the robot is said to be redundant. Of course, for a different task the same robot may not be redundant, unless $n > 6$. In any case, redundancy often translates into greater dexterity and flexibility rather than clumsiness [23] [31]. If instead $n < r$ the robot cannot achieve arbitrary motion of the end-effector [9].

### 1.1.4. Jacobian and Differential Forward Kinematics

The quantities $\mathbf{q}$ and $\mathbf{x}$ vary continuously in time: $\mathbf{q}(t)$ and $\mathbf{x}(t)$. Therefore the differentiating Equation 1.2 with respect to time yields:

$$\dot{\mathbf{x}} = \frac{\partial f}{\partial \mathbf{q}} \, \dot{\mathbf{q}} \tag{1.3}$$

$$=: J(\mathbf{q}) \, \dot{\mathbf{q}} \tag{1.4}$$

The matrix function $J : \mathbb{R}^n \to \mathbb{R}^{r \times n}$ is called analytic Jacobian, and it relates the joint velocities $\dot{\mathbf{q}}$ with the velocities of the end-effector in the working space [9] [31].

Letting $\mathbf{J}_i$ be the $i$-th column of $J(\mathbf{q})$, and $\dot{q}_j$ the $j$-th element of $\dot{\mathbf{q}}$, Equation 1.4 can be rewritten to:

$$\dot{\mathbf{x}} = \dot{q}_1 \, \mathbf{J}_1 + \dot{q}_2 \, \mathbf{J}_2 + \cdots + \dot{q}_n \, \mathbf{J}_n \tag{1.5}$$

Equation 1.5 shows that the working space velocity is a linear combination of the columns of the Jacobian matrix. Therefore the $i$-th column of $J(\mathbf{q})$ is the direction in which the $i$-th dof moves the end-effector.

Similarly to the direct kinematics problem, the differential forward kinematics problem consists in finding the working space velocities $\dot{\mathbf{x}}$ when the joints are moving with velocities $\dot{\mathbf{q}}$. Of course, the differential problem yields different results in different configurations $\mathbf{q}$.

### 1.1.5. Inverse Kinematics

The inverse kinematics problem is the search of a posture $\mathbf{q}$ that places the end-effector in $\mathbf{x}$. The problem is generally difficult for open-chain mechanisms and easier for closed-chain mechanisms. Contrary to the forward kinematics, which always has exactly one solution for closed loops, a solution to the inverse kinematics problem might not exists or, if exists, there might be multiple or infinite solutions. In case $n > r$ the robot has dof in excess and a point in the working space can be reached by multiple configurations. If instead $n < r$ only $n - r$ working space variables can be controlled, thus a solution might not exist. Therefore the problem

$$\mathbf{q} = f^{-1}(\mathbf{x}) \tag{1.6}$$

is well-posed only if $n = r$, which is a necessary condition for $f(\mathbf{q})$ to be invertible. Even if $\mathbf{x}$ belongs to the workspace and a solution $\mathbf{q}$ exists, it might not be easy to compute it analytically through the inversion of $f$ because in general it is a complicated nonlinear function. Furthermore any solution must also consider geometrical limits, such as limited joint rotations and self-collisions.

For this reason numerical computation are often implemented in robotic controllers. This approach also allows to account for link limits and obstacles in the workspace. Moreover, relying on analytic solutions might provide wrong result in case of geometric imprecision such as misaligned axis. In this case the analytic solution can be used as the initial guess in an iterative numerical computation relying solely on $f(\mathbf{q})$ [9].

### 1.1.6. Differential Inverse Kinematics

The differential inverse kinematics problem consist of finding the joint velocity $\dot{\mathbf{q}}$ that causes the end-effector to have a given velocity $\dot{\mathbf{x}}$. As for the inverse kinematics problem, solving analytically

$$\dot{\mathbf{q}} = J^{-1}(\mathbf{q})\,\dot{\mathbf{x}} \tag{1.7}$$

is only possible if the Jacobian is square ($n = r$) and has maximum rank (thus it can be inverted). Therefore also the differential kinematics is conveniently solved with numerical approaches.

If $n \neq r$ the actuator velocity can be computed exploiting the Moore-Penrose pseudo-inverse $J^\dagger$:

$$\dot{\mathbf{q}} = J^\dagger(\mathbf{q})\,\dot{\mathbf{x}} \tag{1.8}$$

If the Jacobian is fat (namely has more columns than rows, so $n > r$), $J^\dagger = J^\mathsf{T}(JJ^\mathsf{T})^{-1}$ (right pseudo-inverse), and the solution found is the one with minimum speed [9]. If instead the Jacobian is tall ($n < r$) an exact solution does not exist; the pseudo-inverse is $J^\dagger = (J^\mathsf{T}J)^{-1}J^\mathsf{T}$ (left pseudo-inverse), and $\dot{\mathbf{q}}$ minimizes the distance $\|J(\mathbf{q})\,\dot{\mathbf{q}} - \dot{\mathbf{x}}\|$ between the computed and desired end-effector velocity [9]. If $n > r$, the pseudo-inverse matrix can be weighted using, for instance, the mass matrix $M$ to reduce the motion of heaviest links [9]:

$$J^\dagger = M^{-1}J^\mathsf{T}(JM^{-1}J^\mathsf{T})^{-1} \tag{1.9}$$

### 1.1.7. Statics

When a mechanism is not accelerating, for the law of conservation of power, the power at the joints equals the power at the end-effector:

$$\mathbf{T}^\mathsf{T}\,\dot{\mathbf{q}} = \mathbf{F}^\mathsf{T}\,\dot{\mathbf{x}} \tag{1.10}$$

Where $\mathbf{T} \in \mathbb{R}^n$ and $\mathbf{F} \in \mathbb{R}^r$ are respectively the generalised forces on the motor and on the end-effector[1], in the same order as in vectors $\mathbf{q}$ and $\mathbf{x}$.

Equation 1.4 can now be used in (1.10) to compute the $n$ motor torques equivalent to the force $\mathbf{F}$ on the end-effector [9, eq. 5.26] [24, eq. 18.9] [31, eq. 3.111]:

$$\mathbf{T}^\mathsf{T} = \mathbf{F}^\mathsf{T} J(\mathbf{q}) \tag{1.11}$$

$$\mathbf{T} = J^\mathsf{T}(\mathbf{q})\,\mathbf{F} \tag{1.12}$$

### 1.1.8. Singularities

A kinematic singularity (or just singularity) is a configuration $\mathbf{q}$ where the end-effector suffers an instantaneous loss of mobility in one or more directions.

For the sake of simplicity, in this paragraph it is assumed that $n = r$.

Considering the interpretation of the Jacobian of Equation 1.5, if the rank of $J(\mathbf{q})$ is less than $r$, $\dot{\mathbf{x}}$ cannot have arbitrary values in the $\mathbb{R}^r$ task space. Therefore the configurations $\mathbf{q}$ where the Jacobian is rank deficient (or singular, thus $\det J(\mathbf{q}) = 0$) are singularities [9] [31]. For the rank-nullity theorem:

$$\dim\ker J(\mathbf{q}) + \operatorname{rank} J(\mathbf{q}) = n \tag{1.13}$$

If the rank is not maximum and equal to $n$, the kernel of $J(\mathbf{q})$ has non null size. The kernel of the linear transformation $\dot{\mathbf{x}} = J(\mathbf{q})\,\dot{\mathbf{q}}$ represents the vectors $\dot{\mathbf{q}}$ that map into $\mathbf{0}$. This means that in singularity points the end-effector can stay still ($\dot{\mathbf{x}} = \mathbf{0}$) even if some joints have a so-called internal motion [31].

[1] Generalised forces are torques or forces. From now on the actuator's generalised forces in vector $\mathbf{T}$ are simply called torques, and the generalised forces on the end-effector in vector $\mathbf{F}$ are called forces.

### 1.1.9. Manipulability

An interesting characteristic of a robot is how the working space velocities translate into joint space velocities. This is particularly interesting in the neighbourhood of a singular point. The manipulability and force ellipsoid can help to quantify the ease of motion in a point, and how close that point is to a singularity.

### Velocity Manipulability Ellipsoid

The manipulability ellipsoid is the representation of the working space velocity in every direction $\dot{x}_i$ for all the possible unitary joint speeds. Unitary joint speeds are such that $\|\dot{\mathbf{q}}\| = 1$, or equivalently $\dot{\mathbf{q}}^\mathsf{T}\,\dot{\mathbf{q}} = 1$; this equation represents an $n$-dimensional sphere in the joint velocity space. The corresponding volume in the working velocity space can be obtained with the following passages, assuming that the Jacobian $J$ is invertible (thus square and full rank) [9, eq. 5.27] [31, eq. 3.123]:

$$
\begin{aligned}
1 &= \dot{\mathbf{q}}^\mathsf{T}\,\dot{\mathbf{q}} \qquad (1.14)\\
&= (J^{-1}\,\dot{\mathbf{x}})^\mathsf{T}\,(J^{-1}\,\dot{\mathbf{x}})\\
&= \dot{\mathbf{x}}^\mathsf{T} J^{-\mathsf{T}}\,J^{-1}\,\dot{\mathbf{x}}\\
&= \dot{\mathbf{x}}^\mathsf{T}(J\,J^\mathsf{T})^{-1}\,\dot{\mathbf{x}}\\
&=: \dot{\mathbf{x}}^\mathsf{T} A\,\dot{\mathbf{x}} \qquad (1.15)
\end{aligned}
$$

The equation $\dot{\mathbf{x}}^\mathsf{T} A\,\dot{\mathbf{x}} = 1$ represents an ellipsoid centred in the origin of an $r$-dimensional space. Letting $\lambda_i$ be the eigenvalues of its inverse $JJ^\mathsf{T} \in \mathbb{R}^{r\times r}$, the eigenvalues of $A := (JJ^\mathsf{T})^{-1}$ are $1/\lambda_i$. The lengths $a_i$ of the principal semi-axis of the ellipsoid are the inverse of the square root of the eigenvalues of $A$, and the volume of the ellipsoid is proportional to the inverse of the square root of the determinant of $A$ [31, eq. 3.124]:

$$
a_i = \frac{1}{\sqrt{1/\lambda_i}} = \sqrt{\lambda_i} \qquad (1.16)
$$

$$
\begin{aligned}
V \propto \frac{1}{\sqrt{\det A}} &= \frac{1}{\sqrt{\frac{1}{\lambda_1}\,\frac{1}{\lambda_2}\cdots\frac{1}{\lambda_n}}} = \sqrt{\lambda_1\,\lambda_2\cdots\lambda_3}\\
&= \sqrt{\det(JJ^\mathsf{T})} = |\det J|
\end{aligned} \qquad (1.17)
$$

The direction of the principal axes of the ellipsoid corresponds to the eigenvectors of $A$, which are the same of $JJ^\mathsf{T} = A^{-1}$.

## Indexes of Manipulability

Since in a singularity point the motion in at least one direction is not possible, the velocity in that direction is zero, and consequently the volume of the ellipsoid in the velocity space is also zero. For this reason the volume (or rather a quantity proportional to the volume $V$) is an index of proximity to a singularity [31]:

$$v := |\det J| \tag{1.18}$$

Another useful index is the ratio between the longest and the shortest semi-axis of the ellipsoid:

$$\mu := \frac{a_{\max}}{a_{\min}} = \sqrt{\frac{\lambda_{\max}}{\lambda_{\min}}} \geq 1 \tag{1.19}$$

Values of $\mu$ close to 1 are a synonym of isotropy of motion, because they indicate a spherical manipulability ellipsoid, hence it is equally easy to move the end-effector in any direction. $\mu$ is just the square root of the condition number of the matrix $JJ^\mathsf{T}$ [9].

## Force Ellipsoid

A similar process can be carried out for forces, considering the working space force corresponding to a torques vector $\mathbf{T}$ of unitary length ($\mathbf{T}^\mathsf{T}\mathbf{T} = 1$). Repeating the passages from (1.14) to (1.15) using Equation 1.12 yields [31, eq. 3.126]:

$$\begin{aligned} 1 &= \mathbf{F}^\mathsf{T}(JJ^\mathsf{T})\,\mathbf{F} && (1.20) \\ &=: \mathbf{F}^\mathsf{T} A^{-1}\,\mathbf{F} && (1.21) \end{aligned}$$

The resulting equation is again an ellipsoid, called force ellipsoid. Since the force ellipsoid matrix $A^{-1}$ is the inverse of the manipulability matrix, it has the same eigenvalues but reciprocal eigenvectors. Consequently the force ellipsoid has the same axis as the manipulability ellipsoid, but the lengths of the semi-axis are the reciprocals, as shown in Figure 1.2.

## Interpretation of Ellipsoids

The reciprocal dimension of the axis of the manipulability and force ellipsoid suggests that when the end-effector can attain high velocity motion in a direction, it can bear small forces in that direction. And when the end-effector struggle to move in a certain direction, it can resist high loads in the same direction.

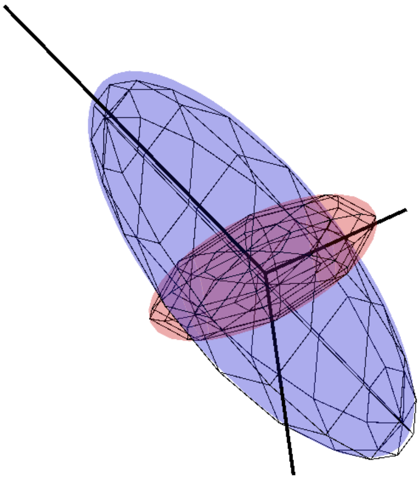


Figure 1.2: Example of manipulabilty and force ellipsoid. The ellipsoids have the same axis direction but inverse axis lengths.

Close to a singularity the manipulability ellipsoid is almost degenerated: it is very stretched in some directions and compressed in others. Since it represents the points in the task velocity space where the joint space speed is 1, end-effector velocities in the direction where the ellipsoid is thinner are converted to very large joint velocities. For the same reason, forces in the directions where the force ellipsoid is thin (and thus the manipulability ellipsoid is large) require large torques to be balanced.

Extending this argument to a singular configuration, the direction where the motion is not possible is the same where unlimited force can be borne (in the sense that it does not require motor actuation for balancing, but it is still subject to structural limits of the links and of the joints.), while forces in the other directions can generate infinite torques on the actuators.

Nonetheless working near a singularity may be desirable in order to exploit the large amplification factor between joints and end-effector. Some applications are accurate positioning in a small volume and force sensing in a particular direction as done by [28] with a Gough-Stewart platform.

## 1.2. Parallel Kinematics

A parallel kinematics machine (or parallel kinematic structure, or just parallel robot) is a closed-chain mechanism, in which a link is connected to the ground by multiple independently actuated chains. Every kinematic chain connected in parallel is called limb or leg. A parallel robot comprises a base platform and a moving platform connected by

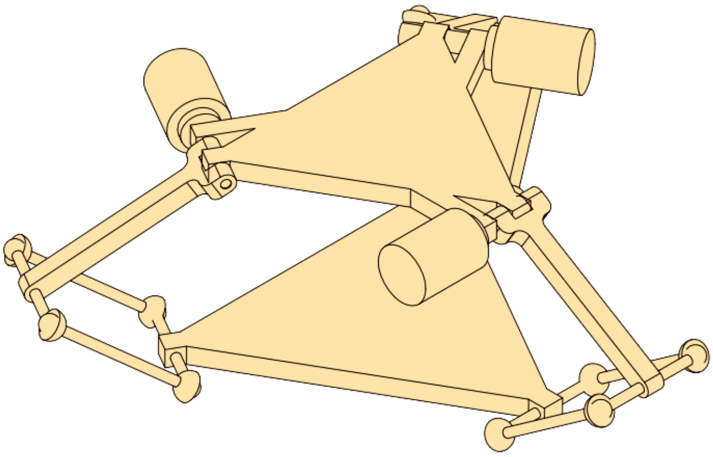


Figure 1.3: Schematic representation of a Delta robot [24, fig. 18.2]. The bottom platform has three translational dof, and is actuated by three motors that are grounded and do not weight on the links.

limbs [9], as visible in the Delta robot and Gough-Stewart platform in Figures 1.3 and 1.4. On the contrary, single-chain mechanisms are called serial robots. While in serial robots all joints must be actuated to allow a rigid connection of the end-effector to the ground, a parallel robot might have passive joints, which are unactuated.

### 1.2.1. Advantages of Parallel Kinematics

Regarding the design, it is relatively easy to scale a parallel robot to build a larger or smaller version. Furthermore, they can exploit the repetition of identical limbs around a symmetry axis, which grants some economic perks (simpler design, reduction of the number of different components, easier assembly, . . . ).

Often parallel robots have higher dynamic performances compared to equivalent serial robots devoted to similar tasks. The reason is that the unactuated links are free of the motor mass, and the motors of the first link of the limbs can be grounded (like in the Delta robot, in Figure 1.3), or just slightly moving (like in the Gough-Stewart platform in Figure 1.4) [24] [23].

The many number of chains in parallel have the advantage of increasing the system stiffness, and allowing motion with great precision. This advantages also the velocity and acceleration in the working space [24].

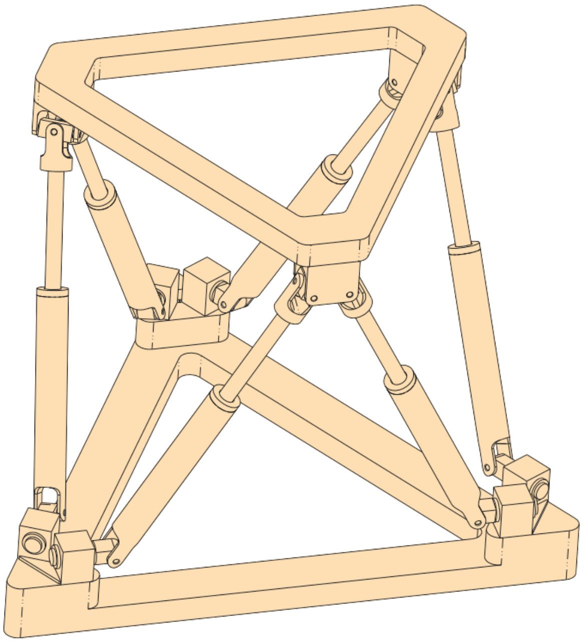


Figure 1.4: Schematic representation of a Gough-Stewart platform [24, fig. 18.1]. The top platform has six dof, and is moved by six linear motors connected to the grounded bottom platform.

### 1.2.2. Disadvantages of Parallel Kinematics

An important disadvantage of parallel kinematics machines is their small workspace (or small ratio between the usable workspace and the overall machine size), in particular considering that closed-chain mechanisms are bulky. The workspace has strange shape and is difficult to describe, and in particular the rotation limits of the end-effector change depending on its position. Moreover, the workspace frequently contains singularities and the behaviour in the working volume is highly anisotropic.

### 1.2.3. Inverse Kinematics

For parallel mechanisms, the inverse kinematics problem is easier than the direct problem because the joint variables can be simply determined by closure equations [24] [23]. The inverse kinematics problem is

$$\mathbf{q} = k(\mathbf{x}) \tag{1.22}$$

If instead the joint space parameters cannot be isolated, the kinematic relation can be expressed in the more generic form

$$f(\mathbf{q}, \mathbf{x}) = 0 \tag{1.23}$$

### 1.2.4. Direct Kinematics

The direct kinematics problem of parallel robots is much more difficult than the inverse problem. The end-effector coordinates can be found solving complicate equations, but the solution is not unique, as there are several ways to assemble the mechanism given certain joint variables $\mathbf{q}$. To discriminate the correct configuration amongst the many viable, some sensors can be placed on passive joints or between limbs [23].

The computation of direct kinematics relies on numerical iterations.

### 1.2.5. Singularities

A singularity arises whenever the Jacobian fails to be full rank. If an analytic matrix function representing the Jacobian is not available, the $n$ closure equations in (1.23) can be exploited to identify singularities. Differentiating Equation 1.23 with respect to time yields:

$$\begin{aligned} 0 &= \frac{\partial f}{\partial \mathbf{q}} \dot{\mathbf{q}} + \frac{\partial f}{\partial \mathbf{x}} \dot{\mathbf{x}} && (1.24) \\ &=: J_q(\mathbf{q}, \mathbf{x}) \, \dot{\mathbf{q}} + J_x(\mathbf{q}, \mathbf{x}) \, \dot{\mathbf{x}} && (1.25) \end{aligned}$$

[7] defines three types of singularities for closed loop kinematics:

- serial singularity when $\det J_q(\mathbf{q}, \mathbf{x}) = 0$. The end-effector can have zero velocity when the joints are moving. The end-effector loses dof because it is not possible to prescribe velocities in some directions;
- parallel singularity when $\det J_x(\mathbf{q}, \mathbf{x}) = 0$. In this singularity point the end-effector can move while the joints have zero velocity, therefore it gains dof which cannot be controlled;
- both $J_q(\mathbf{q}, \mathbf{x})$ and $J_x(\mathbf{q}, \mathbf{x})$ are simultaneously singular. This situation corresponds to a degenerate case of the Equation 1.23. The end-effector can undergo finite motion when the actuator are locked, or a finite motion of the actuators does not move the end-effector.

## 1.3. Delta Robots

A Delta robot is a three dof a parallel kinematics mechanism whose moving platform has constant attitude (constant orientation regarding the base in the horizontal plane - the moving platform is always parallel to the base). The moving platform is connected to the ground by three legs. Every leg features a motor that actuates the first link, and a parallelogram mounted via universal or spherical joints onto the moving platform and the first link. The parallelograms are the key element to keep the moving platform at constant orientation. Figure 1.3 represent a schematic Delta robot in which two triangular platforms are visible: the top one is grounded, and the bottom one is moving parallel to the first.

Because the three actuators can be grounded, the moving parts are relatively light and thus the Delta robot can achieve very fast motions. It is therefore widespread for assembly, packaging, and pick and place of lightweight objects. Thanks to its stiffness and precision, it has also found applications for additive manufacturing [2, 3, 29], haptics [8, 25–27, 30] and even in the medical field [32].

Some commercial Delta robot are shown in Figure 1.5. All of them feature parallelograms and links free of motors; the robots in Figure 1.5b and 1.5c have extra rods to rotate the end-effector, thereby having more than three dof.

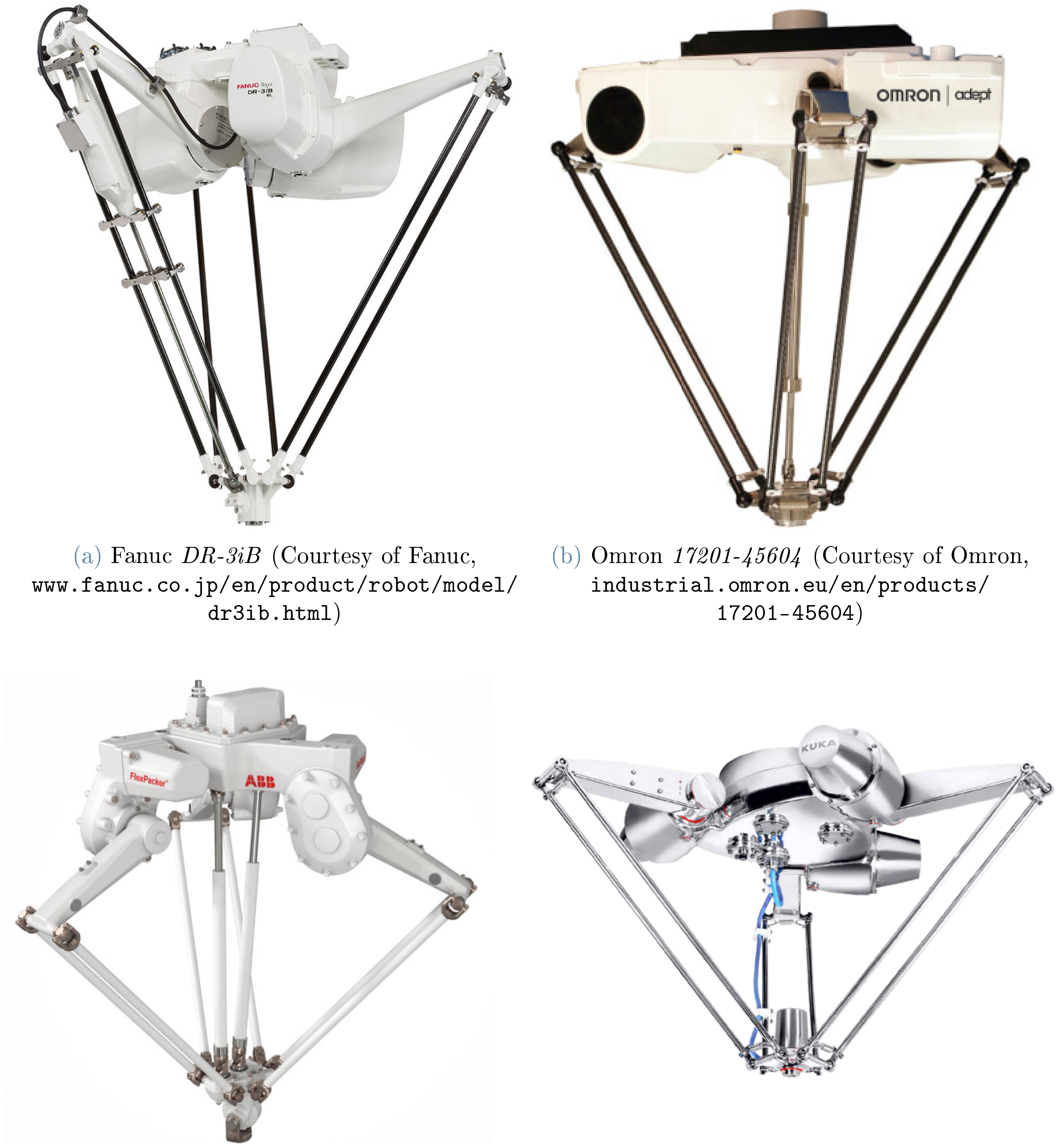
### 1.3.1. History

The Delta robot was invented in the early '80s by Professor Reymond Clavel in the laboratory of the École Polytechnique Fédérale de Lausanne (EPFL) in Switzerland. The original version patented in 1990 (shown in Figure 1.7) has the possibility to rotate the end-effector with a fourth motor mounted on the fixed base that transmits the rotation via a thin rod. It was firstly produced by the Swiss company Demaurex in 1987 [1].

### 1.3.2. Workspace

The typical workspace of an industrial Delta robot is a wide cylinder with a short truncated inverted cone on the bottom. Figure 1.6 shows the workspaces of the Delta robot in Figure 1.5.

As for most of parallel kinematics robots, the workspace is limited compared to the extension of the mechanism. Further dof that rotate a wrist on the moving platform allow to have a larger workspace, such as for the ABB *IRB 390* in Figure 1.6c.



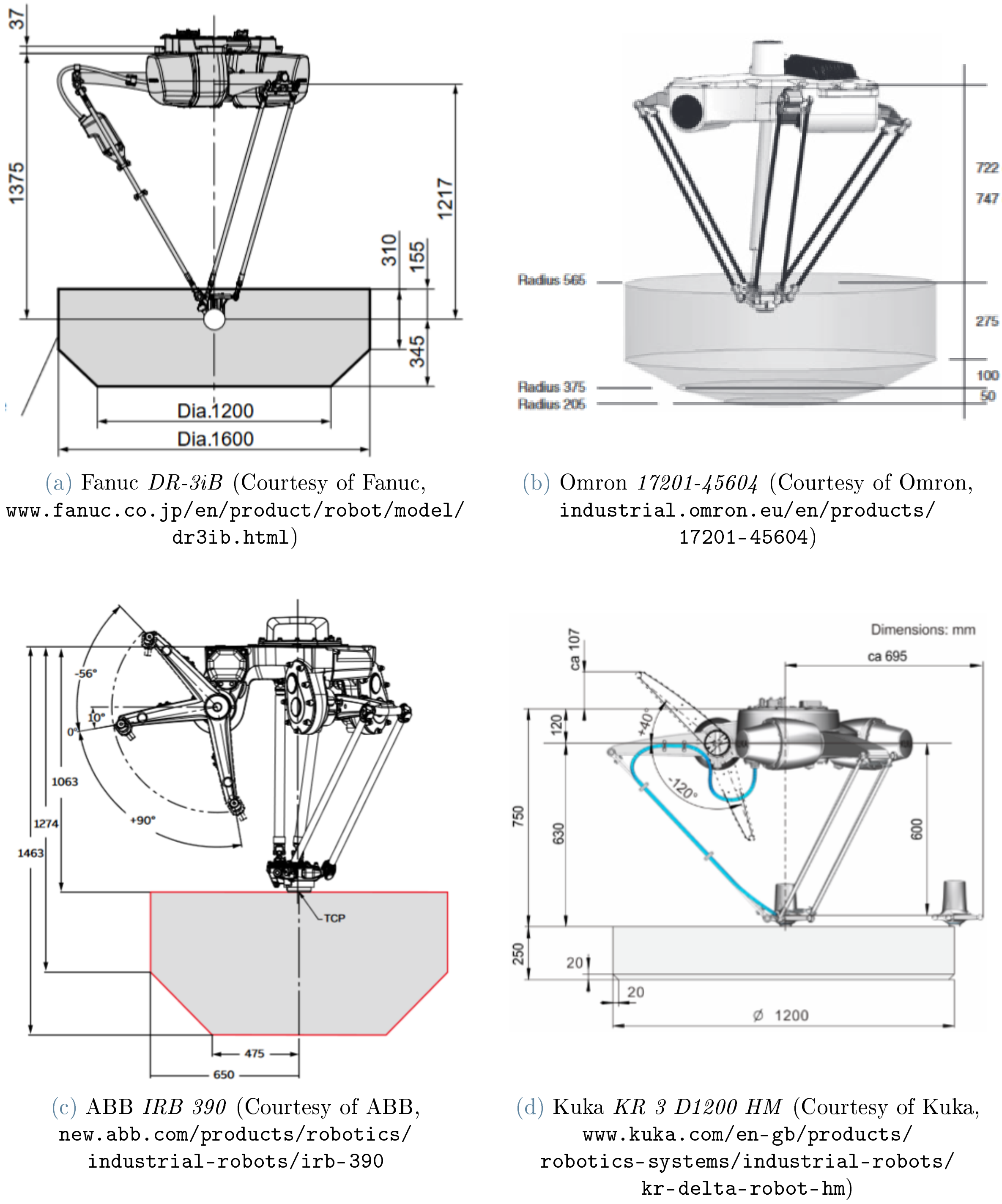
(a) Fanuc *DR-3iB* (Courtesy of Fanuc, `www.fanuc.co.jp/en/product/robot/model/dr3ib.html`)

(b) Omron *17201-45604* (Courtesy of Omron, `industrial.omron.eu/en/products/17201-45604`)

(c) ABB *IRB 390* (Courtesy of ABB, `new.abb.com/products/robotics/industrial-robots/irb-390`)

(d) Kuka *KR 3 D1200 HM* (Courtesy of Kuka, `www.kuka.com/en-gb/products/robotics-systems/industrial-robots/kr-delta-robot-hm`)

Figure 1.5: Some commercial Delta robots.



(a) Fanuc *DR-3iB* (Courtesy of Fanuc, `www.fanuc.co.jp/en/product/robot/model/dr3ib.html`)

(b) Omron *17201-45604* (Courtesy of Omron, `industrial.omron.eu/en/products/17201-45604`)

(c) ABB *IRB 390* (Courtesy of ABB, `new.abb.com/products/robotics/industrial-robots/irb-390`

(d) Kuka *KR 3 D1200 HM* (Courtesy of Kuka, `www.kuka.com/en-gb/products/robotics-systems/industrial-robots/kr-delta-robot-hm`)

Figure 1.6: Workspaces of the robots in Figure 1.5.

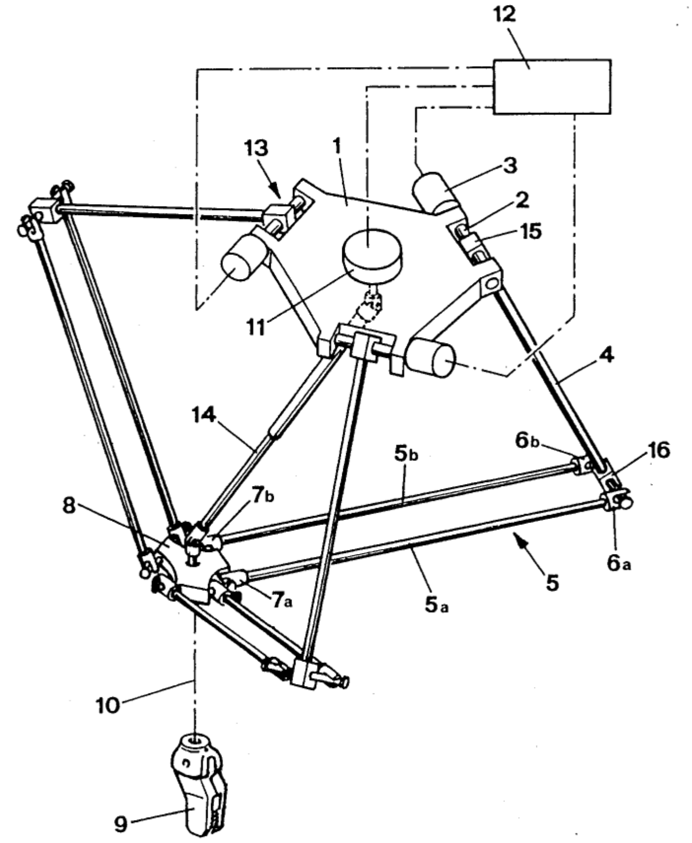


Figure 1.7: Clavel's Delta robot [4, fig. 1].

# 2 Introduction to Fanuc M-1$i$A/0.5A

Fanuc produces four families of Delta robots: M-1$i$A, M-2$i$A, M-3$i$A and DR-3$i$B. The M-1$i$A version has 6 models, which are the combination of short or large motion range and different number of axes. The models of the series M-1$i$A are collected in Table 2.1 The repeatability for all the M1-$i$A products is ±0.02 mm [5, 6].

A six dof M-1$i$A/0.5A is shown in Figure 2.1, in which are recognisable the three limbs with a parallelogram with spherical joints that connect the moving platform to the base. The moving platform is a three dof Wrist actuated by the rotation of three links departing from the Base.

## 2.1. Kinematic Chain

The M-1$i$A Delta robots feature an unusual link configuration, different from the one used in most of the commercial Delta robot (for instance, those in Figure 1.5). Figure 2.2 shows a drawing of the M-1$i$A/0.5A under the Cover. The kinematic chain is based on the swing of two semi-rotary arms connected by tie rods. The primary arm (Arm 3 in

**Models of Fanuc M-1$i$A**

| | | | **Axes** | | |
|---|---|---|---|---|---|
| | | | *6* | *4* | *3* |
| **Motion range** | *Small* | ⌀ 280 mm<br>$H$ 100 mm | M-1$i$A/0.5A<br>(0.5) | M-1$i$A/0.5S<br>(0.5) | M-1$i$A/1H<br>(1) |
| | *Large* | ⌀ 420 mm<br>$H$ 150 mm | M-1$i$A/0.5AL<br>(0.5) | M-1$i$A/0.5SL<br>(0.5) | M-1$i$A/1HL<br>(1) |

Table 2.1: Models names and payloads in kg (in round brackets) of the Fanuc robot series M-1$i$A [6].

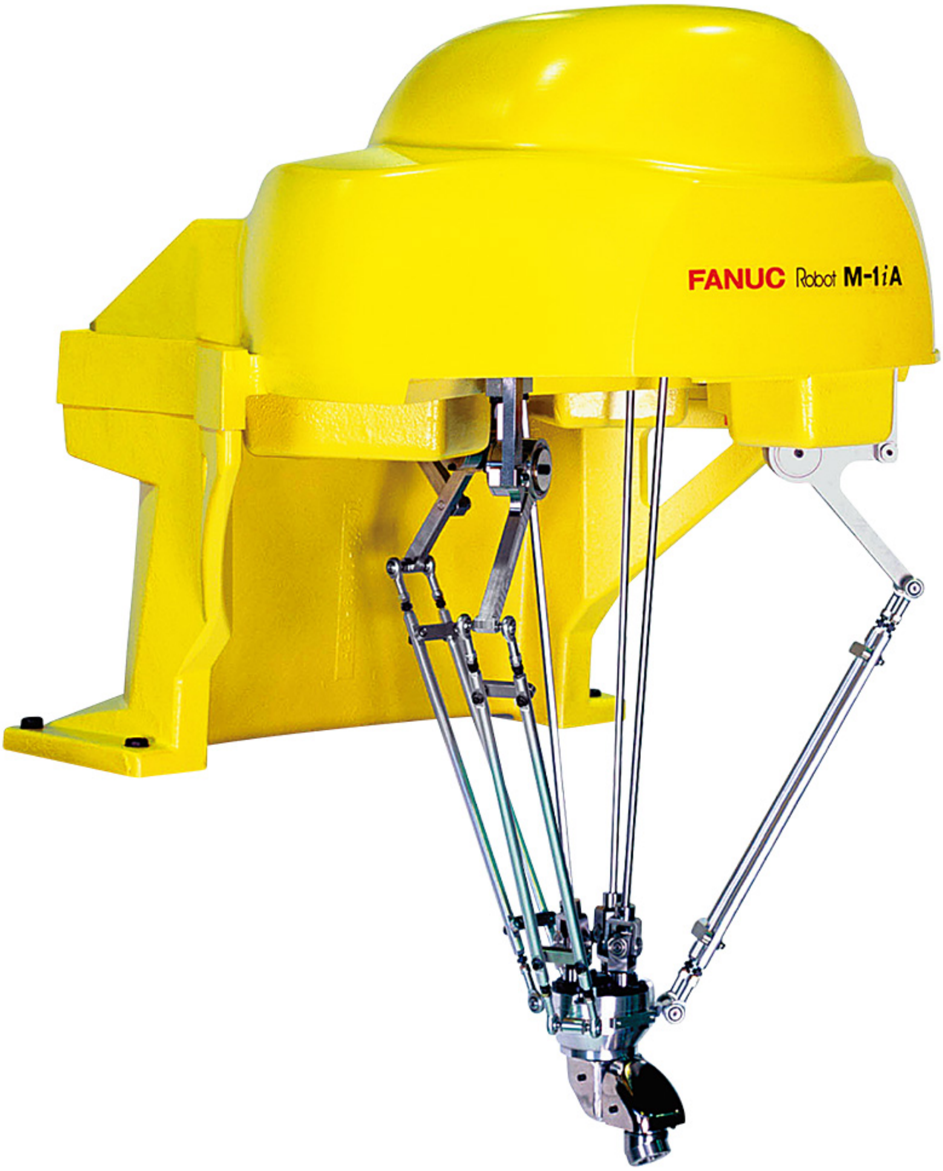


Figure 2.1: Fanuc M-1*i*A/0.5A. (Courtesy of Fanuc, `fanuc.co.jp/en/product/robot/f_r_genkotu.html`)

Figure 2.3) is linked to the parallelogram, while the secondary arm (Arm 1) is connected to the motor. This configuration allows to move the drive units into the top Cover, thereby freeing space in the area between the primary rotary joints (points B in Figure 2.3). The free area is needed to accommodate the three links that drive the Wrist dof, and at the same time to eliminate some of the possible collisions states.

The translation in space of the end-effector is actuated by the first three dof, called J1, J2 and J3, which move Leg 1, Leg 2 and Leg 3 respectively by rotating the Arm 1 of the kinematic chain shown in Figure 2.3.

The three Legs are placed at 120 deg around the central axis of the workspace. All the Legs have identical Arm 1 and Arm 2. Moreover, the distances BD and BE of Arm 3 (see Figure 2.3) are the same for all the Legs.

The other three dof J4, J5 and J6 are responsible for the rotation of the Wrist, and are actuated by three motors mounted in the . The rotation is transmitted via three rods mounted on prismatic joints to accommodate different distances to the end-effector as it moves in the workspace. The product datasheet claims that J4, J5 and J6 can rotate at up to 1440 deg/s [5, 6].

## 2.2. Workspace

The workspace of the M-1*i*A/0.5A is shown in Figure 2.4. It was obtained from the *Fanuc Roboguide* software and imported into *PTC Creo Parametric*. Figure 2.4 contains also the world origin frame of the robot, which was adopted throughout this work.

The workspace is the union of a cylinder and two frustums (truncated cones). Their axis are aligned with the $z$ world axis.

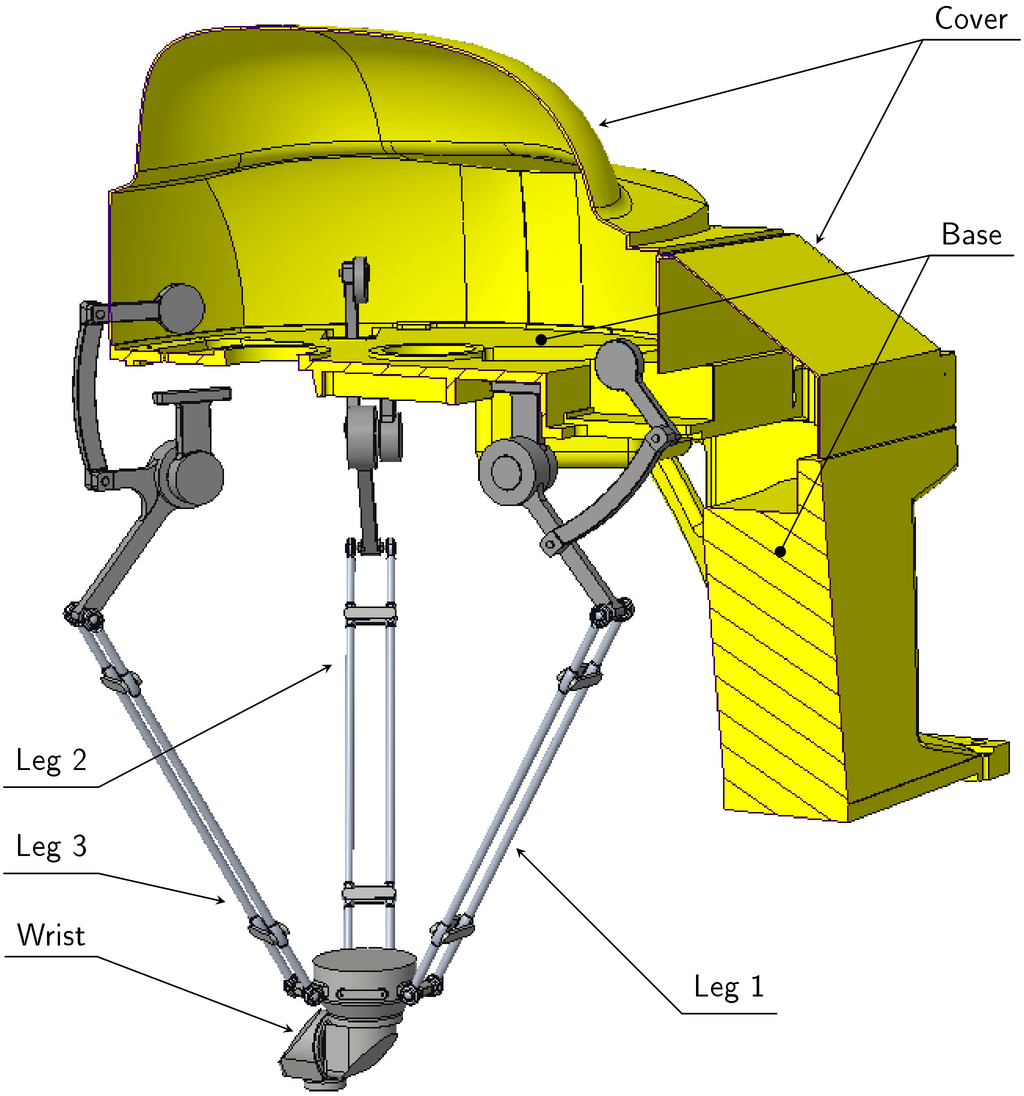
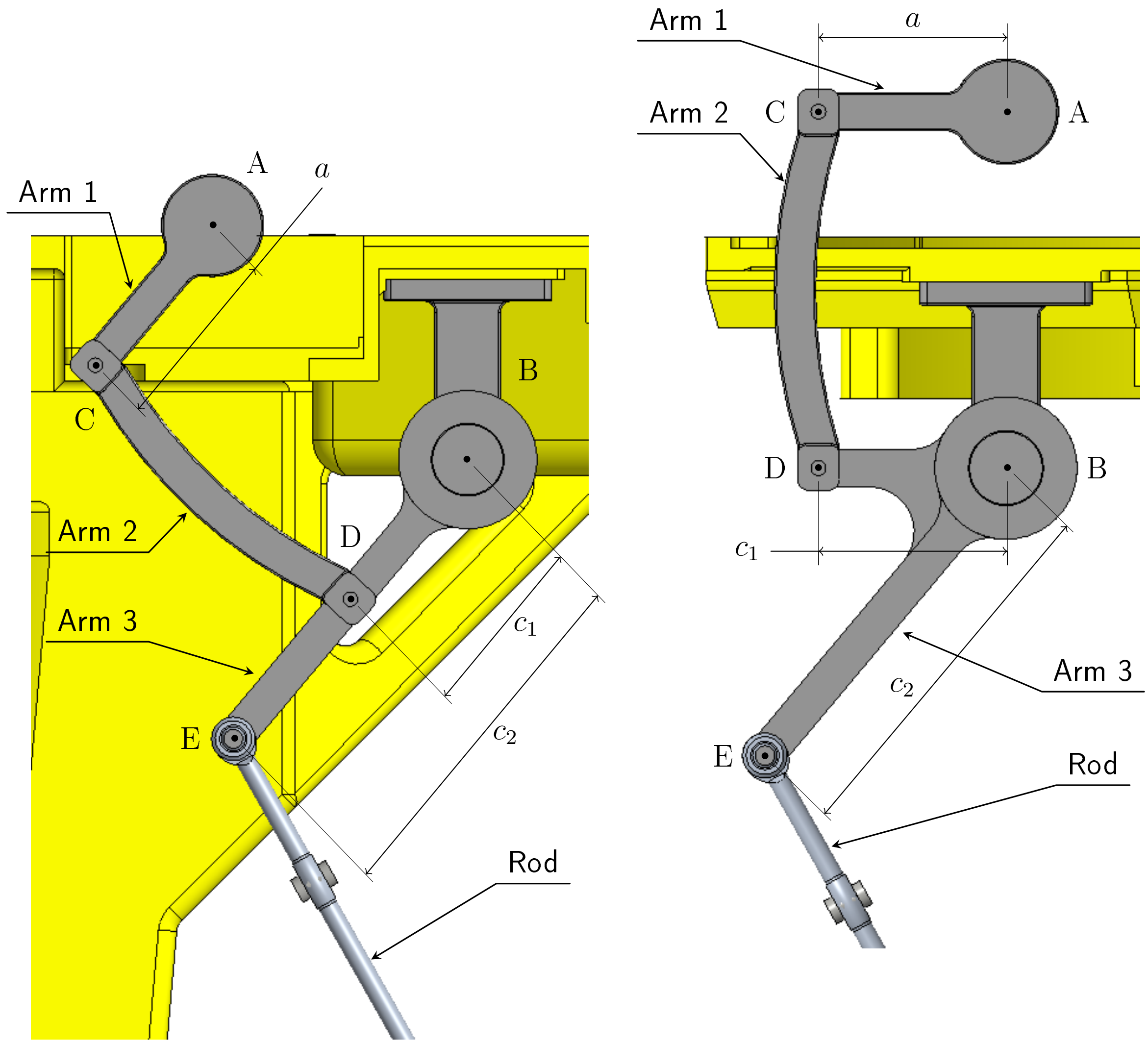


Figure 2.2: Side view of the M-1*i*A/0.5A with the nomenclature used in this document. The Base and Cover are sectioned to show inside. The motors, the transmission gears and the components other than links are not shown. The robot is in the zero configuration.



(a) Kinematic chain of Leg 1. Arm 1 is tilted of 50 deg below the horizontal plane, so that the Arm 3 of all Legs have the same inclination.

(b) Kinematic chain of Leg 2, identical to Leg 3.

Figure 2.3: Links of Leg 1 and Leg 2 of the M-1*i*A/0.5A, along with the nomenclature used further in this document. The kinematic chain shown here is from the motor (not shown) to the Rods constituting the parallelogram, represented in the zero configuration. $a = 50$ mm, $c_1 = 50$ mm and $c_2 = 100$ mm for all the Legs.

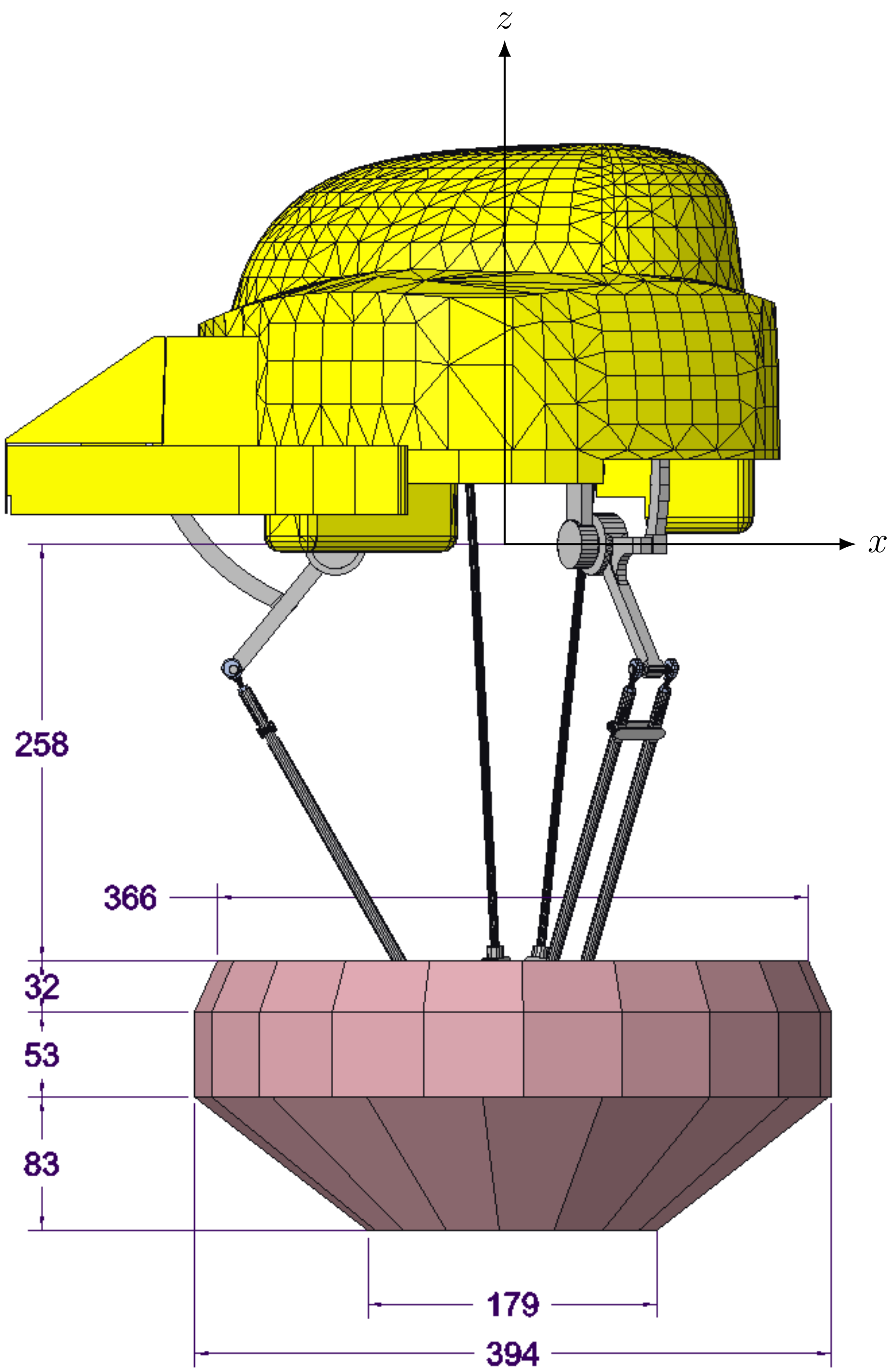


Figure 2.4: Workspace of the M-1*i*A/0.5A obtained from the *Fanuc Roboguide* software. The frame depicted represents the world coordinate system.

# 3 | Introduction to Matlab Simulink and Simscape Multibody

This Chapter introduces the core concepts of *Simscape Multibody* and some features used for the creation of the robot model. After a brief explanation of *Simulink*, the fundamental blocks of *Simscape Multibody* and their most important characteristics are presented. Next, it is discussed how to simulate the model and in particular in which way to control it and how to collect the results.

## 3.1. Simulink and Simscape

*Simulink* is a software to create and simulate continuous or discreet models using a block diagram logic. It is smoothly integrated with the *Matlab* software platform [13]. *Simulink* automatically converts the block diagram to a set of equations (algebraic and/or differential) that are solved in a specified time range.

*Simscape* is a *Simulink* extension used to create models of physical systems within *Simulink*. *Simscape* has a basic so-called *Foundation Library* containing blocks for sources, sensor and broad modellings in various physical fields: electrical, gas, hydraulic, magnetic, mechanical, thermal, ... [15]. *Simscape* can be further extend to create detailed models of complex physical system, by using one or more of the four different extensions [14]:

- *Simscape Electrical* for electronic and electrical power systems;
- *Simscape Driveline* for transmission systems;
- *Simscape Multibody* for 3D multibody systems;
- *Simscape Fluids* for fluid systems.

The four *Simscape* realms and the *Simulink* blocks integrate and work together, therefore a system of engineering interest spanning multiple physical domains can be modelled, controlled and tested in *Simulink*.

## 3.2. Simscape Multibody blocks

*Simscape Multibody* works in the same fashion as any *Simulink* block diagram. The most common blocks used to build *Simscape Multibody* models are of four types [16]:

**body** to define objects;

**transform** to translate and rotate coordinate frames;

**joint** to connect two different bodies;

**force** to exert a force upon a frame.

The general idea is to connect bodies by joints in positions specified using transform blocks.

Furthermore, a model usually requires the following blocks [16]:

**World Frame** to create an inertial reference frame for the model,

**Mechanism Configuration** to introduce gravitational forces in the model,

**Solver Configuration** to set Simscape solver options.

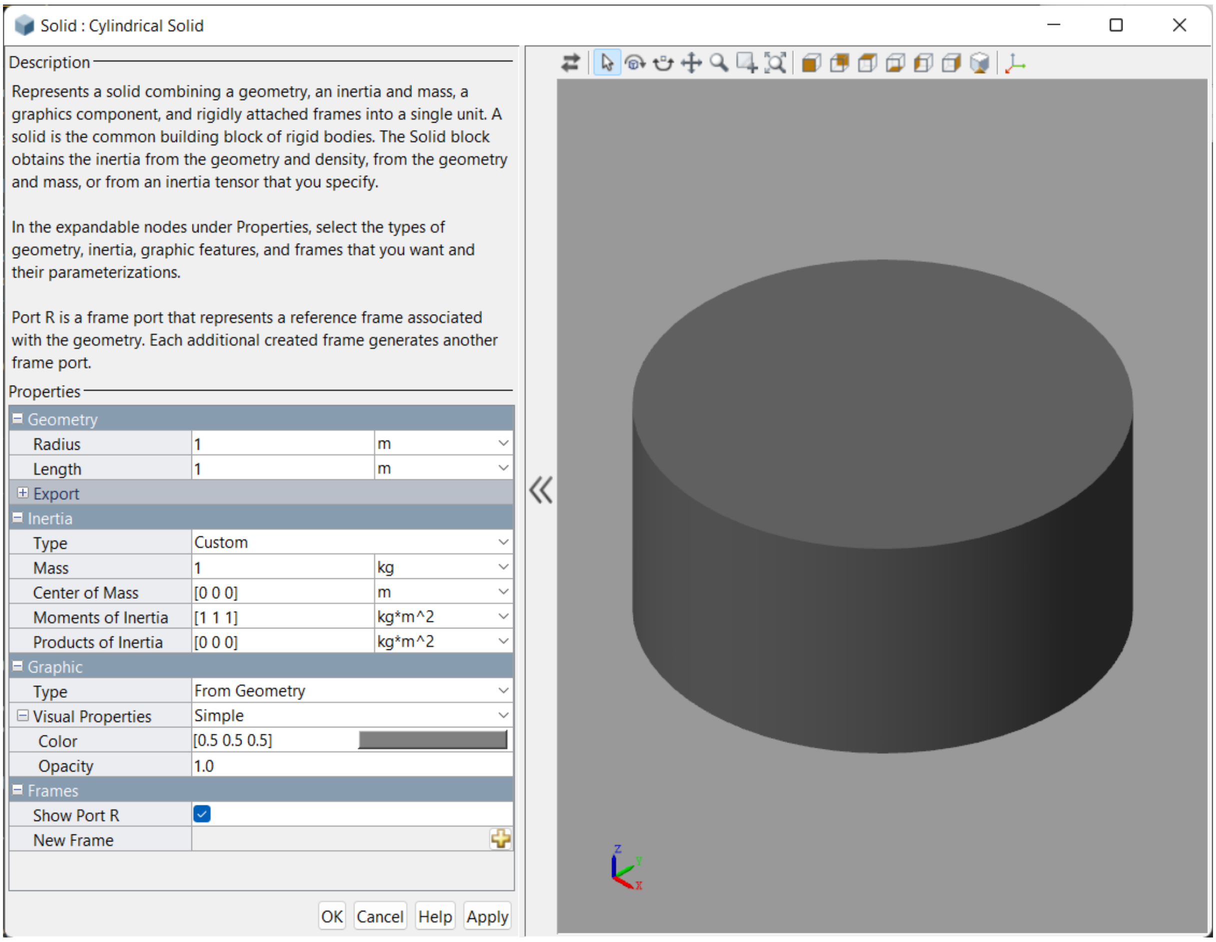
### 3.2.1. Bodies

Bodies represent an object with a mass. Normally they are rigid and with constant mass, but *Simscape Multibody* has the capability to define flexible bodies and variable mass bodies. A body has information about its geometry, its inertia and its graphical representation.

*Simscape Multibody* offers some elementary body geometry with customizable parameters: brick, cylinder, sphere, ellipsoid, and point (eventually including inertia). In alternative, the user can specify a cross-section and build a geometry by extrusion or revolution, or can import a custom CAD file to represent the geometry of a body.

The inertia information comprises the mass, the centre of mass and the inertia matrix. For any type of geometry, the inertial properties can be either automatically evaluated or manually specified. In the firmer case only the density or the mass is required.

The numerical values specified in *Simscape Multibody* must have units, that can be chosen with a drop-down list.



Description

Represents a solid combining a geometry, an inertia and mass, a graphics component, and rigidly attached frames into a single unit. A solid is the common building block of rigid bodies. The Solid block obtains the inertia from the geometry and density, from the geometry and mass, or from an inertia tensor that you specify.

In the expandable nodes under Properties, select the types of geometry, inertia, graphic features, and frames that you want and their parameterizations.

Port R is a frame port that represents a reference frame associated with the geometry. Each additional created frame generates another frame port.

Properties

| | | |
|---|---|---|
| ⊟ Geometry | | |
| Radius | 1 | m |
| Length | 1 | m |
| ⊞ Export | | |
| ⊟ Inertia | | |
| Type | Custom | |
| Mass | 1 | kg |
| Center of Mass | [0 0 0] | m |
| Moments of Inertia | [1 1 1] | kg*m^2 |
| Products of Inertia | [0 0 0] | kg*m^2 |
| ⊟ Graphic | | |
| Type | From Geometry | |
| ⊟ Visual Properties | Simple | |
| Color | [0.5 0.5 0.5] | ▨ |
| Opacity | 1.0 | |
| ⊟ Frames | | |
| Show Port R | ☑ | |
| New Frame | | |

Figure 3.1: Settings of the Cylindrical Solid block, including a preview of the body.

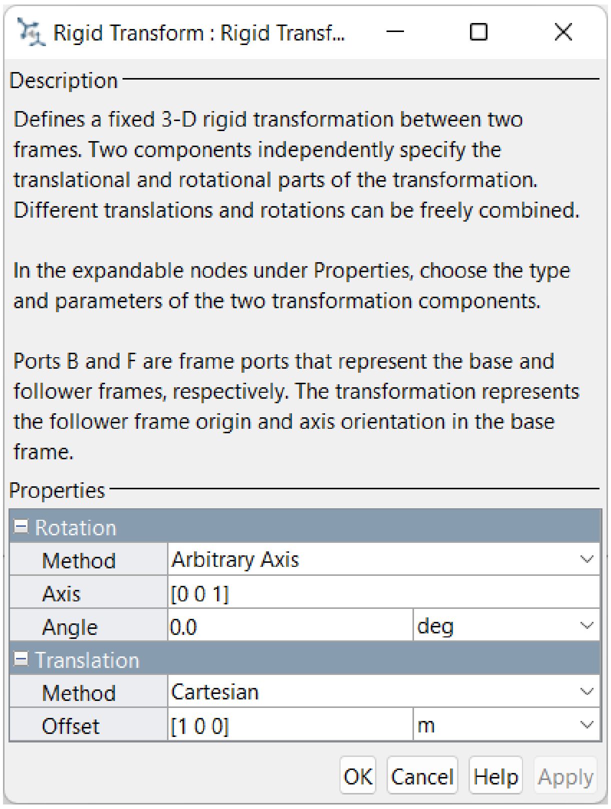
The graphic parameters specify how the solid should be represented in the *Mechanics Explorer*. Apart from the simple colour and transparency, more sophisticate settings can be specified, such as the specular colour, emissivity colour, shininess, and more.

Finally, every object has a default reference frame $R$ corresponding to the body origin. If needed, more reference frames can be defined based on the geometry definition. Another common way to create reference frames is to use Rigid Transform blocks connected to the $R$ frame.

Figure 3.1 shows the interface to edit a Cylindrical Solid block; other body blocks are analogous.

### 3.2.2. Transforms

Coordinate systems (or frames) consist of a point called origin, and three perpendicular axes ($x$, $y$, and $z$) with direction, intersecting in the origin [17]. The Rigid Transform



Rigid Transform : Rigid Transf...

Description

Defines a fixed 3-D rigid transformation between two frames. Two components independently specify the translational and rotational parts of the transformation. Different translations and rotations can be freely combined.

In the expandable nodes under Properties, choose the type and parameters of the two transformation components.

Ports B and F are frame ports that represent the base and follower frames, respectively. The transformation represents the follower frame origin and axis orientation in the base frame.

Properties

| | | |
|---|---|---|
| ⊟ Rotation | | |
| Method | Arbitrary Axis | |
| Axis | [0 0 1] | |
| Angle | 0.0 | deg |
| ⊟ Translation | | |
| Method | Cartesian | |
| Offset | [1 0 0] | m |

Figure 3.2: Settings of the Rigid Transform block.

blocks are used to perform a coordinate change by translating the origin and rotating the axes. The reference frame upon which transformations are applied is called *base frame* $B$, and the resulting frame is the *follower frame* $F$.

Translations can be expressed in Cartesian or cylindrical coordinates. Rotations are defined with respect to the standard $x$ $y$ $z$ axes, a specified axis, using a rotation matrix or using a rotation sequence.

Figure 3.2 shows the interface to edit a Rigid Transform block.

### 3.2.3. Joints

Joints create kinematic constraints between two frames: the *Base frame* and the *Follower frame*. The Table 3.1 summarizes all the joint types and the dof they have with respect to translation and rotation about the standard $x$ $y$ $z$ axis, called *joint primitives* [12]. In this contest, *joint primitive* is a synonym of dof.

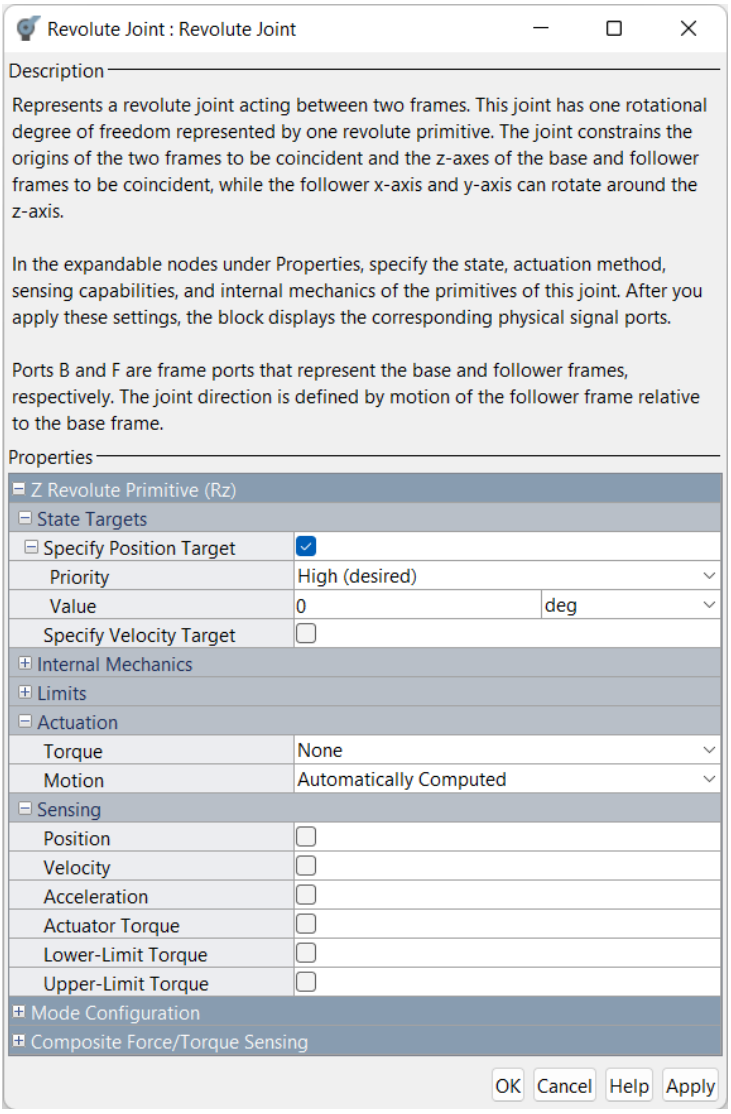
Consider, for instance, the Revolute Joint: it allows rotation between the $z$ axis of the base and follower frame. This implies that both frames must have collinear $z$ axis, and coincident origins. On the contrary, the Cylindrical Joint allows rotation *and* translation

**Joint types**

| Joint block name | Translation | | | Rotation | | | |
|---|---|---|---|---|---|---|---|
| | $x$ | $y$ | $z$ | $x$ | $y$ | $z$ | $S$ |
| 6-DOF | ● | ● | ● | · | · | · | ● |
| Bearing | ○ | ○ | ● | ● | ● | ● | · |
| Bushing | ● | ● | ● | ● | ● | ● | · |
| Cartesian | ● | ● | ● | ○ | ○ | ○ | ○ |
| Constant Velocity* | | | | | | | |
| Cylindrical | ○ | ○ | ● | ○ | ○ | ● | · |
| Gimbal | ○ | ○ | ○ | ● | ● | ● | · |
| Leadscrew* | | | | | | | |
| Pin Slot | ● | ○ | ○ | ○ | ○ | ● | · |
| Planar | ● | ● | ○ | ○ | ○ | ● | · |
| Prismatic | ○ | ○ | ● | ○ | ○ | ○ | ○ |
| Rectangular | ● | ● | ○ | ○ | ○ | ○ | ○ |
| Revolute | ○ | ○ | ○ | ○ | ○ | ● | · |
| Spherical | ○ | ○ | ○ | · | · | · | ● |
| Telescoping | ○ | ○ | ● | · | · | · | ● |
| Universal | ○ | ○ | ○ | ● | ● | ○ | · |
| Weld | ○ | ○ | ○ | ○ | ○ | ○ | ○ |

The joints marked with * cannot be simplified in terms of *joint primitives*.

Table 3.1: The 17 types of joints and their dof. The full dot (●) represents a dof along which motion is allowed, an empty dot (○) represents a constraint. The column $S$ is used when the joint exploits a *spherical primitive* instead of the three *revolute primitive* to eliminate the risk of gimbal lock using a quaternion representation of the rotation. The two representations are mutually exclusive, hence the use of · [18].



Revolute Joint : Revolute Joint

Description

Represents a revolute joint acting between two frames. This joint has one rotational degree of freedom represented by one revolute primitive. The joint constrains the origins of the two frames to be coincident and the z-axes of the base and follower frames to be coincident, while the follower x-axis and y-axis can rotate around the z-axis.

In the expandable nodes under Properties, specify the state, actuation method, sensing capabilities, and internal mechanics of the primitives of this joint. After you apply these settings, the block displays the corresponding physical signal ports.

Ports B and F are frame ports that represent the base and follower frames, respectively. The joint direction is defined by motion of the follower frame relative to the base frame.

Properties

| | |
|---|---|
| ⊟ Z Revolute Primitive (Rz) | |
| ⊟ State Targets | |
| ⊟ Specify Position Target | ☑ |
| Priority | High (desired) |
| Value | 0 deg |
| Specify Velocity Target | □ |
| ⊞ Internal Mechanics | |
| ⊞ Limits | |
| ⊟ Actuation | |
| Torque | None |
| Motion | Automatically Computed |
| ⊟ Sensing | |
| Position | □ |
| Velocity | □ |
| Acceleration | □ |
| Actuator Torque | □ |
| Lower-Limit Torque | □ |
| Upper-Limit Torque | □ |
| ⊞ Mode Configuration | |
| ⊞ Composite Force/Torque Sensing | |

Figure 3.3: Settings of the Revolute Joint block.

about the $z$ axis, therefore the base and follower must have collinear $z$ axis, but the origin of the follower can lye anywhere on the base's $z$ axis.

Some important settings of the Revolute Joint block are illustrated in Figure 3.3. Of particular importance are the sections: *State Target*, *Actuation* and *Sensing* for the only dof of this block: the *Z Revolute Primitive* (Rz). Joints with more dofs have the same options for all their dofs.

## State Target

*State Target* is used to define an optional starting position and velocity of a joint configuration. The user can also decide with which *Priority* the initial conditions of the assembly

**Joint actuation modes**

|  |  | *Motion* |  |
|---|---|---|---|
|  |  | *Provided by Input* | *Automatically Computed* |
| *Torque*<br>*Force* | *None* | Unactuated Motion | Passive |
|  | *Provided by Input* | Fully Specified | Forward Dynamics |
|  | *Automatically Computed* | Inverse Dynamics | Fully Computed |

Table 3.2: Possible combinations of actuation modes for a dof that can be used to achieve different control modes [12].

should be satisfied: *High* or *Low*. It may happen that a model has many valid legal configurations (for instance the so-called *lefty* and *righty* poses of a SCARA robot [9]), in such case it is crucial to specify initial conditions that allow the correct assembly of model.

## Actuation

The *Actuation* settings is used to actuate the joint either by specifying a *Torque* (or *Force* for *translation primitives*) or a *Motion*. The possible combinations and their use are summarised in Table 3.2 [12].

When selecting *Provided by Input*, the joint block offers the possibility to connect another block with the motion or torque signal. The signal must be a physical signal, the unit should be specified (otherwise they are inferred), and it must also include the first and second derivative. These three requirements can be satisfied using the Simulink-PS Converter block, which can be set to automatically compute derivatives if needed [21].

## Sensing

The *Sensing* section is used to specify outputs of the joint blocks. These are the position, velocity and acceleration and force of the dof. If needed, also the constraint forces can be sensed using the *Composite Force/Torque Sensing* section. The output can be converted to a *Simulink* signal by using the PS-Simulink Converter block.

Figure 3.4 shows how to impose a motion and compute the corresponding torque (Inverse Dynamics).

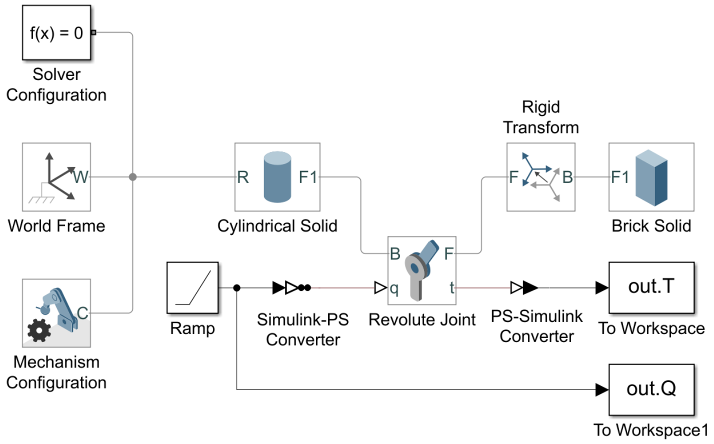


Figure 3.4: A simple mechanical system with two bodies connected by an actuated Revolute Joint which outputs the torque required to accomplish the motion to a To Workspace block. The motion law is given by a Ramp source converted to radiants and derived with the Simulink-PS Converter block.

### 3.2.4. Forces

The blocks of type force are used to introduce forces and torques in the model. Forces can either be an internal or external force. The internal forces are exerted between two frames: the *Base frame* and the *Follower frame*. These are: Gravitational Field, Internal Force, Inverse Square Law Force, Spatial Contact Force, and Spring and Damper Force. The external forces can be introduced using the block External Force and Torque, which requires only the follower frame.

For instance, Figure 3.5 shows a sample use of the force blocks.

## 3.3. Mechanics Explorer

A *Simscape Multibody* model can be visualized in *Matlab* using the *Mechanics Explorer* interactive window. It offers the capabilities to select an object (a body, joint, transform or force) and view its frames, to hide bodies, and to review the block parameters. Of course, it can show the animation of the computed motion.

## 3.4. Principles of Simulation

Once the model is correctly assembled the user should decide how to actuate the model, which input to use and what are the valuable outputs to save. The two main ways to control a multibody model are direct dynamics (and kinematics) and inverse dynamics

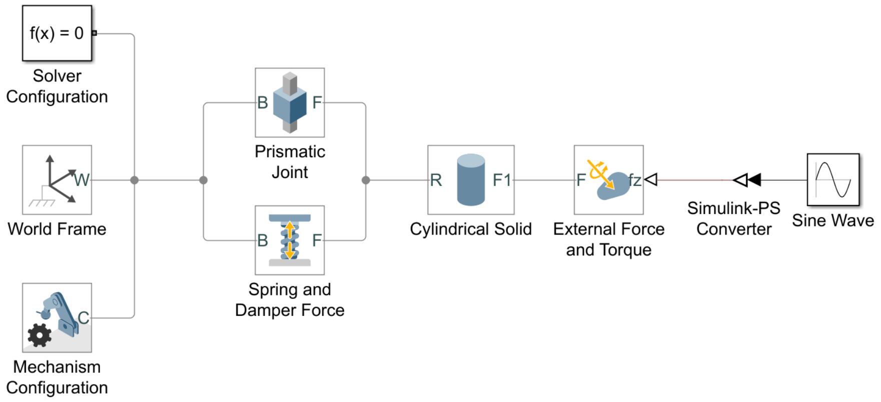


Figure 3.5: A simple mechanical models of an oscillator. A Cylindrical Solid can move vertically due to the Prismatic Joint (the $z$ axis of the World Frame and of $R$ are aligned). On the frame $F1$, on the top surface of the cylinder, is acting a vertical harmonic external force generated with a Sine Wave and converted into a physical signal by the Simulink-PS Converter block. The motion of the body is damped by the Spring and Damper Force connected in parallel with the joint.

(and kinematics).

A simulation can be run until a specified *Stop Time* (and, if needed, *from* a certain *Start Time*), or indefinitely by setting the *Stop Time* parameter to *inf*.

## 3.5. Input and Outputs of the Model

It is clear that the position and the force of joints can be easily imposed or read by using the *Actuation* and *Sensing* sections of the joint block. To measure the position and the force of a body (or rather, of one of its frames) a simple workaround can be implemented. A 6-DOF Joint block can be used to measure or impose the relative position, orientation and forces between the base and the follower frames. Therefore using a 6-DOF Joint to connect a body frame to the World Frame allows one to actuate the body and measure its position and forces. Of course, this joint does not generate any constraints (hence the name), therefore can be added anywhere in the model without compromising the physical behaviour. Figure 3.6 shows an example of this usage.

### 3.5.1. Input

*Simulink* offers many built-in signal sources, such as Constant, Ramp, Step, Sine Wave, waveforms (with the blocks Waveform Generator or Signal Generator), and random se-

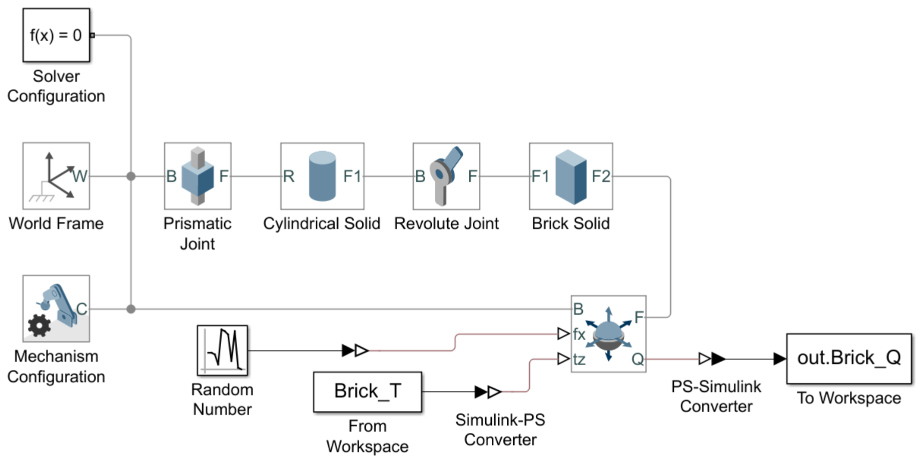


Figure 3.6: A simple mechanical model using the 6-DOF Joint block to impose the a force and a torque on a frame of the Brick Solid, and measure its angular position with respect to the World Frame. The force is generated by a Random Number block, the torque is obtained by a From Workspace block, and the output is saved using To Workspace.

quences. For more complex input signal, it might be convenient to create a signal using the Signal Builder, or to import the data from the *Matlab* workspace (From Workspace block), from a *Matlab* .mat file (From File), or from a spreadsheet (From Spreadsheet).

The From Workspace and From File blocks can import signals represented in different ways: a so-called *Matlab* timeseries format, a matrix and a struct. The easiest and most versatile is to format the input in a 2D matrix. The first column must contain the timestamp, and the other column must contain a signal each [19]. Thus every row has a timestamp $t_i$ and a vector $(a_i, b_i, c_i, \dots)$:

$$\begin{array}{ccccc} t_1 & a_1 & b_1 & c_1 & \dots \\ t_2 & a_2 & b_2 & c_2 & \\ & & \vdots & & \\ t_n & a_n & b_n & c_n & \end{array}$$

*Simulink* can interpolate linearly the values of the vector between timestamps. If the simulation time is longer than the last timestamp ($t_n$ in the above example), the signal can be extrapolated linearly, set to zero, held to the final value, or repeated cyclically.

## 3.5.2. Output

The output of a multibody simulation consists typically of position and force values of one or more joints. For instance, when simulating a robot with inverse dynamics, the desired output might be the positions and torques history of the motor joints.

A signal history can be saved using the To File or To Workspace *Simulink* blocks. The output format is analogous to the input format: a timeseries, a matrix or a struct. When using the To Workspace block, the outputs are saved into the *Matlab* workspace, or in the workspace of the callee if the simulation is run programmatically using the `sim` command.

# 4 | Generation of the Model

This Chapter presents how a 3D model of the M-1$i$A/0.5A was used to create a *Simscape Multibody* model. The setup of the *Simscape Multibody* model is paramount to correctly simulate the robot. The robot can be actuated in direct and inverse kinematics and dynamics thanks to the use of the 6-DOF Joint, and can be conveniently controlled from *Matlab*.

The main goal of this thesis is to simulate some of the working characteristics of the M-1$i$A/0.5A using the resulting model in *Matlab*. The main characteristics pursued are:

**workspace** definition, verification of theoretical and usable workspace, envelope;

**kinematics** computation (direct and inverse);

**force and torques** on individual elements of the kinematic chain.

## 4.1. Simplified Model and Full Model

This work mainly deals with the Wrist kept at constant attitude. For this reason it was chosen to simplify the model by inhibiting the three dof associated with the rotation. Such model retains the three dofs J1, J2 and J3 that control the translation of the end-effector, while the joints J4, J5 and J6 are fixed in their zero position.

## 4.2. CAD Drawing

The Computed Aided Design (CAD) drawing of the M-1$i$A/0.5A was created by Ing. Michal Bartoš, a doctoral student at the University of Žilina. The drawing is an assembly made using the CAD software *PTC Creo Parametric*. It is a mechanism, meaning that it includes the information about the connection and the relative motion between all the parts that compose the kinematic chain.

**Material densities**

| Material | | Density [kg/m³] |
|---|---|---|
| *Steel* | *Medium carbon* | 7850 |
| *Aluminium* | *Al-Si-Cu alloy* | 2680 |

Table 4.1: Density of the material used to model the M-1*i*A/0.5A parts.

### 4.2.1. Export Pre-processing

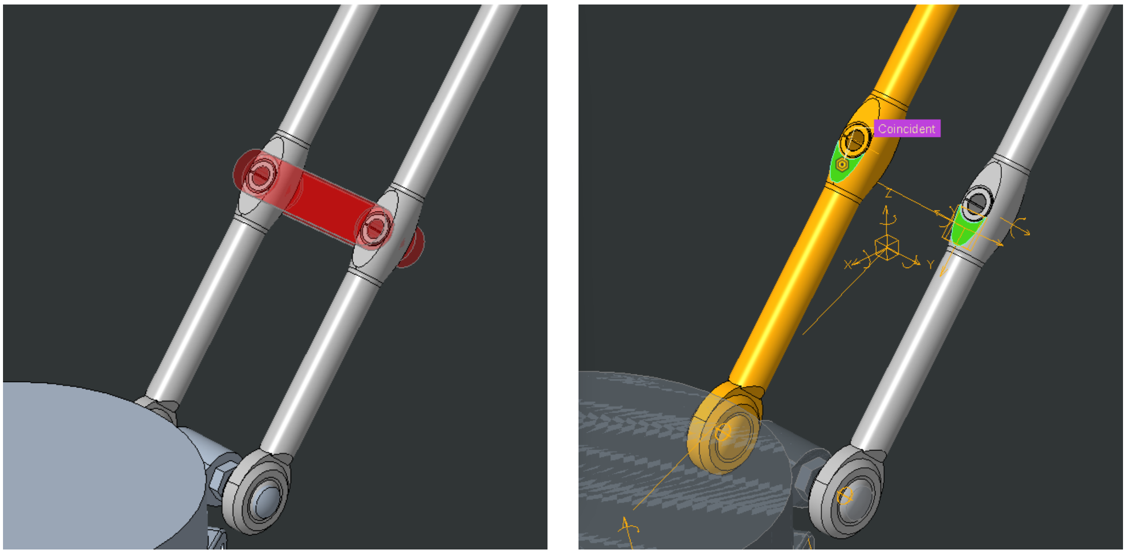
The *Simulink* model can be created from the CAD drawing using the *Simscape Multibody Link* plug-in for *PTC Creo Parametric* [11]. It allows to export the robot structure from a CAD assembly drawing to a *Simscape Multibody* model. This process preserves the robot mechanism by substituting assembly constraints with equivalent joints in *Simscape Multibody*.

Before exporting the CAD drawing to *Simulink*, some pre-processing operations were carried out. In particular: the units and the material were checked, the position of the origin was updated to match the manufacturer's one, some parts were removed to simplify the model, and some others were joined into a single object.

The six Rods are made of aluminium, while all the other relevant components are made of steel. The moving platform contains a gearbox to actuate the three dof J4, J5 and J6 of the Wrist. To account for the internal cavity, it was supposed that 50% of its volume is empty. The values used for the densities are listed in Table 4.1.

The components removed were the twelve (four for each of the three parallelograms) plates joining the Rods in pairs to form four-bar mechanisms. They have only a mechanical support role, and are irrelevant from a kinematic point of view. But since all the Rods are constrained at both ends only by spherical connections, it is now necessary to prevent the rotation along their longitudinal axis. The solution adopted was to include one fictitious plane coincidence constraints between the Rods of each pair, as shown in Figure 4.1.

Finally, all the grounded and immobile parts where joined together to ease the export and prevent unnecessary clutter in the *Simulink* model. These consist mainly in the robot case and support structure, and the shafts on which the Arm 3 hinge. The aforementioned parts were removed from the assembly and substituted by the new single part. The assembly constraints lost were recreated identical to the original ones. The final simplified assembly is shown in Figure 4.2.



(a) Original four-bar mechanism with two connecting Links (in red).

(b) Coincidence constraint introduced in *PTC Creo Parametric* to prevent free axial rotation of the Rods.

Figure 4.1: Link of four-bar mechanism replaced with planar constraint.

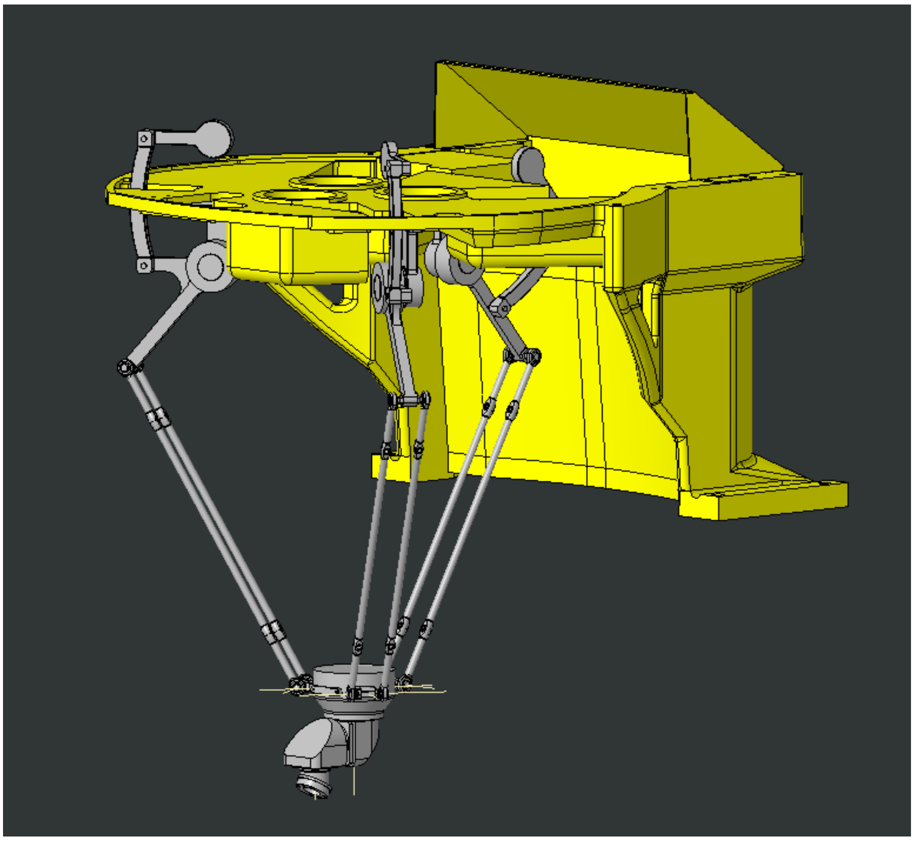


Figure 4.2: Simplified assembly in *PTC Creo Parametric*. The Links are removed (cf. Figure 2.2). The top part of the Cover is hidden. The robot is shown in the zero position.

### 4.2.2. Export to Simscape

The *Simscape Multibody Link* plug-in for *PTC Creo Parametric* stores the data required to build the *Simulink* model in an .xml file containing information regarding the parts, their masses and volumes, and the position and type of joints connecting them. Furthermore, every part of the assembly is converted into STEP format in order to visualize the model in the *Mechanics Explorer* (3.3).

The geometry in the .step files are only used only for visual representation and animation, and the computations do not rely on them. To ease the burden of the graphical work, all the drawing files have been modified with *SolidWorks* to remove unnecessary features (like chamfers), and reduce the number of points of the geometry, while keeping an acceptable level of details.

Finally, the *Simscape Multibody* model was generated in *Matlab* with the command `smimport`. This produced a *Simulink* model .slx file resembling the mechanism in the CAD drawing. Some of the constants used in the blocks of the models are collected in a *Matlab* script executed before every simulation.

## 4.3. Simscape Multibody Model

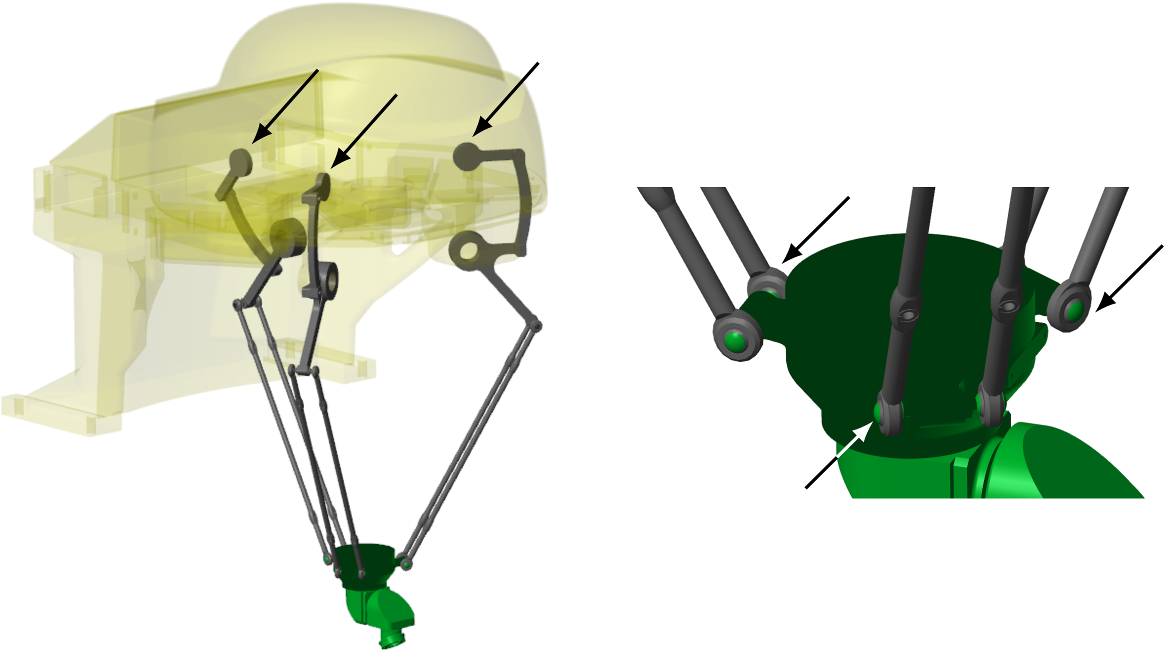
The blocks of the *Simscape Multibody* model and their connection were thoroughly checked. Particular attention was given to the masses of the bodies and the transforms blocks. The gravity force was disabled as it was not required in the simulations.

### 4.3.1. Import Post-processing

Some extra Rigid Transform blocks were added in order to move and rotate some frames. This allowed to measure angles between parts more easily. Furthermore, in this way the angular displacement and the zero configuration correspond in the *Simulink*, model, in the CAD mechanism and in the physical robot.

Some frames were also added to reference interesting positions, such as the end-effector and the central axis of the workspace.

To guide the assembly of the model to the only desired position amongst the many kinematically valid, the initial condition of the some joint parameters were set by specifying the *State Target* position and priority for some the joint blocks (3.2.3) [10, 12]. In particular, the three motor angles were set to 0 deg with high priority, and three of the bottom spherical joints (one per kinematic chain) were set to their value at import with



(a) Whole robot in default pose.

(b) Close up of the spherical joints that connect the end-effector to the parallelogram Rods.

Figure 4.3: View in the *Mechanics Explorer* of the robot in the default configuration. The joints whose initial position was set are indicated by arrows.

low priority (Figure 4.3). The zero pose is shown in Figure 4.2.

### 4.3.2. Comparison of Simscape Model with CAD Mechanism

Figure 4.4 shows a comparison of the newly built *Simscape Multibody* model and the CAD assembly.

### 4.3.3. Measuring and Actuation

Some of the most important parameters of this analysis are the motor angles and torques. For this reason the Revolute Joint blocks representing the motors were set to output the angular position and the torque. This data was converted and collected by a To Workspace block (3.5), as visible in Figure 4.5.

In order to measure and actuate of the end-effector, a 6-DOF Joint block was added between the World Frame and the end-effector (3.5), as shown in Figure 4.6.

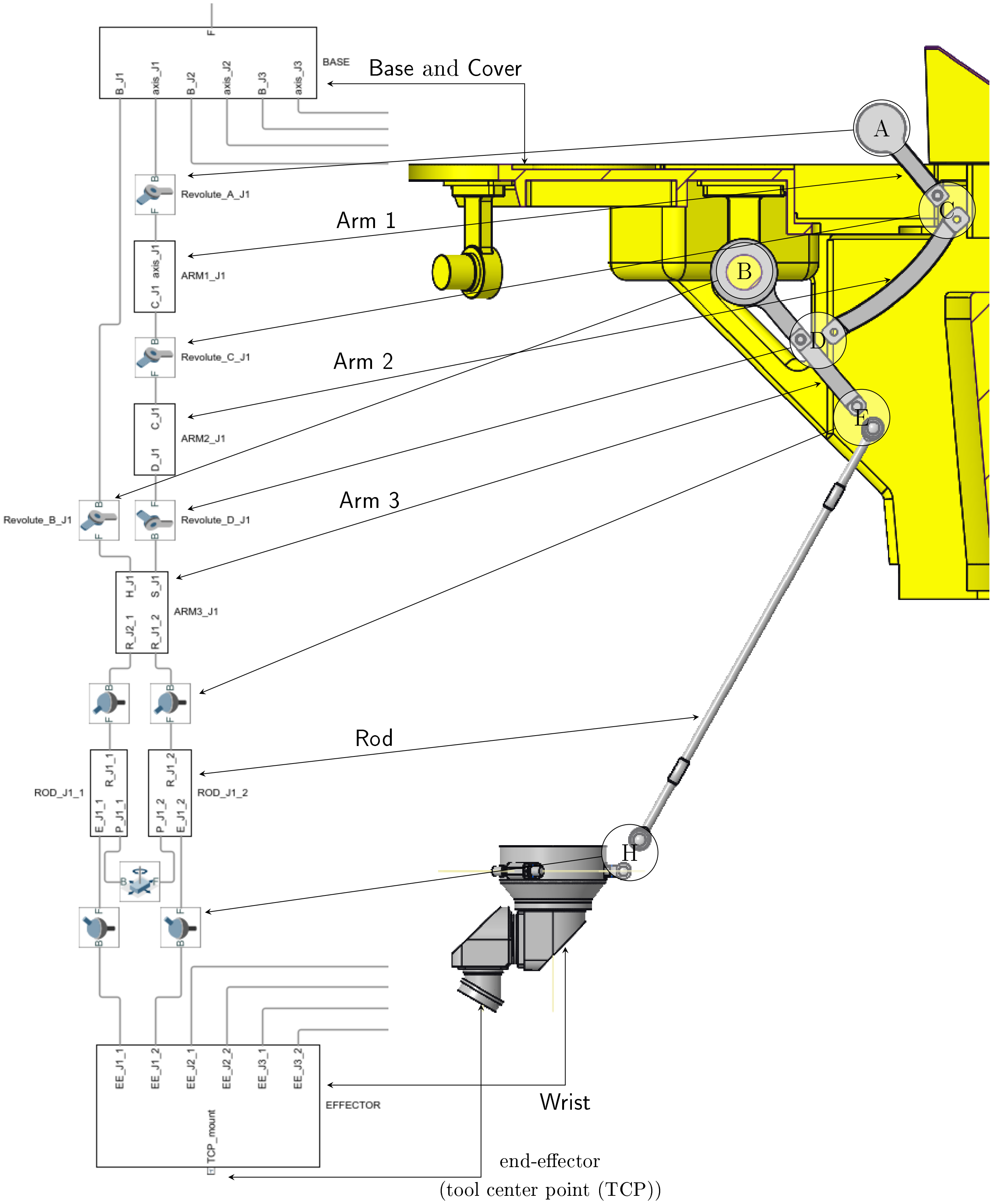


Figure 4.4: Comparison Leg 1 in *Simulink* model (left) and CAD model (exploded view) (right). The port F of the Base block is connected to the World Frame.

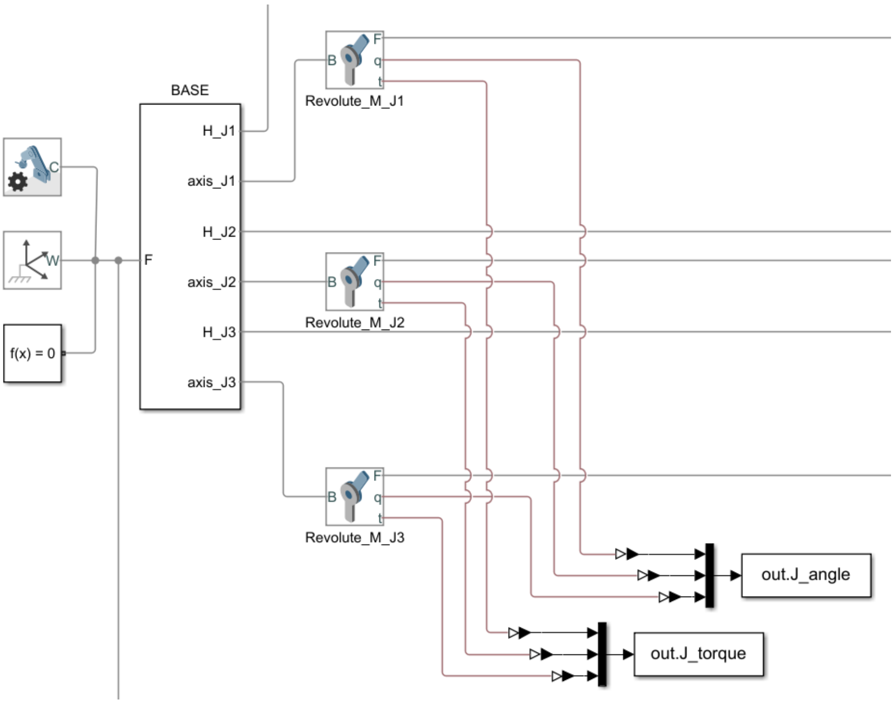


Figure 4.5: To Workspace blocks collecting torques and positions of the motors.

## 4.4. Control of the model

The model can be controlled by prescribing a motion to the joint blocks, according to their specific constraints. To actuate a *Simscape Multibody* joint, the *Actuation Motion* and *Actuation Torque* (or *Force*) settings must be set to *Provided by Input* or *Automatically Computed*, depending on the desired type of kinematic and dynamic motion (3.2.3)

In this model the joints representing the motors and the 6-DOF Joint connecting the end-effector to the World Frame should be actuated. All the other non-actuated joints are passive: the *Actuation Torque* is *None*, and the *Actuation Motion* is *Automatically Computed*.

### 4.4.1. Control of the Model from Matlab

The input variables required by the From Workspace blocks of the *Simulink* model can be created in the *Matlab Workspace*, and the simulation can be run from the *Simulink* interface or with the command `sim` from the *Command Window*, a script or a function (3.5).

Most of the simulations required inverse kinematics analysis. Therefore the motion law of the end-effector had to be saved into the *Workspace* in matrix form prior to the simulation.

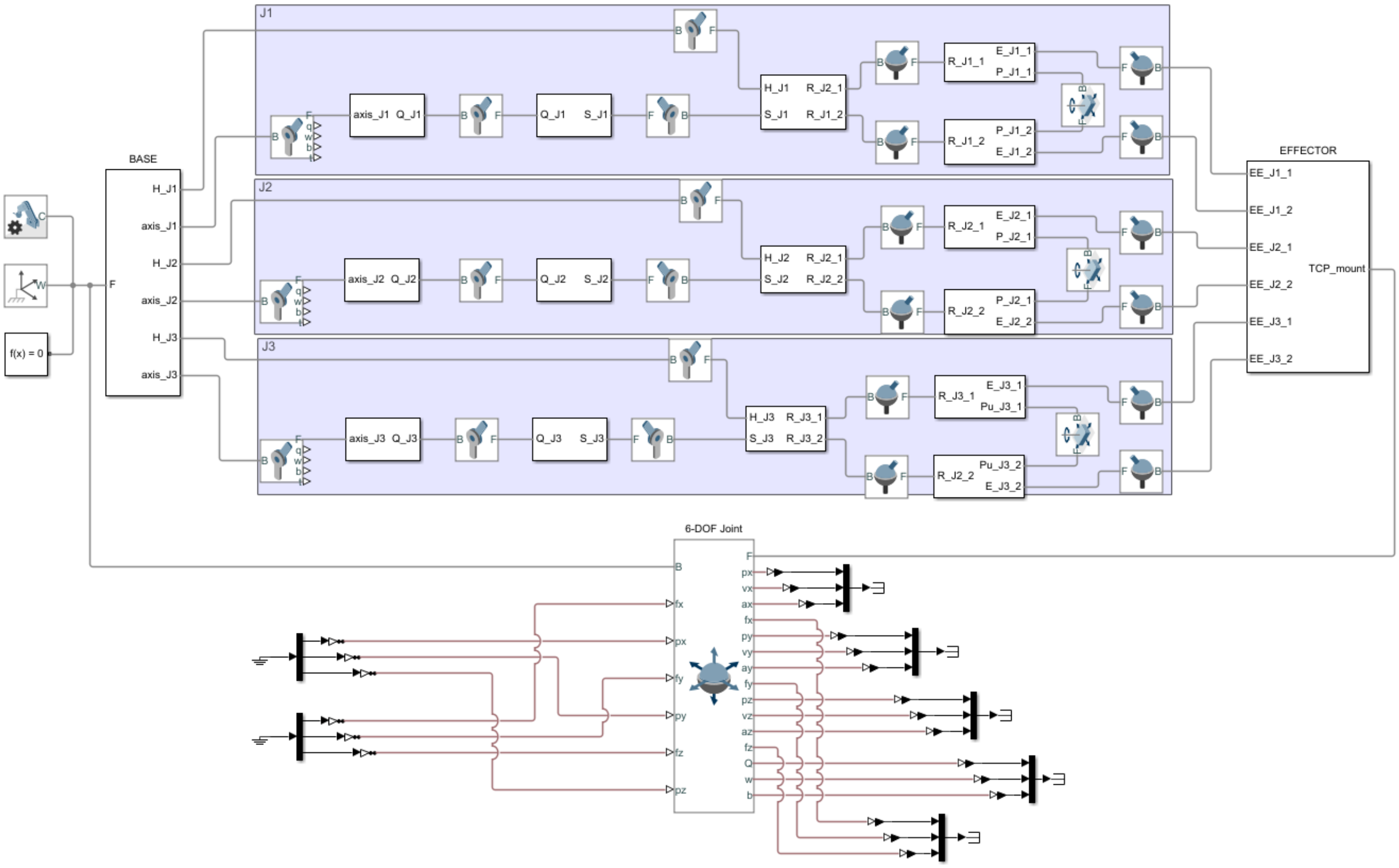


Figure 4.6: On the top part are visible the three kinematic chains of the Legs connecting the Base to the rigid Wrist (effector). On the bottom, the 6-DOF Joint connects the World Frame and the end-effector; actuation inputs are connected on its left, and sensor outputs on the right.

The simulation outputs generated from the To Workspace blocks could be read after the job, along with the vector of discrete time instants at which the model was evaluated.

# 5 | Workspace

In this and the following Chapters the results of the analysis on the simplified model will be reported. Despite not representing the real robot, this information is still useful for some scenarios of operation, for instance when the attitude is kept constant or if the task requires only translations. Moreover, similar methods could be used for other product of the M-1$i$A series, like the three dof *M-1iA/1H* (see Table 2.1).

The workspace is a very important characteristic for industrial robot. Since the end-effector is bounded to be inside it, it is of primary importance to identify the workspace of the M-1$i$A/0.5A. The results shown here are obtained with the Wrist in the zero position. The same still holds valid for other Wrist orientations: the resulting workspaces will be equivalent but translated.

## 5.1. Preliminary Assumptions on the Workspace

Due to the symmetry of the M-1$i$A/0.5A, it is reasonable to assume that the workspace also has some type of symmetry (radial (also called rotational) or axial) about a vertical axis. Given the dimensional equivalence of the three links (as discussed in 2.1), there is not a preferential direction where the end-effector could move farther, thus the workspace axis is assumed coincident with the end-effector endpoint in the zero configuration; namely, passing through the point $(x = 42.6, y = 0)$ mm.

## 5.2. Procedure for Evaluation of the Workspace

The core idea of the test was to define a blatantly larger *potential workspace*, and find which of its points can be actually reached. These shall be used to build the robot workspace.

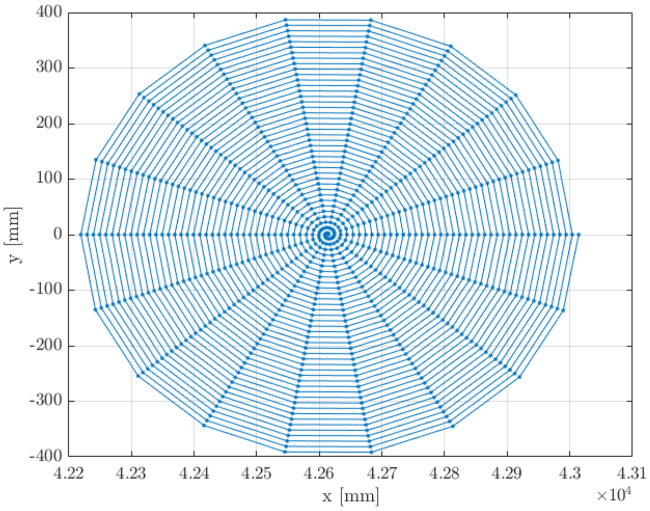


Figure 5.1: Spiral motion prescribed to the end-effector for a generic layer $z = z_i$.

### 5.2.1. First Potential Workspace

The potential workspace initially considered was a cylinder with radius of 400 mm and height of 300 mm, from $z = -500$ mm to $z = -200$ mm (cf. Figure 2.4). It was sliced into 60 layers every 5 mm. At each layer a simulation was run, prescribing a spiral motion to the end-effector (or rather, a straight-segments approximation of a spiral). The spiral centre was coincident with the workspace axis ($(x = 42.6, y = 0)$ mm), and the distance between consecutive loops was 10 mm. The prescribed motion for a generic layer is depicted in Figure 5.1. The so-built workspace will have a vertical resolution of 5 mm and a radial resolution of 10 mm.

To prevent assembly errors, all the simulations started from the zero position of the robot, then the end-effector was moved to the desired $z$ plane, and ultimately the spiral motion could begin.

The many simulations were carried out in parallel using `parsim`. A simulation would stop if any of the following condition would arise:

- the model cannot be assembled, because the pose is physically unreachable;
- any of the arms collide another part.

The first condition is verified automatically by the *Simscape Multibody* assembler. The second condition arises whenever a motor angle (J1, J2 or J3) is outside the range $]-66, +72[$ deg, therefore can be enforced by using a Stop Simulation block. Figure 5.2 shows how the stop logic was implemented. The aforementioned range was chosen by

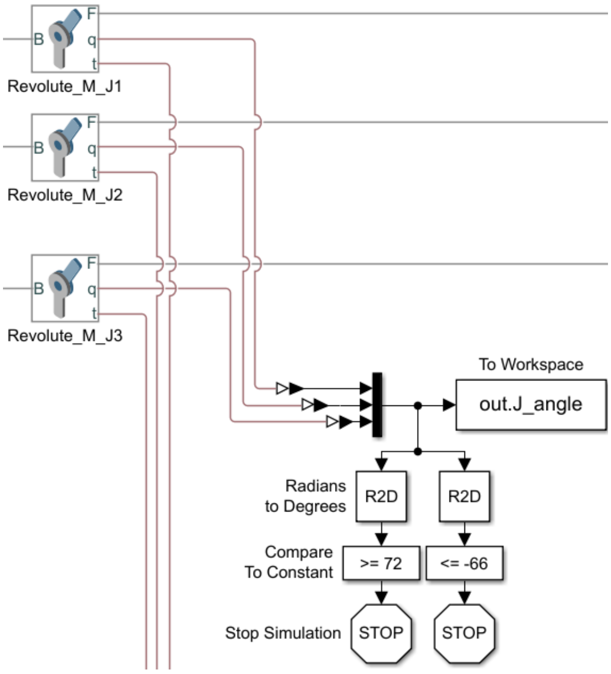


Figure 5.2: The motor angles in output from the Revolute Joint corresponding to the J1, J2 and J3 motors are compared to hardcoded boundaries. If at least one is out of range, the simulation halts.

rounding towards zero (truncating) the maximum angles at which Arm 1 can move. Those angles were measured in *PTC Creo Parametric* by placing the three Arm 1 in contact with the frame, and are the same for J1, J2 and J3.

### 5.2.2. First Result

To account for singularities, a force approach was used. It is known that in proximity of a singular position the motion is limited and the torques can reach very high values (1.1.8). Therefore also in the *Simscape Multibody* model, when the motion approaches a singularity, the motor torques are extremely large. To be able to detect these points, it was necessary to have non-zero static torques on all three motors, therefore a load of 10 N in $x$, $y$ and $z$ direction was applied to the end-effector. This resulted in motor torques within 5 Nm (in absolute value) in most of the tested points. The points were any of the motors experienced a torque greater than 10 Nm (in absolute value) were considered singular points, and were excluded from the workspace. The remainder of the points are considered *legal*. Finally, a *preliminary workspace* was built using only the legal points.

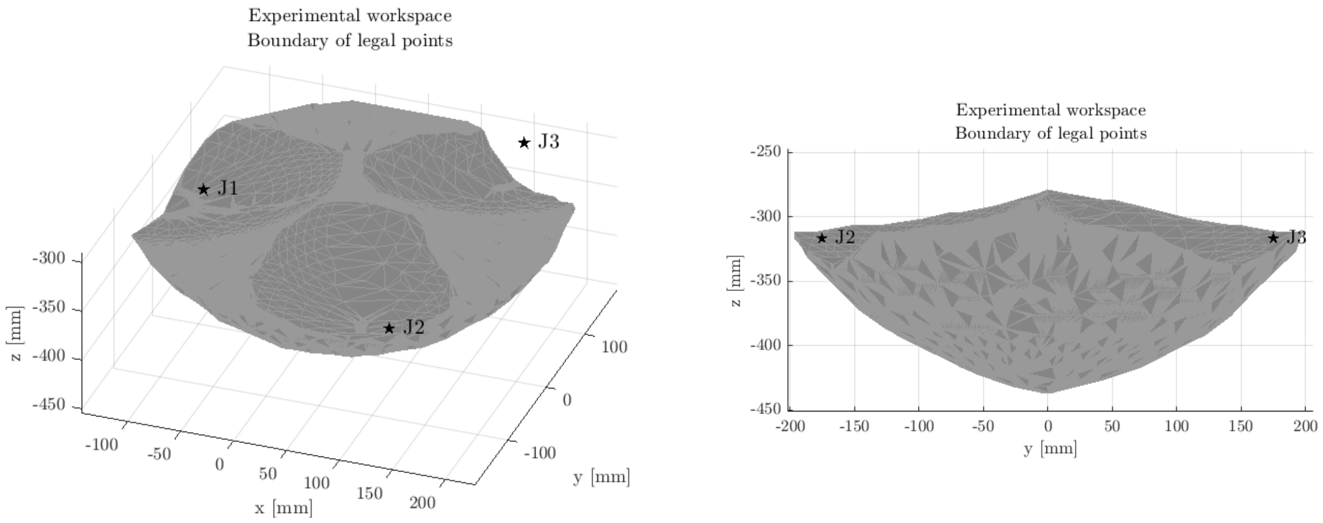


Figure 5.3: Tridimensional (left) and Frontal (right) views of workspace obtained by all legal points.

### 5.2.3. Second Potential Workspace

To have greater spatial resolution, a second batch of simulations was run. The vertical layers were at a distance of 1 mm, and the distance between consecutive spiral loop was also 1 mm. This time the potential workspace was a smaller hollow cylinder: the height was reduced to include just the preliminary workspace plus 5 mm on top and bottom (the vertical resolution of the preliminary workspace), and the minimum radius of the spirals was at the closest singular point minus 10 mm (the radial resolution).

## 5.3. Results

Similarly as done before, the singular points were eliminated considering where the torque in any of the motor was greater than 10 Nm. The resulting workspace is bowl-shaped with three hemispherical carving on the top. Figure 5.3 shows the boundary of the points of the workspace.

The largest motion range in the $xy$ plane is at $z = -312$ mm, where the maximum distance from the workspace axis is 202 mm, but some point in this plane are unreachable. The largest circle constituted of only legal points is at $z = -342$ mm, and has a diameter of 177 mm. In the layers shown in Figure 5.4 is clearly visible the third-order rotational symmetry about the vertical axis passing through the end-effector at zero position. The workspace distortions are placed at 120 deg, and coincide with the position of the joints.

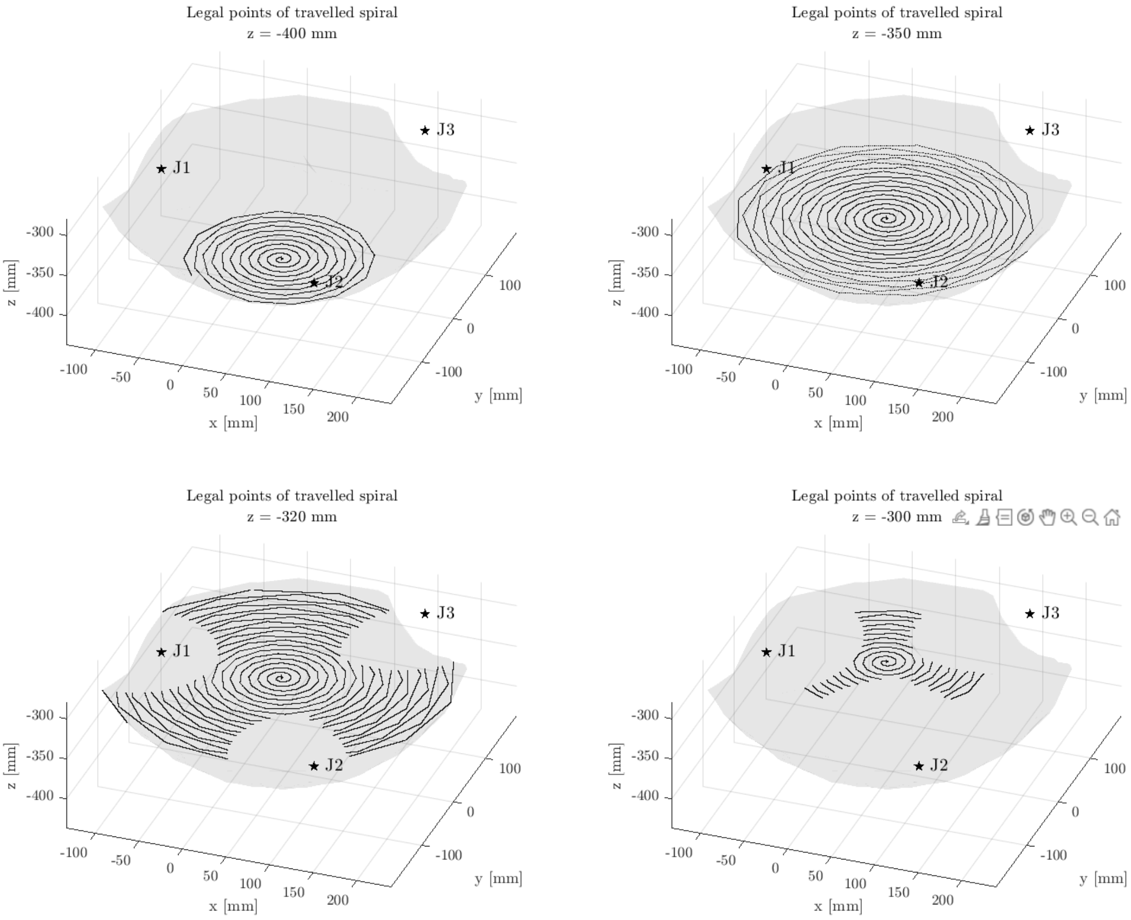


Figure 5.4: Legal points on the spirals travelled by the end-effector in different $z$ planes.

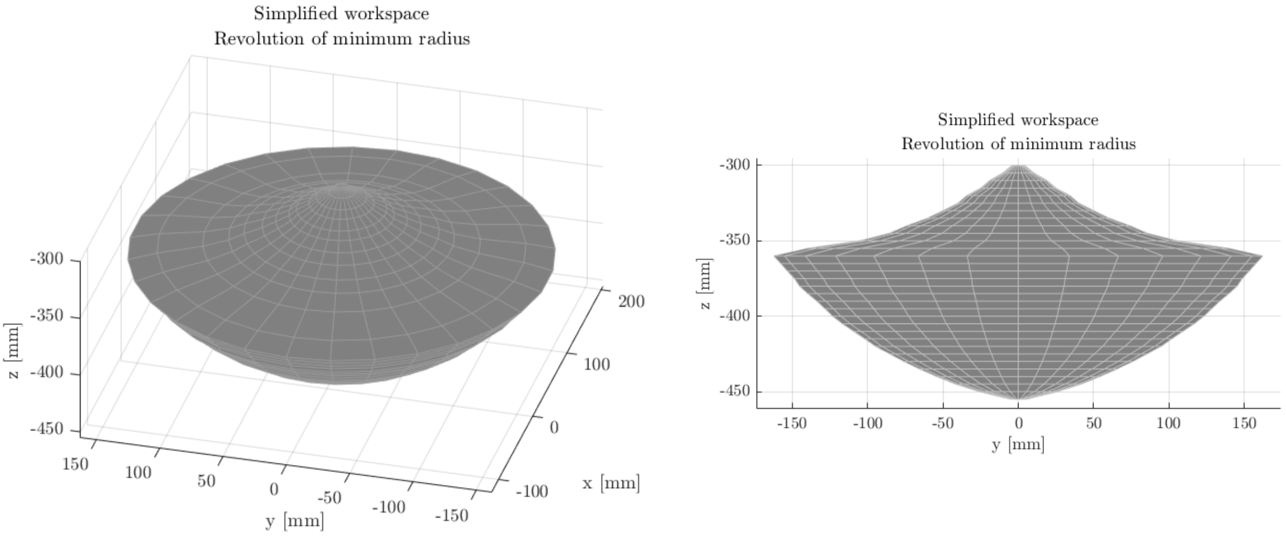


Figure 5.5: Tridimensional (left) and Frontal (right) views of the simplified workspace obtained by revolution of the minimum legal radius.

Nonetheless, most of the workspace has axial symmetry, therefore its definition can be greatly simplified by discarding some points and expressing the radius as a function of $z$.

## 5.4. Geometric Simplification of the Workspace

For simplicity's sake the workspace can be approximated to an axisymmetric volume obtained by revolution of the curve $R(z)$ representing the distance from the revolution axis. At each layer $z_i$ the radius $R(z_i)$ must be the largest containing *only* legal points. The curve $R(z)$ is shown in Figure 5.6. In this way the radial symmetry is lost (because there might be farther reachable points), and of course the volume of the workspace is reduced. Figure 5.5 shows the approximated axisymmetric workspace obtained by revolution.

An even simpler geometry is similar to those of the Delta robots in Figure 1.6, and with the nominal workspace in Figure 2.4. It consists in a cylinder with two different cone trunks on its bases. This geometry can be obtained by revolution of the curve shown in Figure 5.6. The final workspace volume is shown in Figure 5.7.

The latter simplified workspace will be used for all the forthcoming work.

## 5.5. Comparison with the 6dof Nominal Workspace

The nominal workspace of the M-1$i$A/0.5A was obtained from the *Fanuc Roboguide* software. Figure 5.8 shows it along with the simplified workspace computed for the rigid wrist

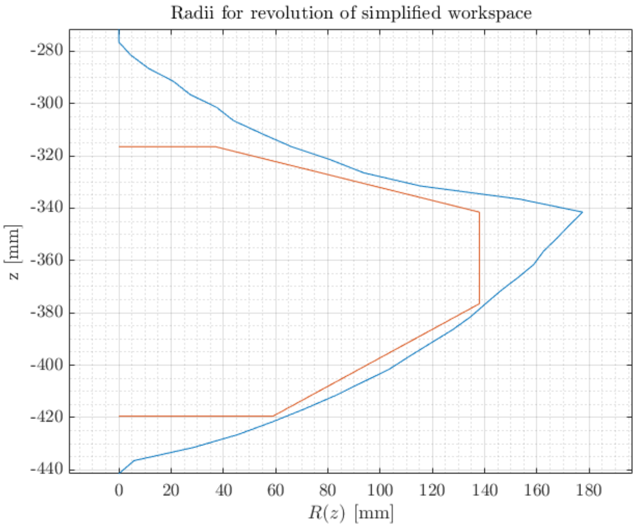


Figure 5.6: Radii used for the simplified workspaces. Blue: first approximation. Orange: second approximation, using straight lines.

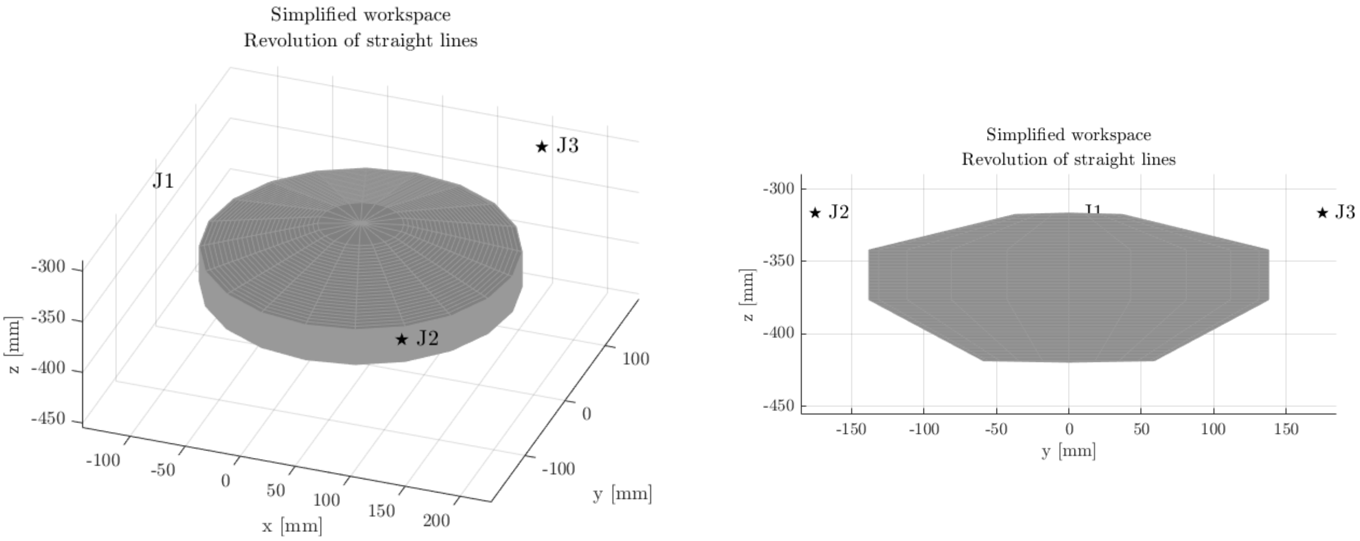


Figure 5.7: Tridimensional (left) and Frontal (right) views of the second simplified workspace obtained by revolution of straight lines in Figure 5.6.

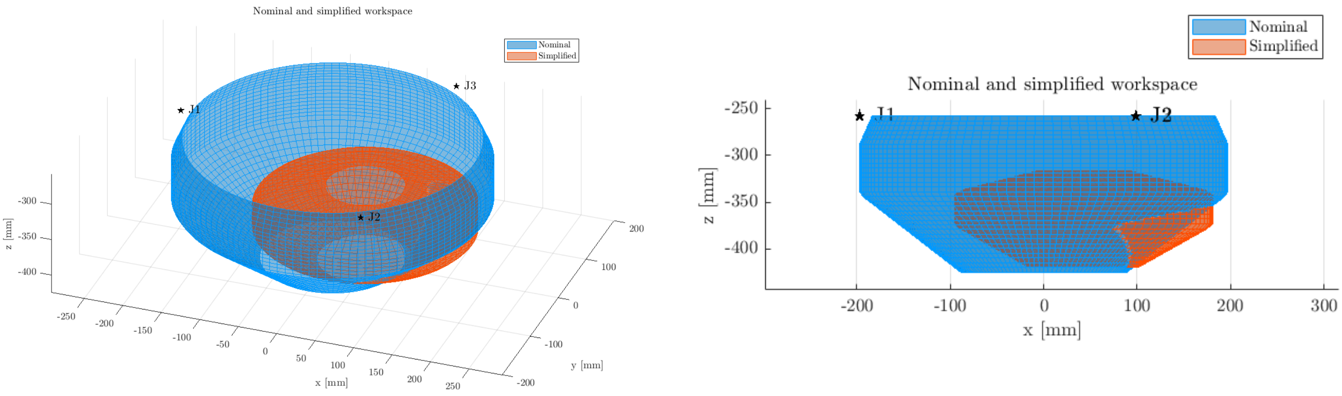


Figure 5.8: Tridimensional (left) and Side view (right) of the nominal workspace of the M-1$i$A/0.5A (blue) and the simplified workspace for the rigid wrist simplified model (red).

model. The firmer is much larger than the latter due to the greater range of mobility granted by the rotation of the Wrist. Nonetheless, the latter develops partly outside of the nominal workspace. This might be simply due to the fact that the robot manufacturer declare a nominal workspace smaller than that considered legal in this analysis. Finally, as expected the workspaces are not coaxial: the small workspace is centred in the end-effector centre in the zero position, while the large workspace is centred in the Wrist center in the zero position.

# 6 | Direct and Inverse Kinematics

Once a workspace for the rigid Wrist model was available, it was possible to undertake some analysis of the behaviour of the robot. The inverse kinematics and the differential inverse kinematics are of paramount importance for the use of any robot, as long as the direct problems. Therefore the model was used first to derive the relation between points in the workspace and the corresponding positions in the joint space; then to compute the Jacobian. The Jacobian was used to solve the differential kinematics problems, to find singularities and to compute the manipulability indexes.

## 6.1. Simulation

The end-effector was moved to $n = 17\,981$ points forming a grid within the workspace. The distance between two consecutive points in the $z$ direction was 4 mm, while the resolution in the $x$ and $y$ direction was 8 mm. These values were chosen in order to divide the workspace into approximately 25 to 35 slices in each direction. A simulation with a finer resolution would have been extremely time consuming.

These points were collected in a $n \times 4$ matrix (one column for the time, three for the position) used by the From Workspace block to control position of the end-effector. Figure 6.1 shows the workspace and the corresponding points in the joint space.

## 6.2. Direct and Inverse Kinematics Problems

The output of the simulation consisted of two matrices with the actual position of the end-effector and the corresponding configuration, namely the angles of the motor joints. These two matrices can be used to solve both the direct and inverse kinematics problem by using them as lookup tables. In particular, letting $E$ and $Q$ be the working space and

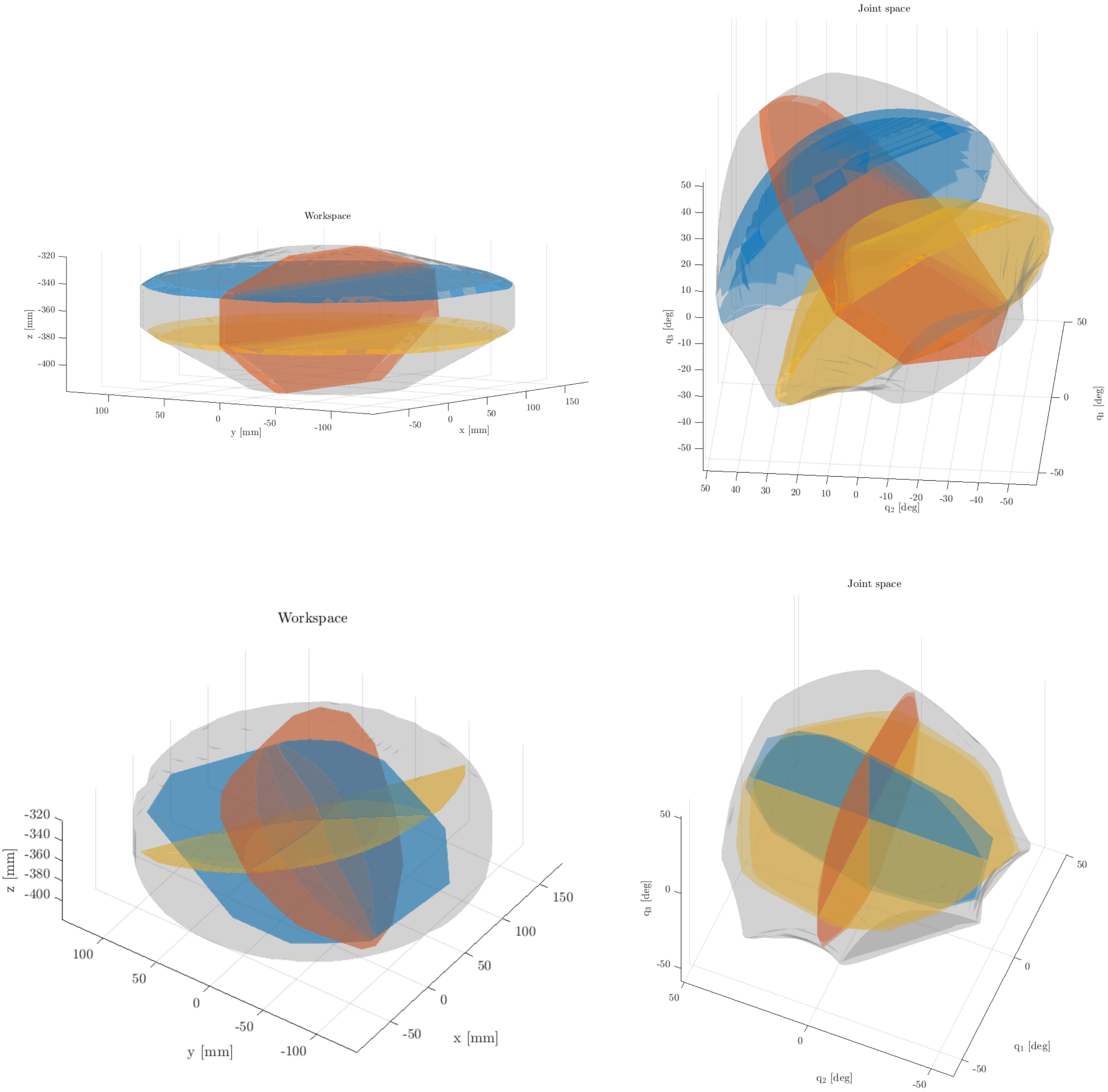


Figure 6.1: Workspace (left) and Joint space (right). Three planes are highlighted: on the top images they correspond to the end-effector in the planes: $z = -340$ mm (blue), $z = -380$ mm (orange) and $y = 0$ mm (red). On the bottom images the planes corresponds to the zero position of J1 (blue), J2 (red) and J3 (orange).

the joint space matrices[1], the *Matlab* command[2]

```
interp3(E(:,1), E(:,2), E(:,3), Q(:,k), x, y, z)
```

returns the motor angle of joint $k$ to place the end-effector in the point $(x, y, z)$, thus solving the inverse problem corresponding to Equation 1.6. While

```
interp3(Q(:,1), Q(:,2), Q(:,3), E(:,k), q1, q2, q3)
```

returns the position end-effector in direction $k$ corresponding to the motors in configuration $(q_1, q_2, q_3)$, thus solving the direct kinematics problem (1.2).

## 6.3. Jacobian and Differential Kinematics

The differential inverse kinematic problem can be solved by computing the Jacobian matrix. The Jacobian is also useful for detecting singular points and evaluate the manipulability in the workspace.

### 6.3.1. Computation of Jacobian

The end-effector position was interpolated across a three-dimensional grid in the joint space, yielding three three-dimensional matrix: $E_x$, $E_y$ and $E_z$ for the position in $x$, $y$ and $z$ respectively. The grid ranged from the minimum to maximum value for each motor angle, with a step between each point of $\Delta q = 1$ deg. For all the three joints the limits were approximately $-58.6$ and $+51.2$ deg.

These matrices $E_i$ are three-dimensional because each of them represents the direct kinematic function $f_i : \mathbb{R}^3 \to \mathbb{R}$:

$$\begin{aligned} E_x(q_1, q_2, q_3) &\approx f_x(q_1, q_2, q_3) = x \\ E_y(q_1, q_2, q_3) &\approx f_y(q_1, q_2, q_3) = y \\ E_z(q_1, q_2, q_3) &\approx f_z(q_1, q_2, q_3) = z \end{aligned} \tag{6.1}$$

Subsequently, the gradient for each of the three $E_i$ matrices was computed, yielding nine three-dimensional matrices, corresponding to the nine elements of the Jacobian matrix

[1] $E$ and $Q$ have three columns. Every row of $E$ is a point $(x, y, z)$ in the working space, while every row of $Q$ is a point $(q_1, q_2, q_3)$ in the joint space. The command `E(:,1)` selects the first column of $E$, thus the $x$ coordinate of all points in the working space.

[2] Other commands such as `scatteredInterpolant` and `griddata` can also be used, and should be preferred if the query points are matrices.

function, where $J_{ij} : \mathbb{R}^3 \rightarrow \mathbb{R}$:

$$\frac{\Delta E_i(q_1, q_2, q_3)}{\Delta q_j} \approx J_{ij}(q_1, q_2, q_3) = \frac{\partial f_i(q_1, q_2, q_3)}{\partial q_j} \tag{6.2}$$

Finally the Jacobian elements were reshaped in the canonical form:

$$J(q_1, q_2, q_3) = \begin{pmatrix} J_{x1}(q_1, q_2, q_2) & J_{x2}(\dots) & J_{x3}(\dots) \\ J_{y1}(\dots) & J_{y2}(\dots) & J_{y3}(\dots) \\ J_{z1}(\dots) & J_{z2}(\dots) & J_{z3}(\dots) \end{pmatrix} \tag{6.3}$$

It must be noticed that not all points of the grid in the joint space can be reached by the robot. Consequently, the unreachable points do not have a direct kinematic solution, and the corresponding value in the $E_i$ matrices is `NaN` (Not A Number). Any arithmetic operation on a `NaN` returns `NaN`, therefore the Jacobian on the border of the workspace has all or some `NaN` elements. These matrices are discarded in the subsequent computations.

### 6.3.2. Differential Kinematics Problems

To solve the forward kinematics problem, the Jacobian matrix function computed can be used as in Equation 1.4 to obtain the velocities of the end-effector:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = J(q_1, q_2, q_3) \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} \tag{6.4}$$

Since $J(q_1, q_2, q_3)$ is a $3 \times 3$ matrix, it can be easily inverted to solve the inverse kinematic problem as per Equation 1.7:

$$\begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = J^{-1}(q_1, q_2, q_3) \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} \tag{6.5}$$

$J$ can be inverted only if it is full rank and its conditioning number is low.

## 6.4. Singularities

A singularity arises where the Jacobian fails to be full rank (1.1.9). The Jacobians computed have rank equal to 3 in every point of the workspace, thus this workspace is free of

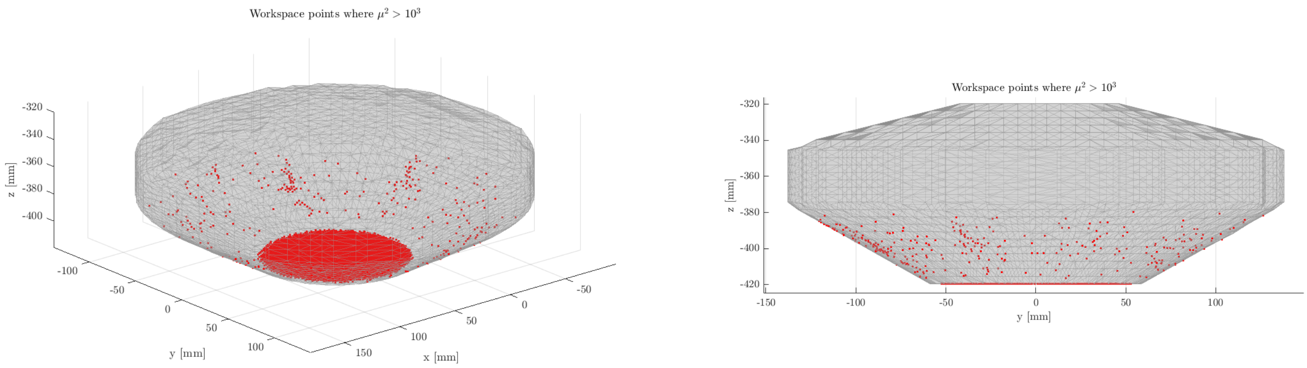


Figure 6.2: Tridimensional (left) and Frontal (right) views of the points of the workspace where the condition number of the Jacobian is greater than $10^3$.

singularities.

Despite the Jacobians are full rank in every point, they are ill-conditioned in some regions of the workspace. A very high condition number is an indicator of near-singularity of a matrix. The condition number is equivalent $\mu^2$, as defined in Equation 1.19. The points for which the Jacobian is ill-conditioned are shown in Figure 6.2. The location of ill-conditioned Jacobians suggests to increase the minimum position of the workspace and reduce the amplitude of the bottom cone.

## 6.5. Manipulability Ellipsoids

The velocity or force manipulability ellipsoid can be simply built from the Jacobian matrix as shown in 1.1.9.

Once the Jacobian matrix $J(q_1, q_2, q_3)$ was known for every point of the workspace, the eigenvalues and eigenvectors of $JJ^\mathsf{T}$ were computed. Finally, the volume index $v$ and the isotropy index $\mu$ were evaluated as per Equations 1.18 and 1.19. Before analysing the results, the outliers were removed from $v$ and $\mu$[3].

These indexes are difficult to visualize from bidimensional images. Nonetheless, the Appendix A contains some figure to show their value in the workspace. The pictures show a large volume index in most of the workspace, which quickly decreases towards the bottom of the bowl. The isotropy index changes linearly from nearly 1 on the top to a maximum of 5.58.

[3] A value was considered an outlier if it is more than three scaled median absolute deviations. This is the default method of the *Matlab* function `rmoutliers` [22].

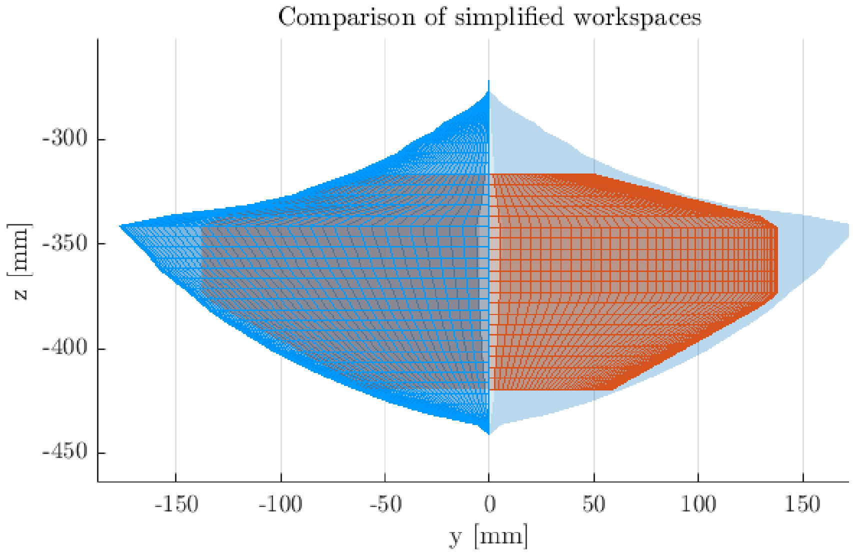


Figure 6.3: Comparison of the two proposed simplified workspace. In red: workspace obtained by revolution of the curve $R(z)$ with the minimum legal radii. In blue: workspace obtained by revolution of three straight segments.

The fact that the workspace is close to singularity in the bottom but not in the top, is due to the last simplification of the workspace operated in 5.4, which removed a lot of volume from the top. This suggests that the top cone trunk can be safely extended to increase the workspace. Figure 6.3 shows both the simplified workspaces proposed in 5.4, and depicted in the Figures 5.5 and 5.7.

As illustrated in 1.1.9, where $\mu \gg 1$ the ellipsoids are compressed in a direction in which the end-effector moves slowly but can withstand high forces. Instead, in the direction of largest elongation the end-effector can move quickly, but has poor ability to bear forces. These two directions are represented respectively by the eigenvectors of $JJ^\mathsf{T}$ with lowest and highest associated eigenvalue. Figure 6.4 shows the aforementioned directions for some ellipsoids in an horizontal plane close to the bottom of the workspace. It is clear that the end-effector can easily move horizontally, but struggles to move vertically, especially in the outer border, where $\mu$ is largest.

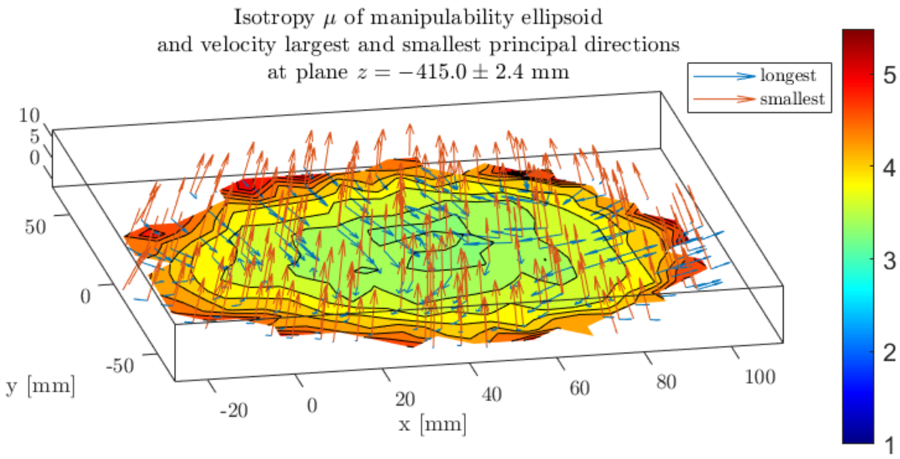


Figure 6.4: Contour of ellipsoid isotropicity, and the longest and shortest directions of the manipulability ellipsoids when the end-effector lies in the plane $z = -415$ mm.

# 7 | Static Forces

This analysis concerns how the forces acting on the end-effector are distributed across the motors loads, and what is the load on the links. Since this work eschews any particular trajectory planning, the investigation will regard only static forces, thus without inertia due to accelerations.

When using a robot, the forces on the end-effector may have components in any of the three directions. Therefore three simulations were run, in each a unitary force was exerted on the end-effector in only one of the three directions.

Due to the well-known symmetry of the workspace (as discussed in 5.3), a cylindrical coordinate system was used. The three directions were: radial ($r$), tangential ($t$) and vertical ($z$), with the cylinder axis ($r = 0$) corresponding to the workspace axis.

The unitary forces were chosen under the assumption that the torque is a linear combination of the forces when the end-effector dwells in a certain pose. For example if the force $\mathbf{F_1}$ applied in a certain point causes a torque $\mathbf{T_1}$, and the force $\mathbf{F_2}$ in the same point causes $\mathbf{T_2}$, then $\alpha\,\mathbf{F_1} + \beta\,\mathbf{F_2}$ yields a response torque $\alpha\,\mathbf{T_1} + \beta\,\mathbf{T_2}$.

The motor torques are computed by *Simulink* using the inverse dynamic model of the robot. Analytical formulas for structural forces were computed with a simple static equilibrium balance, and were implemented in the *Simulink* model using appropriate blocks.

## 7.1. Static Equilibrium Analysis

Figure 7.1 shows an exploded view of Leg 2, identical to Leg 3. On the Figure are sketched the force vectors acting on the joints in the plane where the Arms lye. The other forces due to non-coplanarity of the Rods are supported by the reactions in A and B.

$M$ is the torque on the motor in response to the loads on the end-effector, and the only unknowns are the compression $F$ of Arm 2, and the compression $G$ of the Rod. They can be obtained by a simple static equilibrium equation of Arm 1 and Arm 3:

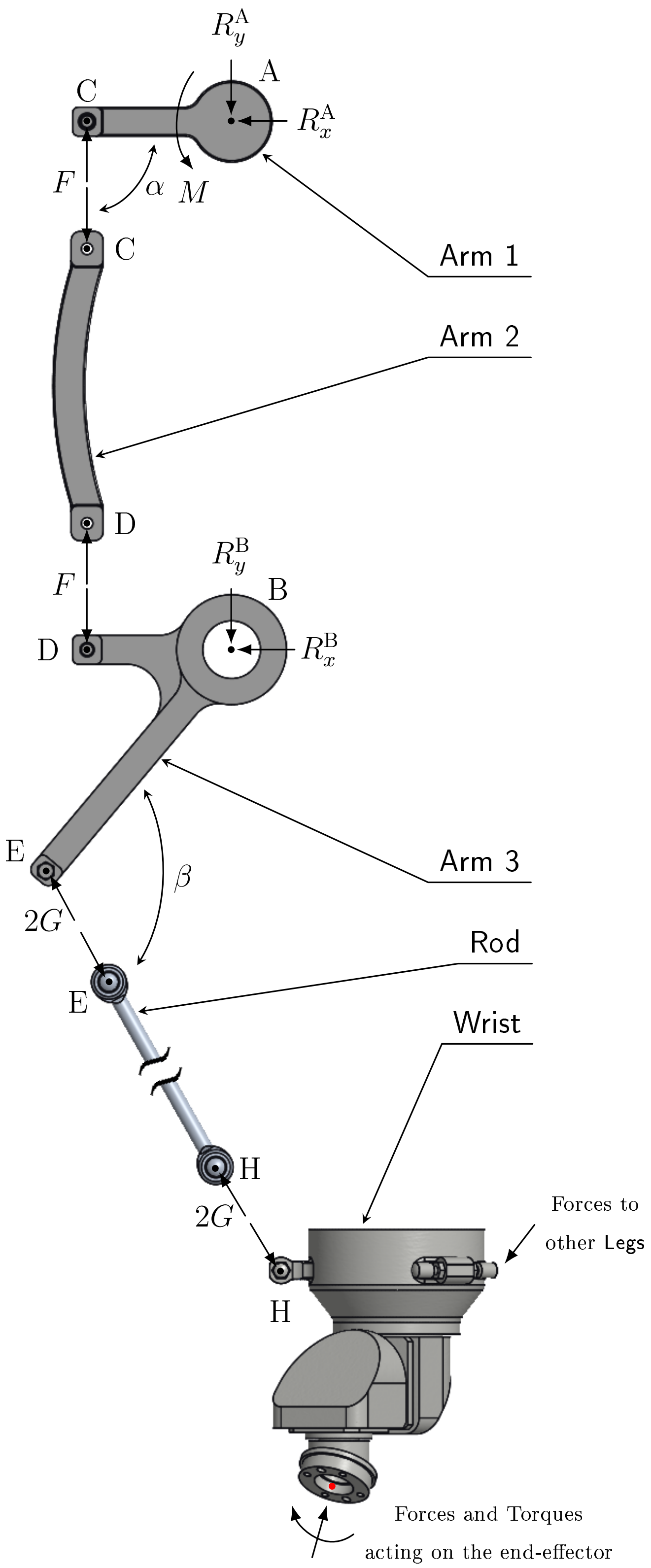
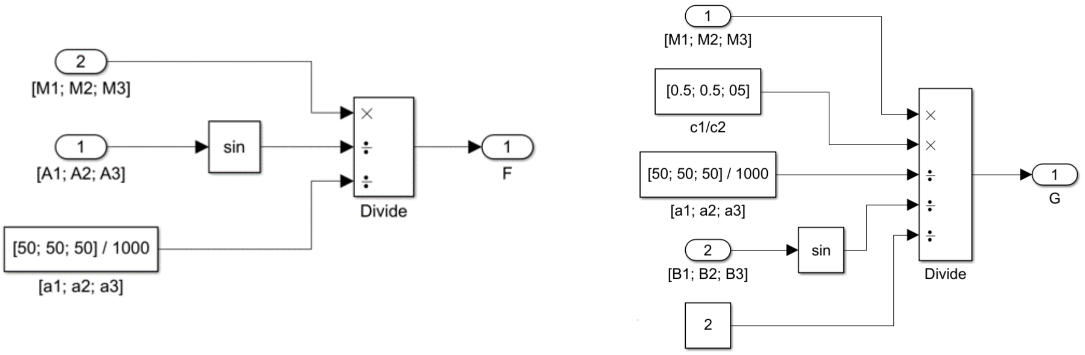


Figure 7.1: Exploded view of the components of Leg 2 or Leg 3 and the Wrist, with the force vectors at the joint points. The forces outside the plane of the paper are not represented. The compression of the Rod is $2G$ because there are two Rod elements per Leg (only one is visible here).



(a)

(b)

Figure 7.2: Subsystems to compute forces $F$ (left) and $G$ (right) according to (7.3) and (7.5). The input of the Divide blocks are vectors (on per Leg ), and it performs element-wise division.

$$\text{Torque balance of Arm 1 in A} \qquad \begin{gathered} M + \vec{a} \times \vec{F} = 0 \\ M - a \sin\alpha \, F = 0 \end{gathered} \tag{7.1}$$

$$\text{Torque balance of Arm 3 in B} \qquad \begin{gathered} \vec{c_1} \times \vec{F} + \vec{c_2} \times 2\vec{G} = 0 \\ c_1 \sin\alpha \, F - c_2 \sin\beta \, 2G = 0 \end{gathered} \tag{7.2}$$

Where $a$, $c_1$ and $c_2$ are the distances between the hinges (in A and B) and the forces application points, as per Figure 2.3. Their values are: $a = 50$ mm, $c_1 = 50$ mm and $c_2 = 100$ mm independently of the Leg.

Therefore the forces are:

$$F = \frac{M}{a \sin\alpha} \tag{7.3}$$

$$G = \frac{1}{2} \frac{c_1}{c_2} \frac{\sin\alpha}{\sin\beta} F \tag{7.4}$$

$$= \frac{1}{2} \frac{c_1}{c_2} \frac{M}{a \sin\beta} \tag{7.5}$$

The result for Leg 1 is analogous and yields the same formulas.

The equations 7.3 and 7.5 were implemented in the *Simulink* model with the block logic in Figure 7.2, and their result was saved by a To Workspace block.

## 7.2. Measuring Procedure

The end-effector was moved across all the workspace using the direct dynamics model setup. The points in workspace and joint space were the same used for the kinematic analysis of Chapter 6. The points where the measures were undertook formed a 3D grid in the workspace with a resolution of 4 mm in the $z$ direction and of 8 mm in the $x$ and $y$ direction. Three separate simulations were run, each with a force of 1 N in one of the three directions.

To measure forces without accelerations, the motors were ordered to remain in the prescribed position by specifying it twice in a row in the input matrix. In this way the From Input interpolation algorithm did not interfere by moving the motors continuously. As usual, the starting position of all the simulations was the zero configuration. The matrix containing the joint angles used in the From Workspace block had the following fashion:

$$\begin{array}{rccc}
0 & \varphi_0 & \vartheta_0 & \gamma_0 \\
t & \varphi_1 & \vartheta_1 & \gamma_1 \\
2\,t & \varphi_1 & \vartheta_1 & \gamma_1 \\
3\,t & \varphi_2 & \vartheta_2 & \gamma_2 \\
4\,t & \varphi_2 & \vartheta_2 & \gamma_2 \\
5\,t & \varphi_3 & \vartheta_3 & \gamma_3 \\
6\,t & \varphi_3 & \vartheta_3 & \gamma_3 \\
 & \vdots & & \\
(2n-1)\,t & \varphi_N & \vartheta_N & \gamma_N \\
2n\,t & \varphi_N & \vartheta_N & \gamma_N
\end{array}$$

Where $t = 0.1$ s is the time resolution used, $n = 17\,981$ is the number of points analysed, and the subscript 0 denotes the stating position.

The matrix of forces acting on the end-effector was different for every simulation, depending on the direction engaged. The first column contained the initial and final time, while the following three columns hosted respectively the force in radial, tangential and vertical direction. These three forces would be later converted during the simulation into the Cartesian components used by the model. Since the force is constant across all the workspace, there are only two rows with the same force values, therefore any interpolation returns the same force vector[1]. For clarity's sake the three force input matrices are here reported:

[1] Actually a single row would have produced the same result, because values can be hold constant by *Simulink* after the last timestamp (3.5.1). But the approach undertook here was deemed more logic and robust.

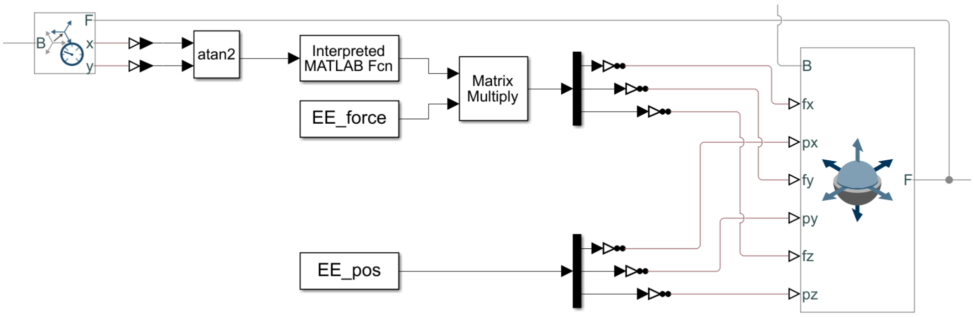


Figure 7.3: Blocks used to convert forces on the end-effector in cylindrical coordinate system to Cartesian coordinate system. As visible, the Transform Sensor shares the same follower with the 6-DOF Joint, but has a different base, namely a point on the central axis of the cylindrical reference frame.

| | | |
|---|---|---|
| 0 1 0 0 | 0 0 1 0 | 0 0 0 1 |
| $2nt$ 1 0 0 | $2nt$ 0 1 0 | $2nt$ 0 0 1 |
| Radial force | Tangential force | Vertical force |

## 7.2.1. Conversion of Cylindrical to Cartesian Coordinates

The conversion of force components from cylindrical to Cartesian coordinates is, of course, dependant on the position of the end-effector with respect to the axis of the cylindrical frame ($r = 0$). The change of coordinate system can be simply performed by a rotation in the $xy$ plane using the following rotation matrix:

$$\begin{pmatrix} F_x \\ F_y \\ F_z \end{pmatrix} = \begin{pmatrix} \cos\vartheta & -\sin\vartheta & 0 \\ \sin\vartheta & \cos\vartheta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} F_r \\ F_t \\ F_z \end{pmatrix} \tag{7.6}$$

Where $\vartheta$ is the azimuth angle between a frame coincident with the cylinder axis and the end-effector, as shown in Figure 7.4. It can be computed as the four-quadrant arctangent (atan2) of the distance in $y$ and $x$ between the end-effector and the workspace axis.

The described conversion was implemented in the *Simulink* model using a Transform Sensor to measure the distances, a Trigonometric Function to compute $\vartheta$, and an Interpreted MATLAB Function block to create the rotation matrix that multiplies the From Workspace input. The blocks are shown in Figure 7.3. The base and follower frames used by the Transform Sensor are visible in Figure 7.4.

Using the *Azimuth* output of the Transform Sensor would have been a more straightforward

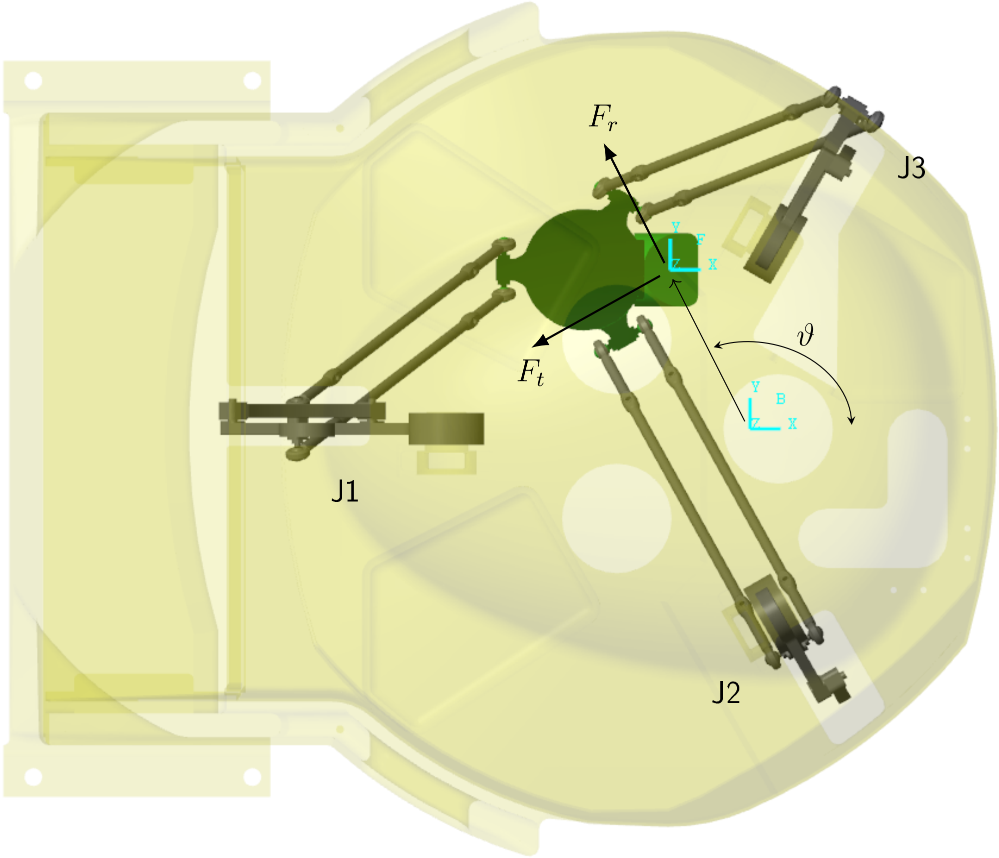


Figure 7.4: Top view of the robot in the *Mechanics Explorer*, in which the base and follower frames used by the Transform Sensor are shown. The angle $\vartheta$ between the frames is the azimuth angle. Radial and Tangential forces act in the directions shown. The position represented is $(\mathsf{J1} = +8, \mathsf{J2} = -29, \mathsf{J3} = +11)$ deg.

approach, but the block throws an error that stops the simulation when the base and follower frames coincide. Instead, the atan2 algorithm implemented in the Trigonometric Function block returns 0 when the frames are coincident.

### 7.2.2. Angle Measurement

The equations 7.3 and 7.5 require the knowledge of the angle $\alpha$ between Arm 1 and Arm 2, and the angle $\beta$ between Arm 3 and the Rod projected in the plane where the Arms lye.

$\alpha$ can be simply measured using the *Position* output of the Revolute Joint blocks in the point C.

The measure of $\beta$ is not straightforward due to the Spherical Joint in $E$, which allows rotations of the follower frame about all axis of the base frame. The base frame belongs to Arm 3, and the follower frame belongs to the Rod. Figure 7.5 shows the position and orientation of the two frames. $\beta$ is only the rotation in the plane of the paper, thus about the $y$ axis of the base frame. The Spherical Joint exploits the *spherical primitive* joint, thus its sensor output is a quaternion representation of the rotation. To obtain $\beta$, the quaternion has to be converted to a rotation sequence whose first axis is $y$. This procedure is conveniently performed by a Quaternions to Rotation Angles block from the *Aerospace Blockset* add-on [20]. The rotation sequence used was $yxz$. Since the angle $\beta$ is the same for both Rods in the parallelogram mechanism, it is measured for only one of them. Figure 7.6 shows how the computation of $\beta$ was implemented in the *Simulink* model for J1.

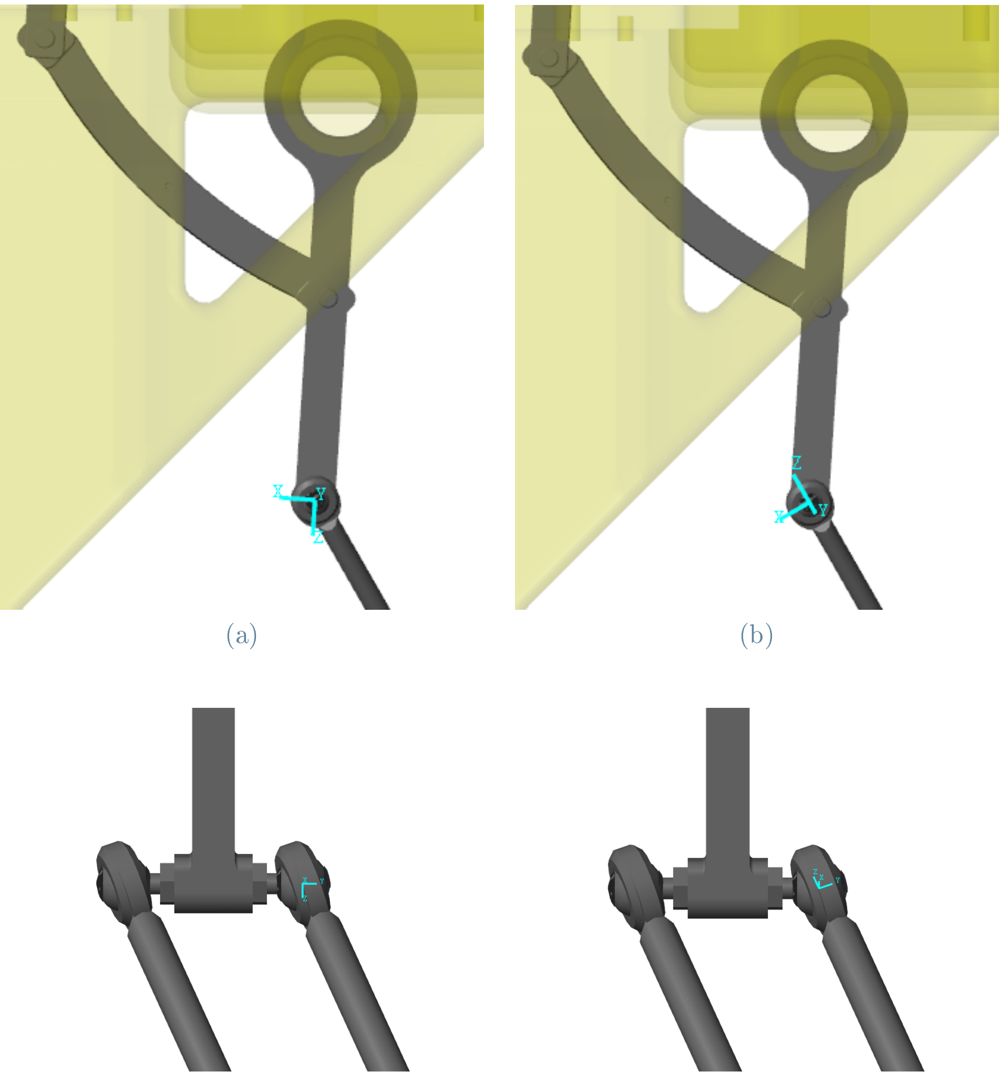
It is worth to notice that the effort of smartly aligning reference frames in the post-processing of the imported model (4.3.1) was now compensated.

### 7.2.3. Isolation of Static Results

The output of the simulation consisted in $N \times 3$ matrices with the position in the working space, position in the joint space, motor torques, and forces $F$ and $G$ for each Leg . Every matrix had three version: one per each force direction.

Before the analysis, the simulation output was filtered to remove non-static points. In particular, the following constraints had to be satisfied:

- norm of acceleration in the joint space: $||\ddot{\mathbf{q}}|| < 10^{-6}\ ^{\mathrm{rad}}/_{\mathrm{s}^2}$ ;
- norm of velocity in the joint space: $||\dot{\mathbf{q}}|| < 10^{-4}\ ^{\mathrm{rad}}/_{\mathrm{s}}$ ;



(a)

(b)

(c)

(d)

Figure 7.5: Orientation of the frames of the Spherical Joint for J1: base (on the left-hand side) and follower (on the right-hand side), seen from the side (top pictures) and from the front (bottom pictures). The position represented is $(\mathsf{J1} = -27, \mathsf{J2} = -39, \mathsf{J3} = +17)$ deg.

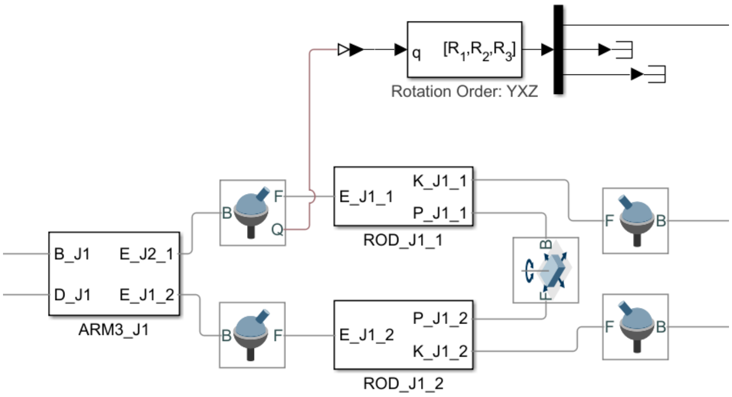


Figure 7.6: Conversion of the quaternion rotation of the Spherical Joint connecting Arm 3 and Rod into angle $\beta$.

- $L_1$ distance[2] of consecutive points in working space: $\|\Delta\mathbf{x}\|_1 < 10^{-2}$ mm;
- $L_1$ distance of consecutive torques: $\|\Delta T\|_1 < 10^{-4}$ Nm.

The threshold values were decided based on empirical experience. Only approximately 35% of the point acquired proved to be static according to the above constraints.

The values preserved after the filter were clustered in groups with constant values, which were then averaged into a single element. Figure 7.7 shows the data before and after filtering.

## 7.3. Results

### 7.3.1. Motor torques

The extreme values of motor torques are reported in Table 7.1. While it is evident the similarity of J2 and J3 in the case of tangential and vertical load, the same does not hold for the radial load. By comparing the values of the table vertically, it appears that J2 behaves differently. This is also clear by confronting the contour plot of J2 and J3 in Figure B.2, or the histograms of $M$. J2 and J3 have overlapping histograms for tangential and vertical force (Figures B.7 and B.13), but not for the force in radial direction (Figure B.1). A possible reason is a slight misalignment of the workspace axis with respect to the actual centre between the Legs. A more likely reason could be an accidental mistake in the code used to post-process the simulation output.

[2] The $L_1$ distance (or $L_1$ norm, or Manhattan distance) is $\|\Delta\mathbf{x}\|_1 = \Delta x + \Delta y + \Delta z$, while the norm refers to the $L_2$ distance (or $L_2$ norm, or Euclidean norm) ($\|\mathbf{q}\| \equiv \|\mathbf{q}\|_2 = \sqrt{q_1^2 + q_2^2 + q_3^2}$).

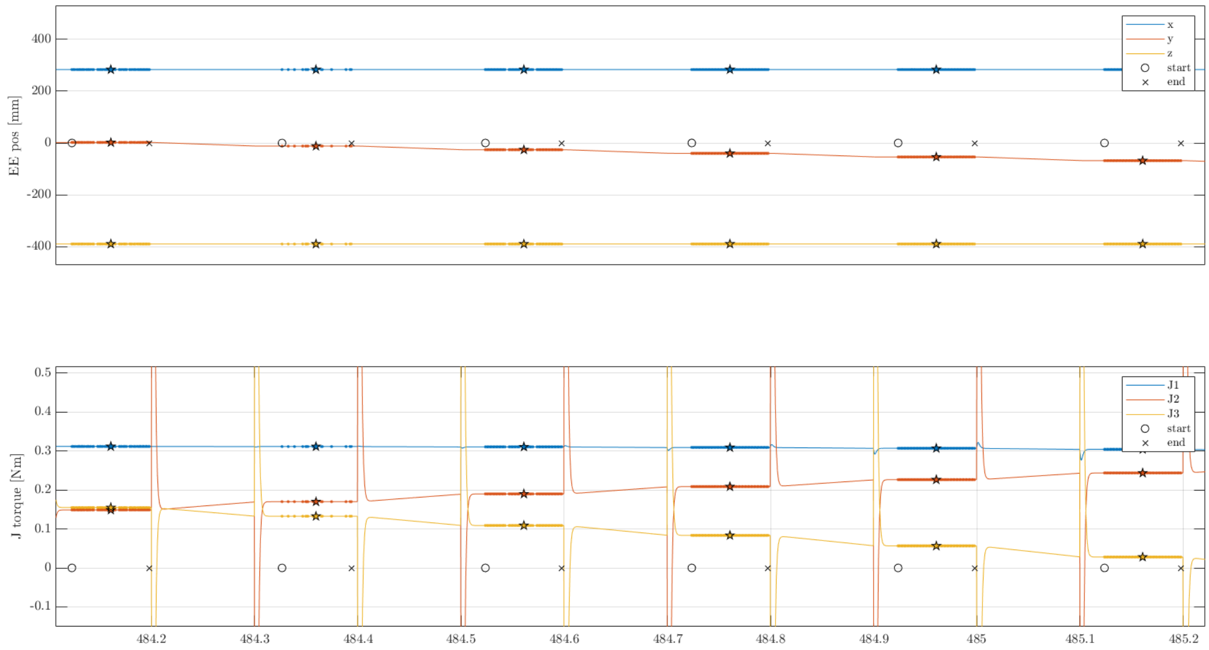


Figure 7.7: Sample of output data before and after filtering: the straight thin lines represent original values, and the points represents filtered values. These filtered values create well separated regions whose mean value is shown by coloured stars (⋆). The ∘ and × symbols on the zero axis designate respectively the begin and end of a region. Only the position in working space and torque are shown here, but the same process was applied to all the outputs.

**Max and Min Torques**

| | | $M$ | | |
|---|---|---|---|---|
| | | **J1** [Nm] | **J2** [Nm] | **J3** [Nm] |
| *Radial load* | *max* | 0.44 | 0.48 | 0.35 |
| | *min* | −0.12 | −0.19 | 0.00 |
| *Tangential load* | *max* | 0.47 | 0.41 | 0.41 |
| | *min* | −0.23 | −0.04 | −0.04 |
| *Vertical load* | *max* | 0.29 | 0.29 | 0.29 |
| | *min* | −0.08 | −0.05 | −0.05 |

Table 7.1: Extreme value of the motor torques in response to unitary load in the three directions.

**Max and Min Forces**

| | | $F$ | | | $G$ | | |
|---|---|---|---|---|---|---|---|
| | | **J1** [N] | **J2** [N] | **J3** [N] | **J1** [N] | **J2** [N] | **J3** [N] |
| **Radial load** | *max* | 11.25 | 13.92 | 9.50 | 2.22 | 2.42 | 1.82 |
| | *min* | −4.55 | −5.95 | −0.11 | −1.06 | −1.47 | −0.03 |
| **Tangential load** | *max* | 13.32 | 10.46 | 10.48 | 2.41 | 2.06 | 2.08 |
| | *min* | −7.90 | −1.43 | −1.44 | −1.80 | −0.36 | −0.36 |
| **Vertical load** | *max* | 8.31 | 8.91 | 8.86 | 1.50 | 1.49 | 1.49 |
| | *min* | −3.35 | −1.96 | −1.96 | −0.76 | −0.48 | −0.48 |

Table 7.2: Extreme value of the forces $F$ and $G$ in response to unitary load in the three directions.

The motor torques in response to a unitary force are six-dimensional data structures. An easy way to visualize them is to consider just the points on a plane of the workspace, and show the corresponding torques as three surfaces. The Appendix B contains some figures that show the motor response to loads on the end-effector.

As one could expect, the three Legs have similar behaviour, and in particular J2 and J3 are specular. The contour plots of $\max|M_i|$ in Appendix B show a minimum value slightly on the $-y$ direction from the centre of the workspace. This is likely due to the offset of the translational workspace used for this analysis and the nominal workspace.

## 7.3.2. Distribution of Forces on the Structure

The extreme values of forces $F$ and $G$ are reported in Table 7.2. As for the motor torques in Table 7.1, J2 and J3 shares similar values except for the radial load due to inconsistent values of forces in J2.

The value of the forces across the workspace for different loads is illustrated in the figures of the second part of Appendix B.

# 8 | Conclusions and future developments

In this thesis the Fanuc M-1*i*A/0.5A robot was modelled in *Simulink* and some of its operational characteristics were analysed. In particular the work focused on determining the workspace of the robot with fixed-attitude Wrist, solving the kinematics problems by building lookup tables, and computing the motor torques and structural forces.

The analysis of the workspace revealed that it could be larger than the nominal workspace declared by the manufacturer. The largest possible translational workspace has a complex shape with a third-order radial symmetry, thus two simplified version of the workspace were proposed. The end-effector has decent manipulability in a vast area.

The model *Simscape Multibody* assembler was exploited to solve the inverse kinematic problem, and ultimately to build lookup tables useful to solve the forward and inverse problems by interpolation. At the same time, the result were used to visualize the joint space and its range. It was found that the limits of a joint motion depends on the position of the other joints. The Jacobian was evaluated in every point of the joint space in order to solve the differential kinematics problems. The Jacobian is always full-rank, but in some point on the workspace border it is ill-conditioned, thus close to singularity.

A static equilibrium analysis of the robot Legs was used to identify how forces transmit in the structure from the end-effector to the motors. The motor torque, and the compression forces of the components were evaluated in three load scenarios: radial, tangential or vertical force with respect to the Wrist central axis. This analysis did not account for inertial forces. The result shows how the aforementioned forces vary in the workspace in the three scenarios. It was found that J2 and J3, which are geometrically identical, have specular response, except for radial loads. As regarding J1, it supports most of the load, and has larger extreme values, both in positive and negative direction.

The results of this thesis advanced the academic knowledge about the Fanuc M-1*i*A robots. Another important achievement was the creation of a framework to study the robot operational characteristics from a CAD drawing. The same analysis can be undertaken on different models with minor edits, or can be scaled to more than three dof. The framework could also be expanded to analyse more characteristics of the robot.

Future works on this topic could expand this study by analysing more characteristic, such as dynamic performances. The *Simulink* model has some possibility of improvement: for instance stiffness, damping and friction of the joints could be modelled. A great improvement would be represented by modelling also the Wrist rotation. Finally, future work may focus on co-simulation to validate the results here presented.

# Bibliography

[1] I. Bonev. Delta parallel robot — the story of success, 2001. URL http://www.parallemic.org/Reviews/Review002.html. Accessed 2022-07-11.

[2] G. Carabin, L. Scalera, T. Wongratanaphisan, and R. Vidoni. An energy-efficient approach for 3d printing with a linear delta robot equipped with optimal springs. *Robotics and Computer-Integrated Manufacturing*, 67, FEB 2021. ISSN 0736-5845. doi: 10.1016/j.rcim.2020.102045.

[3] R. Celi, A. Sempertegui, D. Morocho, D. Loza, D. Alulema, and M. Proano. Study, design and construction of a 3d printer implemented through a delta robot. In M. Fernandez, G. Lefranc, and R. Perez, editors, *2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, pages 717–722, 2015. ISBN 9781467387569.

[4] R. Clavel. Device for the movement and positiong of an element in space, 1990. URL https://pdfpiw.uspto.gov/.piw?docid=04976582.

[5] Fanuc. *M-1iA/0.5A Datasheet*, 2019. URL https://www.fanuc.eu/~/media/files/pdf/products/robots/robots-datasheets-en/m-1ia/datasheet%20m-1ia-05a.pdf.

[6] Fanuc. *Fanuc Robot M-1iA Brochure*, 2019. URL https://www.fanuc.co.jp/en/product/catalog/pdf/robot/RM-1iA(E)-06c.pdf.

[7] C. Gosselin and J. Angeles. Singularity analysis of closed-loop kinematic chains. *IEEE Transactions on Robotics and Automation*, 6(3):281–290, 1990. doi: 10.1109/70.56660.

[8] D. Grajewski, F. Gorski, D. Rybarczyk, P. Owczarek, A. Milecki, and P. Bun. Use of delta robot as an active touch device in immersive case scenarios. In E. Ginters and J. Kohlhammer, editors, *ICTE 2016*, volume 104 of *Procedia Computer Science*, pages 485–492, 2017. doi: 10.1016/j.procs.2017.01.163.

[9] K. M. Lynch and F. C. Park. *Modern Robotics.* Cambridge University Press, 2017. ISBN 9781107156302. URL `http://modernrobotics.org/`.

[10] MathWorks. How multibody assembly works, . URL `https://www.mathworks.com/help/physmod/sm/ug/assembling-multibody-systems.html`. Accessed 2022-06-23.

[11] MathWorks. Cad translation, . URL `https://www.mathworks.com/help/physmod/sm/ug/cad-translation.html`. Accessed 2022-06-23.

[12] MathWorks. Specifying joint actuation inputs, . URL `https://www.mathworks.com/help/physmod/sm/ug/joint-actuation.html`. Accessed 2022-06-23.

[13] MathWorks. Simulink, . URL `https://www.mathworks.com/products/simulink.html`. Accessed 2022-06-26.

[14] MathWorks. Physical modeling, . URL `https://www.mathworks.com/solutions/physical-modeling.html`. Accessed 2022-06-26.

[15] MathWorks. Simscape block libraries, . URL `https://www.mathworks.com/help/physmod/simscape/ug/introducing-the-simscape-block-libraries.html`. Accessed 2022-06-26.

[16] MathWorks. Multibody model anatomy, . URL `https://www.mathworks.com/help/physmod/sm/gs/multibody-model-anatomy.html`. Accessed 2022-06-26.

[17] MathWorks. Working with frames, . URL `https://www.mathworks.com/help/physmod/sm/ug/frames-and-frame-transforms.html`. Accessed 2022-06-27.

[18] MathWorks. Modeling joint connection, . URL `https://www.mathworks.com/help/physmod/sm/ug/joints.html`. Accessed 2022-06-27.

[19] MathWorks. Load Data Using the From Workspace Block, . URL `https://www.mathworks.com/help/simulink/ug/load-data-using-the-from-workspace-block.html`. Accessed 2022-06-28.

[20] MathWorks. Quaternions to Rotation Angles, . URL `https://www.mathworks.com/help/aeroblks/rotationanglestoquaternions.html`. Accessed 2022-07-02.

[21] MathWorks. Simulink-PS converter, . URL `https://www.mathworks.com/help/physmod/simscape/ref/simulinkpsconverter.html`. Accessed 2022-06-28.

[22] MathWorks. rmoutliers, . URL `https://www.mathworks.com/help/matlab/ref/rmoutliers.html`. Accessed 2022-07-20.

[23] J.-P. Merlet. *Parallel Robots.* Springer, second edition, 2006. ISBN 9781402041330/4.

[24] J.-P. Merlet, C. Gosselin, and T. Huang. *Handbook of Robotics*, chapter Parallel Mechanisms, pages 443–461. Springer, second edition, 2006. ISBN 9783319325507/21. doi: 10.1007/978-3-319-32552-1. URL `http://handbookofrobotics.org`.

[25] C. Mitsantisuk and K. Ohishi. Haptic human-robot collaboration system based on delta robot with gravity compensation. In *Proceedings of the Iecon 2016 - 42nd Annual Conference of the Ieee Industrial Electronics Society*, IEEE Industrial Electronics Society, pages 5796–5801, 2016. ISBN 9781509034741.

[26] C. Mitsantisuk, S. Stapornchaisit, N. Niramitvasu, and K. Ohishi. Force sensorless control with 3d workspace analysis for haptic devices based on delta robot. In *Iecon 2015 - 41st Annual Conference of the Ieee Industrial Electronics Society*, IEEE Industrial Electronics Society, pages 1747–1752, 2015. ISBN 9781479917624.

[27] C. Preault, H. Saafi, M. A. Laribi, and S. Zeghloul. 4haptic: A dexterous 4 d.o.fs haptic device based on delta architecture. In S. Zeghloul, L. Romdhane, and M. Laribi, editors, *Computational Kinematics*, volume 50 of *Mechanisms and Machine Science*, pages 307–314, 2018. ISBN 9783319608662/79. doi: 10.1007/978-3-319-60867-9_35.

[28] R. Ranganath, P. Nair, T. Mruthyunjaya, and A. Ghosal. A force–torque sensor based on a stewart platform in a near-singular configuration. *Mechanism and Machine Theory*, 39(9):971–998, 2004. ISSN 0094-114X. doi: https://doi.org/10.1016/j.mechmachtheory.2004.04.005.

[29] E. Rodriguez, C. Riano, A. Alvares, and R. Bonnard. Design and dimensional synthesis of a linear delta robot with single legs for additive manufacturing. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 41(11), November 2019. ISSN 1678-5878. doi: 10.1007/s40430-019-2039-6.

[30] H. Saafi, C. Preault, M. A. Laribi, and S. Zeghloul. A novel kinematic of a 4 d.o.fs haptic device based on the delta robot architecture. In C. Ferraresi and G. Quaglia, editors, *Advances in Service and Industrial Robotics*, volume 49 of *Mechanisms and Machine Science*, pages 699–706, 2018. ISBN 9783319612751/68. doi: 10.1007/978-3-319-61276-8_74.

[31] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics.* Springer, 2009. ISBN 9781846286414/21. doi: 10.1007/978-1-84628-642-1.

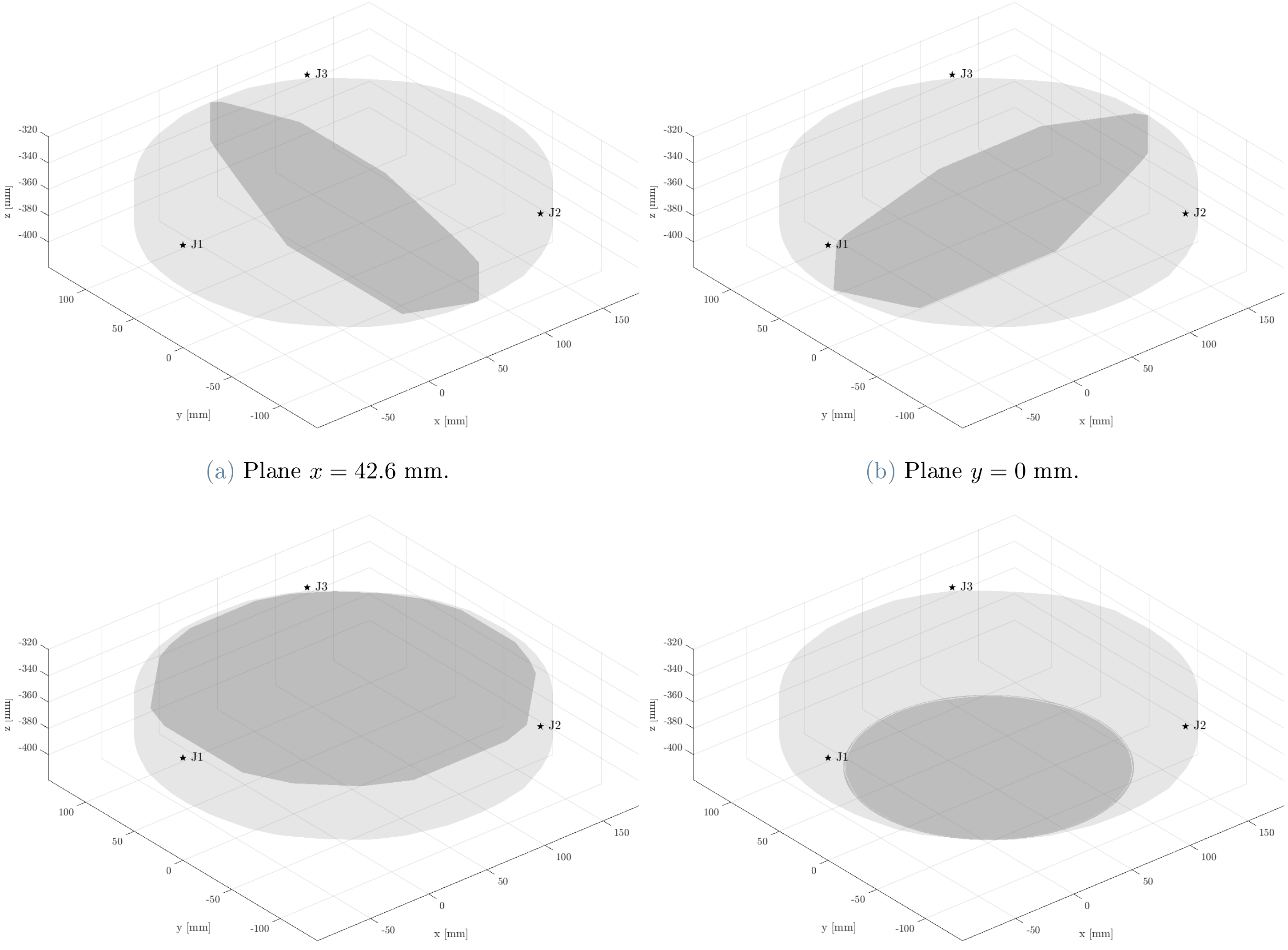
[32] B. Xiao, A. Alamdar, K. Song, A. Ebrahimi, P. Gehlbach, R. H. Taylor, and I. Ior-

dachita. Delta robot kinematic calibration for precise robot-assisted retinal surgery. In *2022 International Symposium on Medical Robotics (ISMR)*, pages 1–7, 2022. doi: 10.1109/ISMR48347.2022.9807517.

# A | Manipulability

In this section of the document are reported some images to show the spatial distribution of the isotropy index $\mu$ and the volume index $v$ of the manipulability ellipsoids, as well as the directions of the longest and shortest ellipsoid semi-axes. The first part shows tridimensional contour plot, while the second has some 2D contour plot in some planes. The planes used for slicing the workspace are shown in Figure A.1.

The contour plot of isotropy features vectors representing some of the eigenvector with the largest and smallest associated eigenvalue. The vectors are scaled by their eigenvalue, therefore the two vectors departing from a point where $\mu \approx 1$ have the same length. Despite the three dimensional vector are not representable in the image, there are clearly distinguishable patterns in eigenvectors direction.



(a) Plane $x = 42.6$ mm.

(b) Plane $y = 0$ mm.

(c) Plane $z = -340$ mm.

(d) Plane $z = -400$ mm.

Figure A.1: Cutting planes used to visualise isotropy and volume in the workspace.

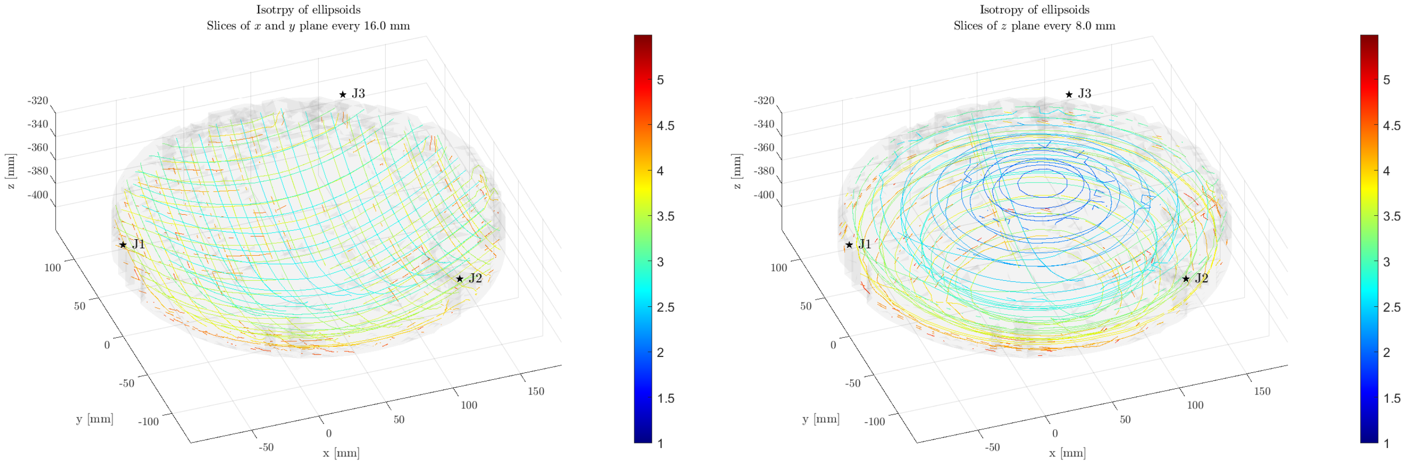


Figure A.2: Tridimensional contour of the ellipsoid isotropy index $\mu$.

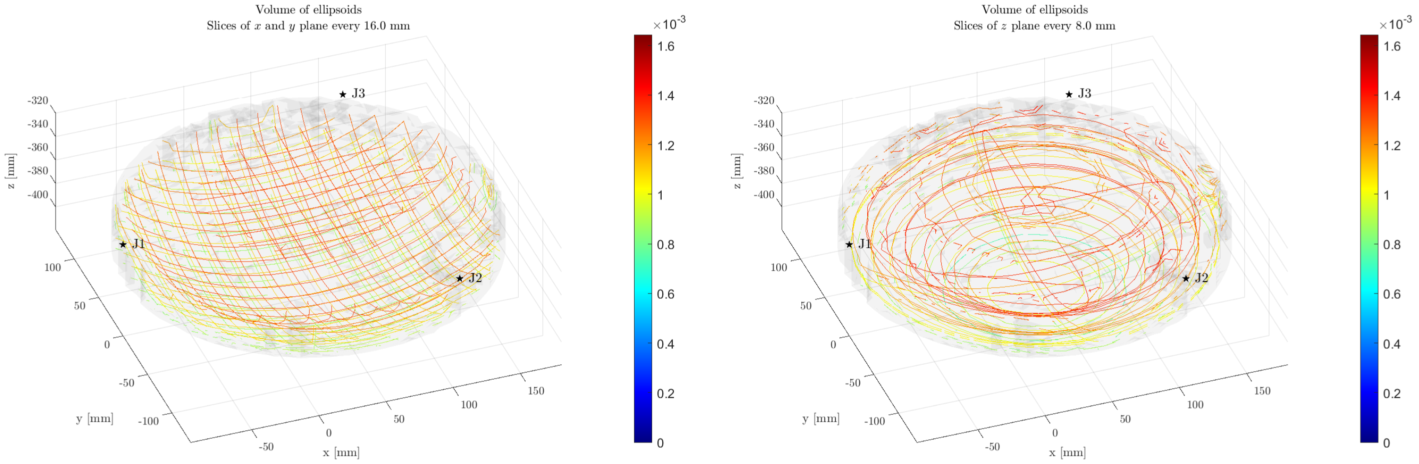


Figure A.3: Tridimensional contour of the ellipsoid volume index $v$.

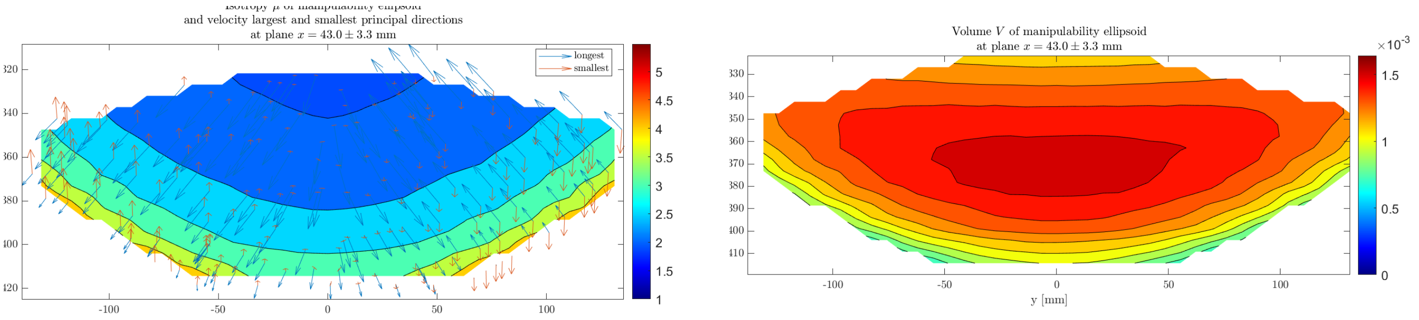


Figure A.4: Isotropy index $\mu$ and volume $v$ on the plane $x = 42.6$ mm.

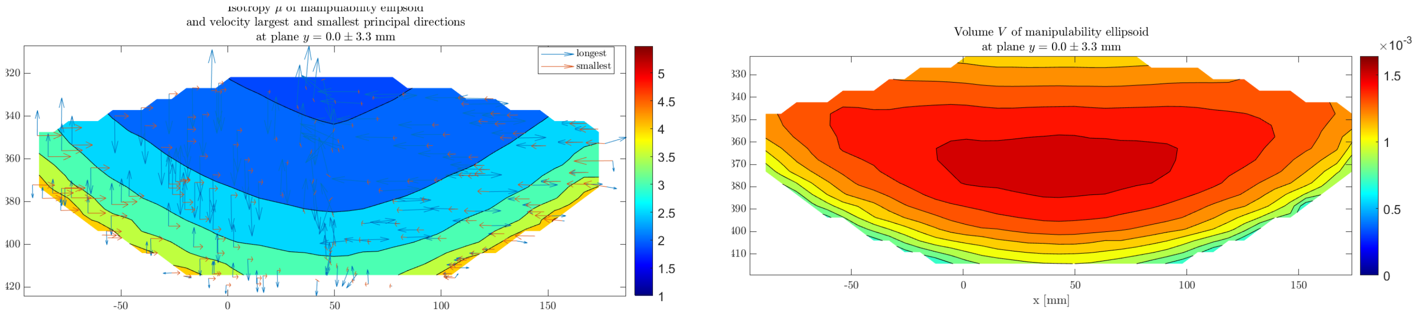


Figure A.5: Isotropy index $\mu$ and volume $v$ on the plane $y = 0$ mm.

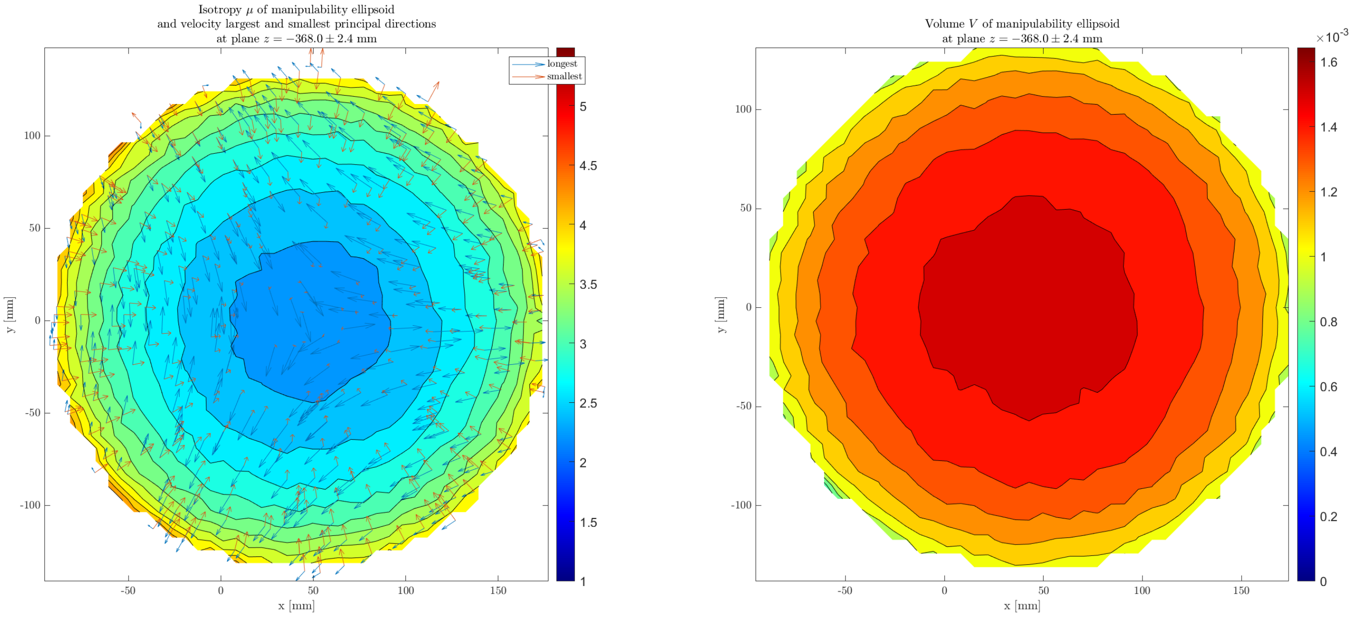


Figure A.6: Isotropy index $\mu$ and volume $v$ on the plane $z = -340$ mm.

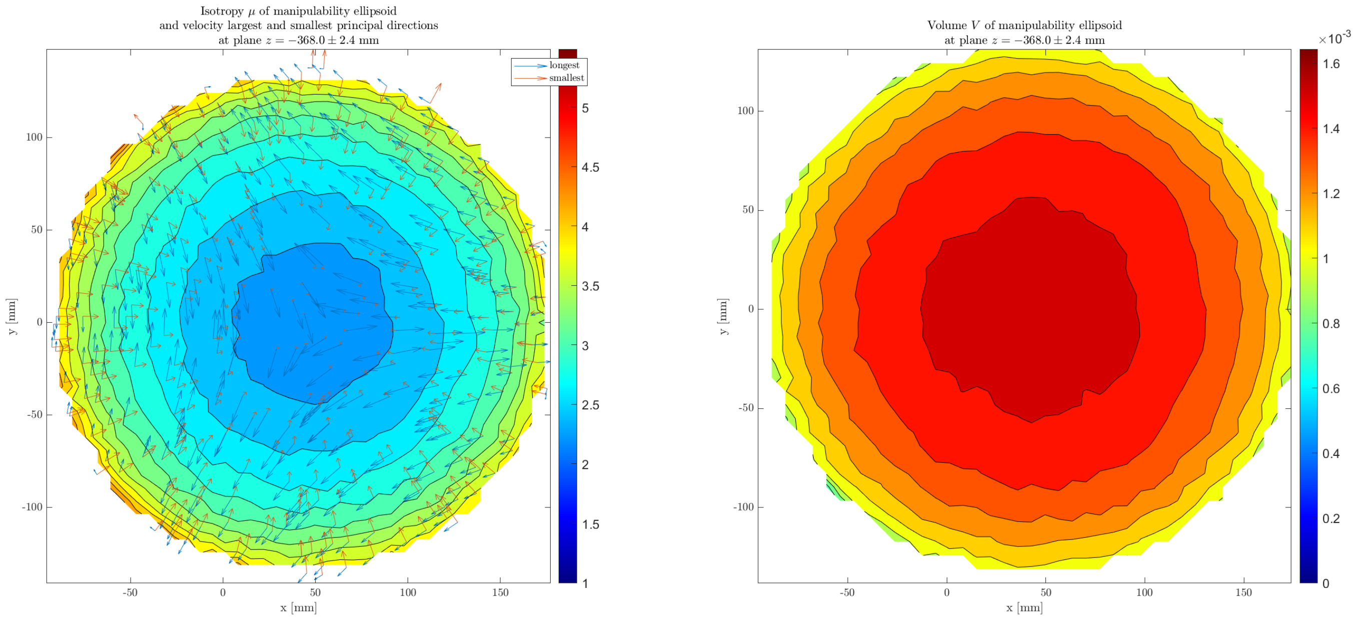


Figure A.7: Isotropy index $\mu$ and volume $v$ on the plane $z = -400$ mm.

# B | Torques and Forces

In this Chapter the result on the static force analysis are presented. The effect of radial, tangential and vertical load is presented separately. Every section has an histogram, a tridimensional contour plot, and a curve and contour plot representing specific planes of the workspace. These planes are the same used in Appendix A, shown in Figure A.1.

## B.1. Motor torques

### B.1.1. Radial Load

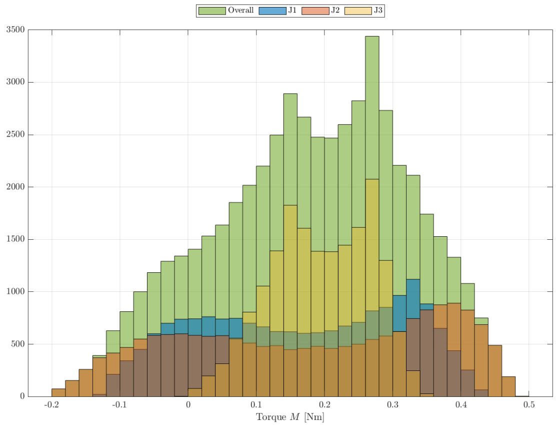


Figure B.1: Histogram of motor torques with radial unitary load.

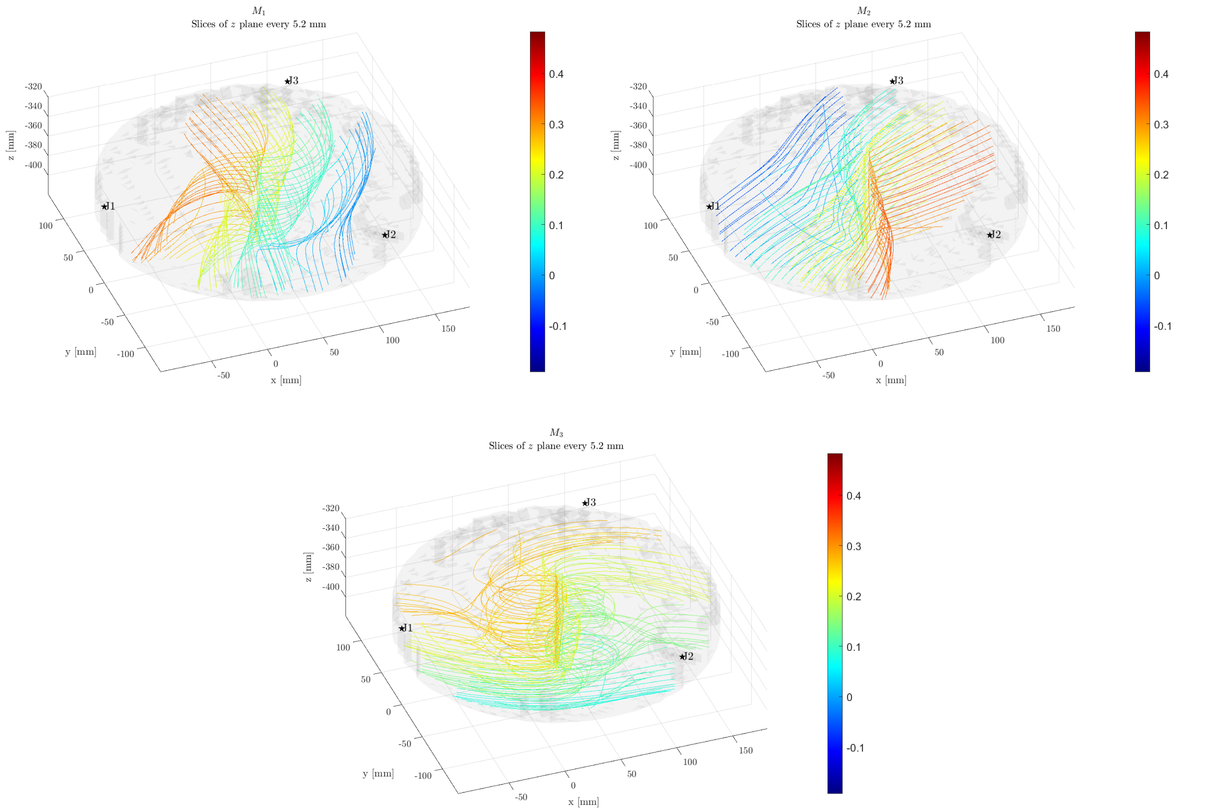
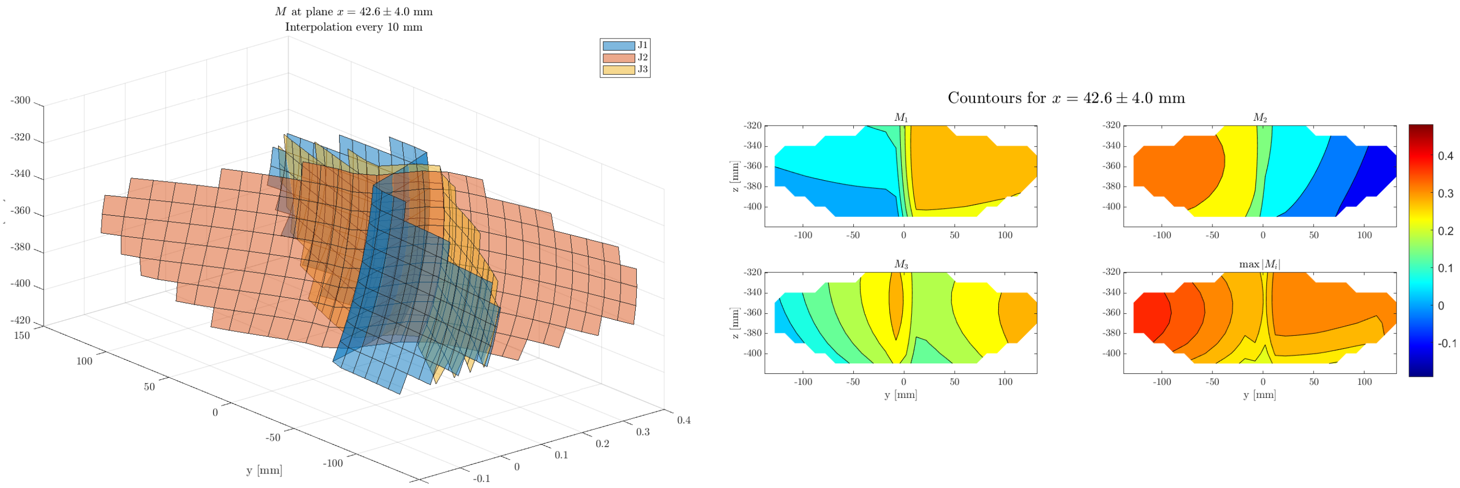


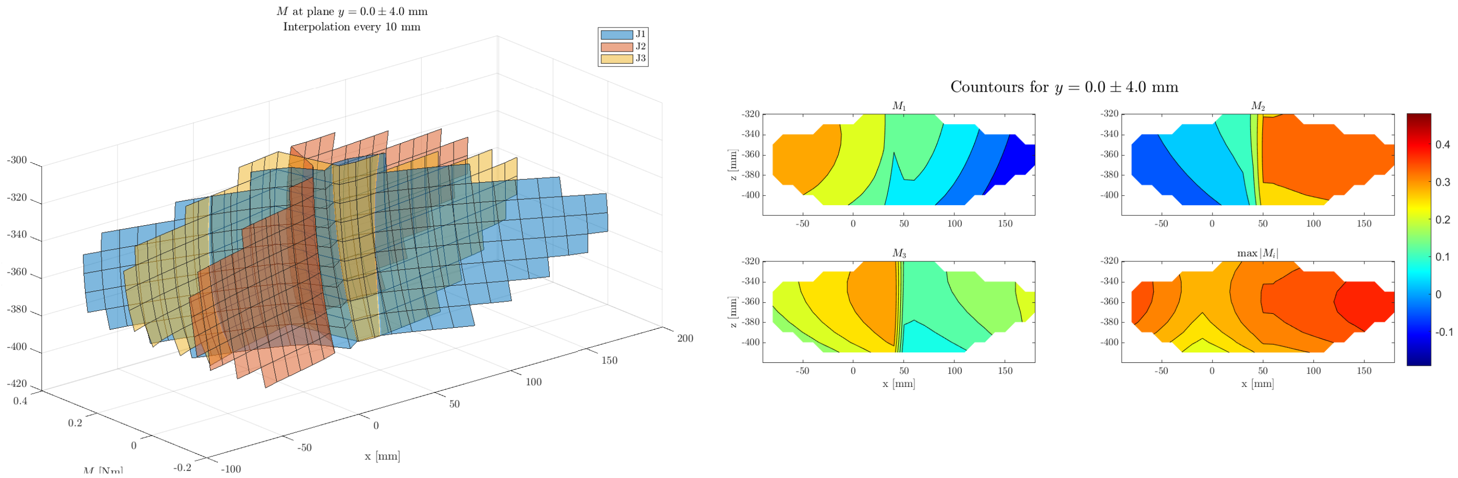
Figure B.2: Volumetric contour plot of torque in response to a radial unitary load on the end-effector.



(a) Torque curves

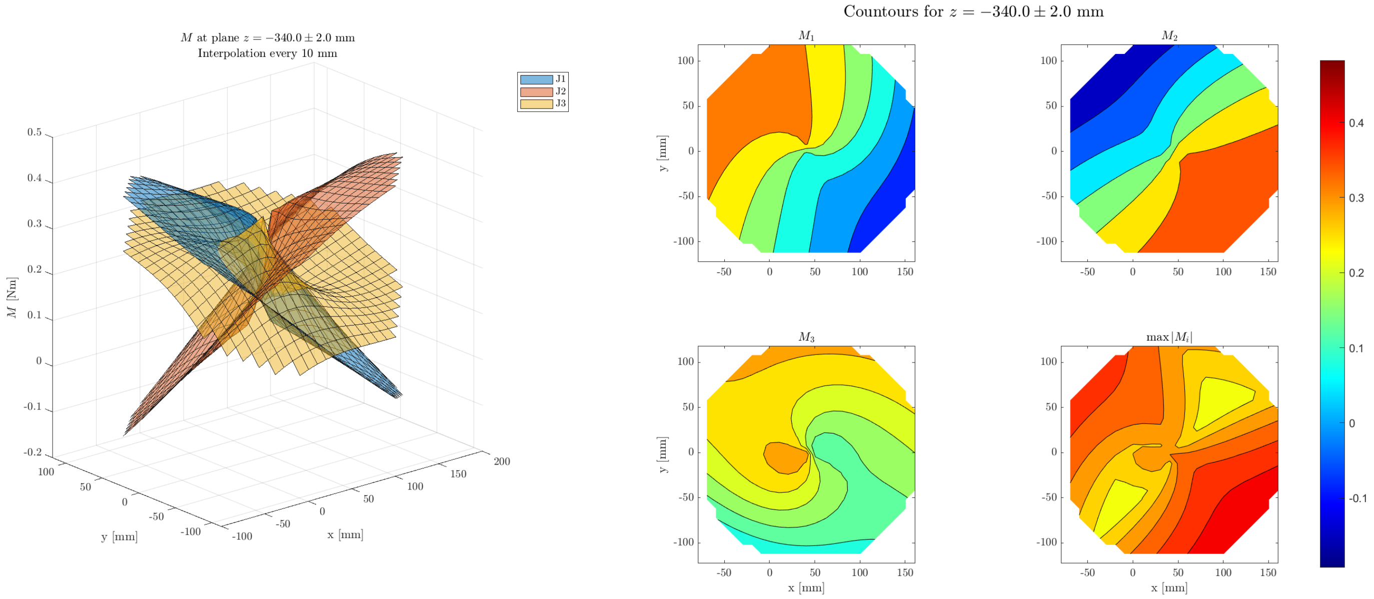
(b) Torque contours

Figure B.3: Motor torque in response to a radial unitary load when the end-effector is in the plane $x = 42.6$ mm.



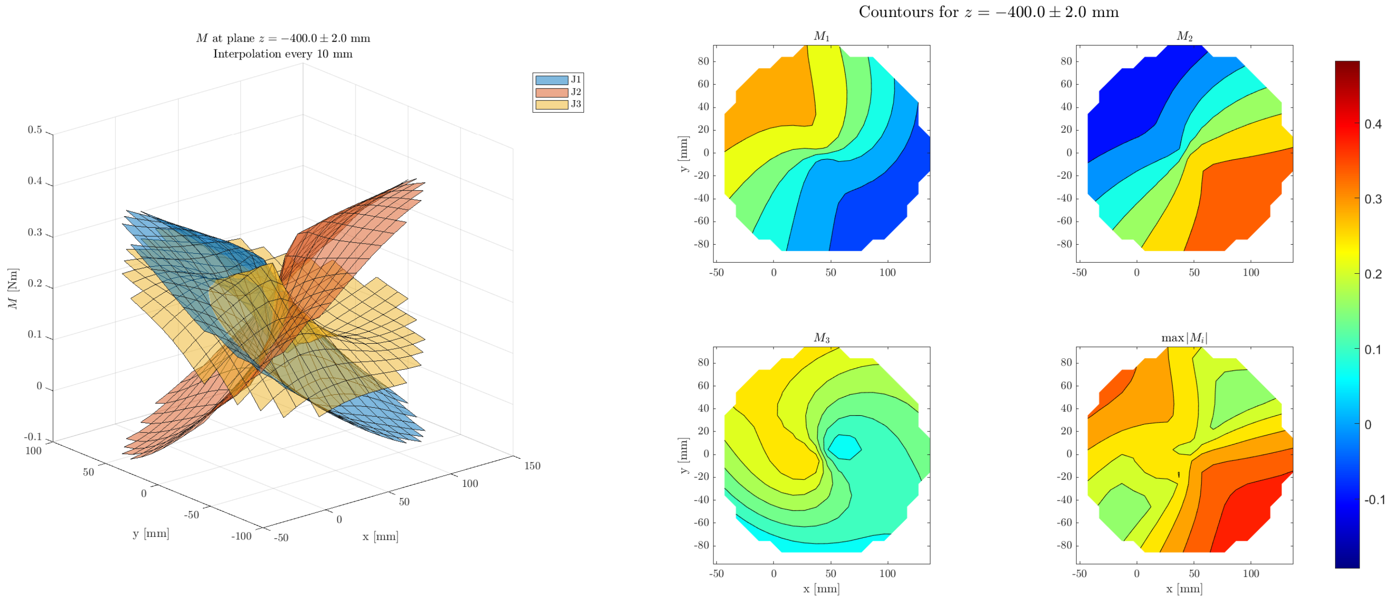
(a) Torque curves (b) Torque contours

Figure B.4: Motor torque in response to a radial unitary load when the end-effector is in the plane $y = 0$ mm.



(a) Torque curves (b) Torque contours

Figure B.5: Motor torque in response to a radial unitary load when the end-effector is in the plane $z = -340$ mm.



(a) Torque curves

(b) Torque contours

Figure B.6: Motor torque in response to a radial unitary load when the end-effector is in the plane $z = -400$ mm.

## B.1.2. Tangential Load

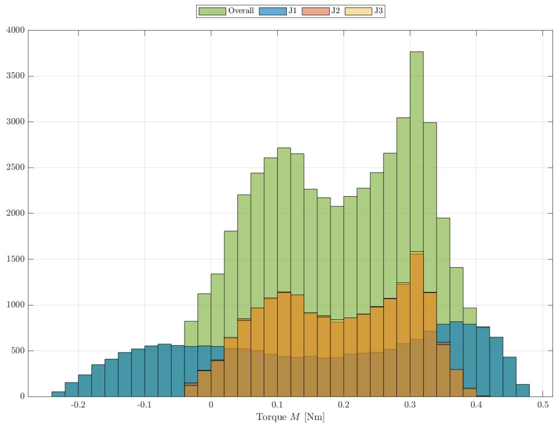


Figure B.7: Histogram of motor torques with tangential unitary load.

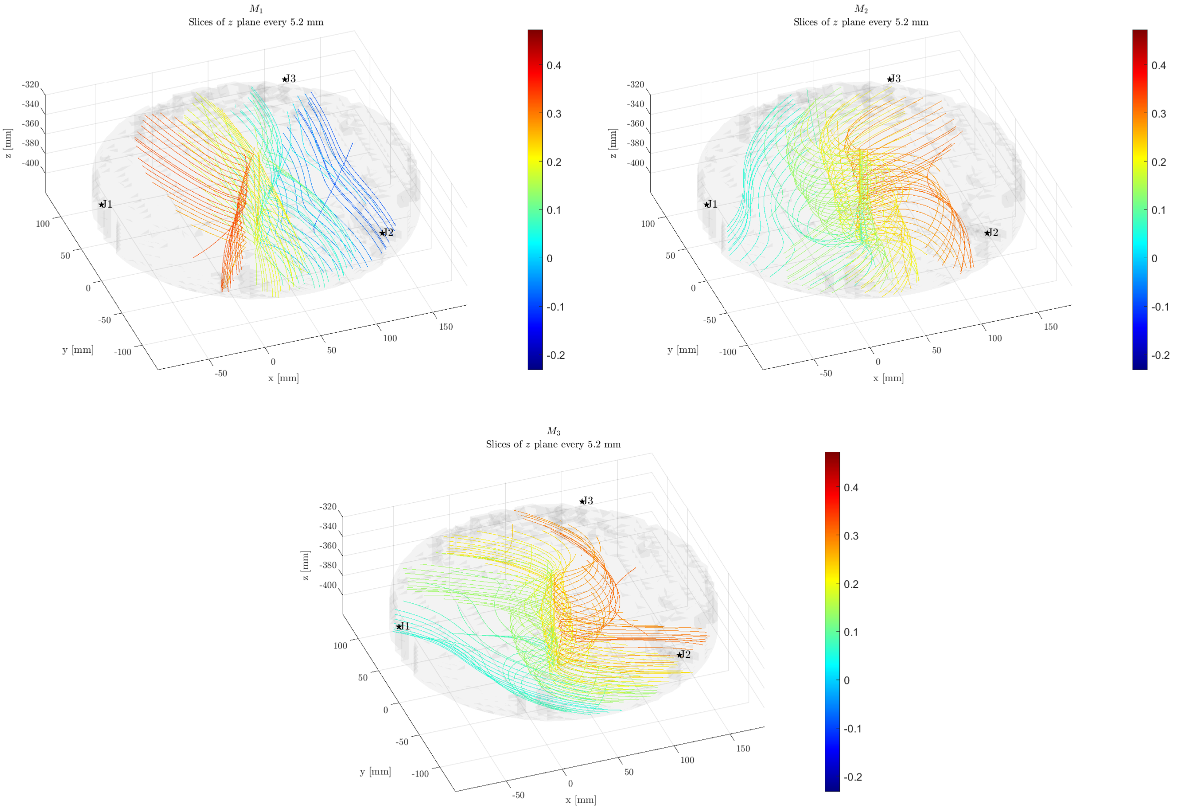
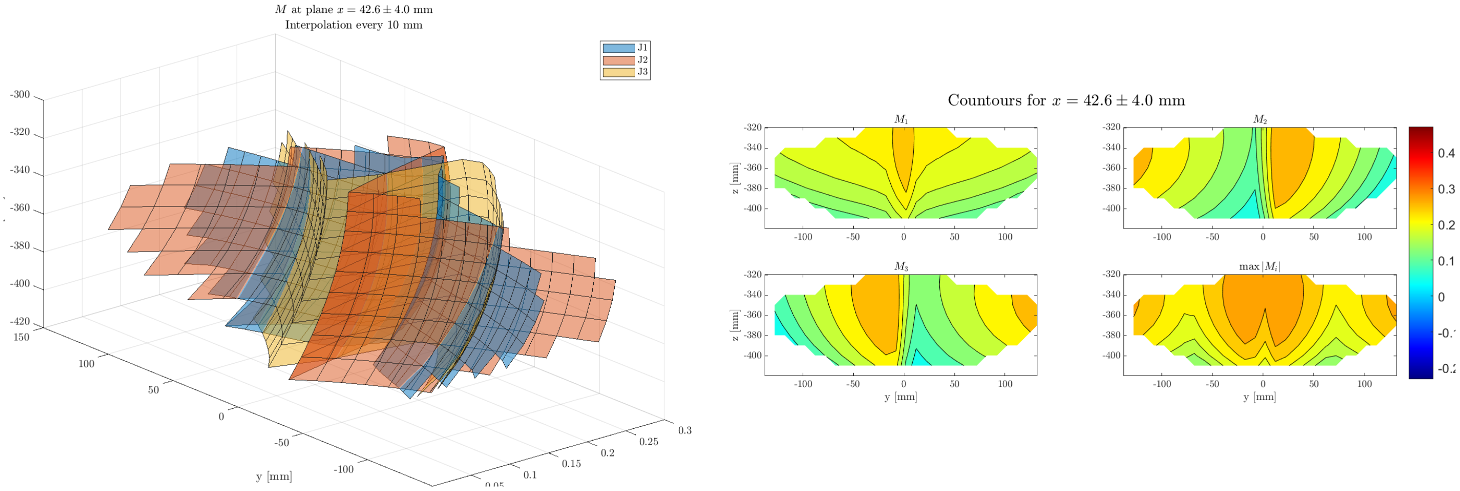


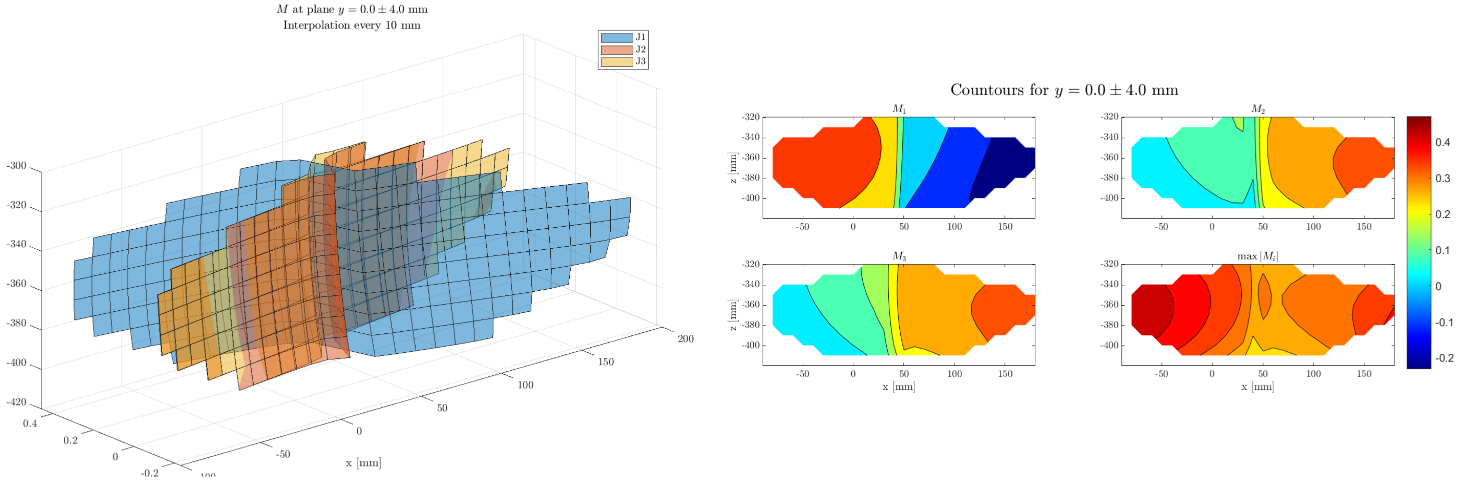
Figure B.8: Volumetric contour plot of torque in response to a tangential unitary load on the end-effector.



(a) Torque curves

(b) Torque contours

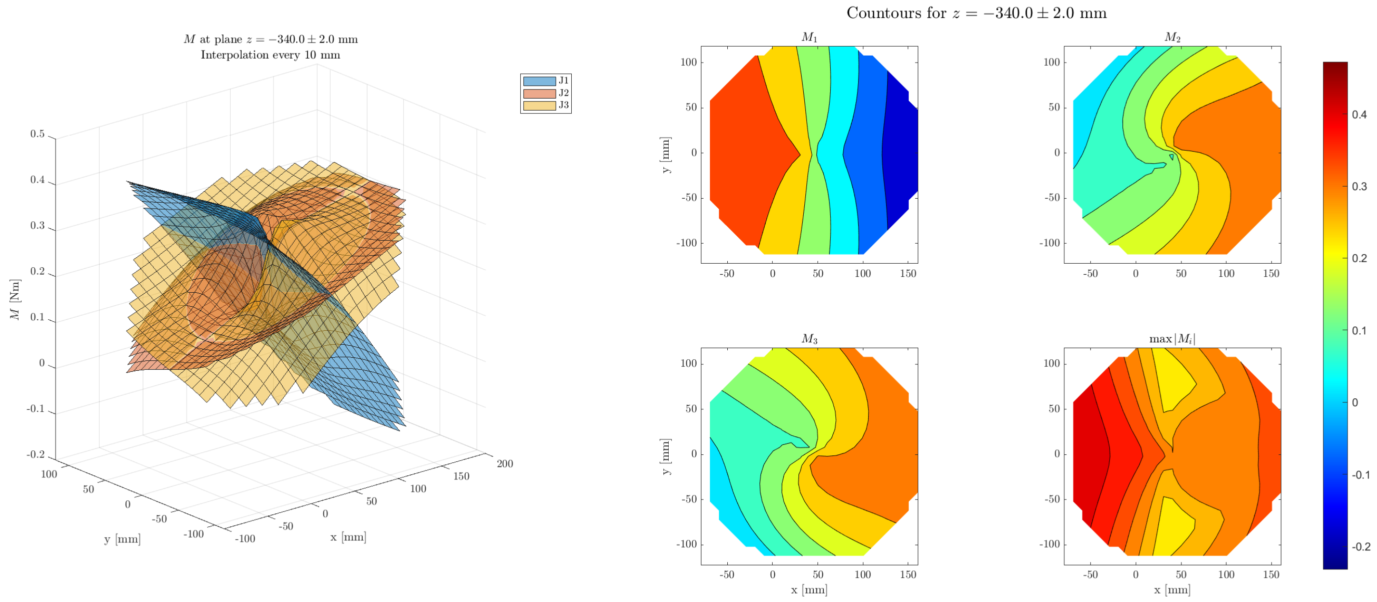
Figure B.9: Motor torque in response to a tangential unitary load when the end-effector is in the plane $x = 42.6$ mm.



(a) Torque curves

(b) Torque contours

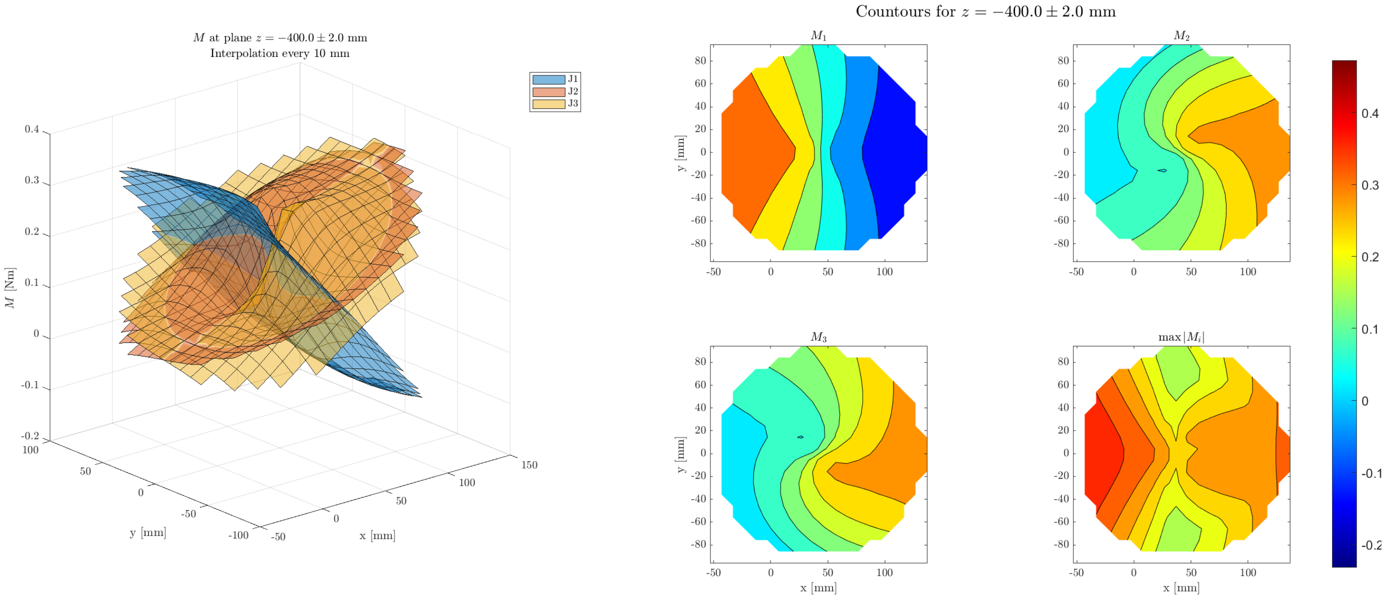
Figure B.10: Motor torque in response to a tangential unitary load when the end-effector is in the plane $y = 0$ mm.



(a) Torque curves

(b) Torque contours

Figure B.11: Motor torque in response to a tangential unitary load when the end-effector is in the plane $z = -340$ mm.



(a) Torque curves

(b) Torque contours

Figure B.12: Motor torque in response to a tangential unitary load when the end-effector is in the plane $z = -400$ mm.

### B.1.3. Vertical Load

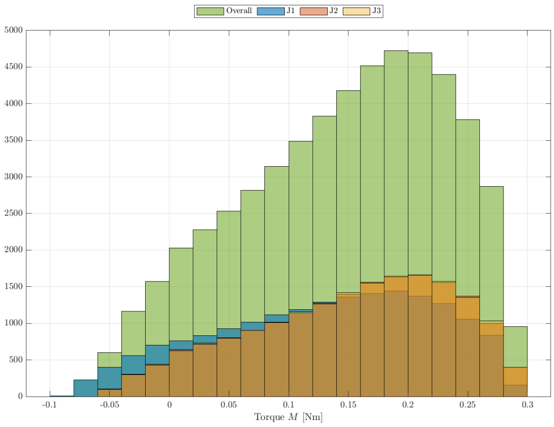


Figure B.13: Histogram of motor torques with vertical unitary load.

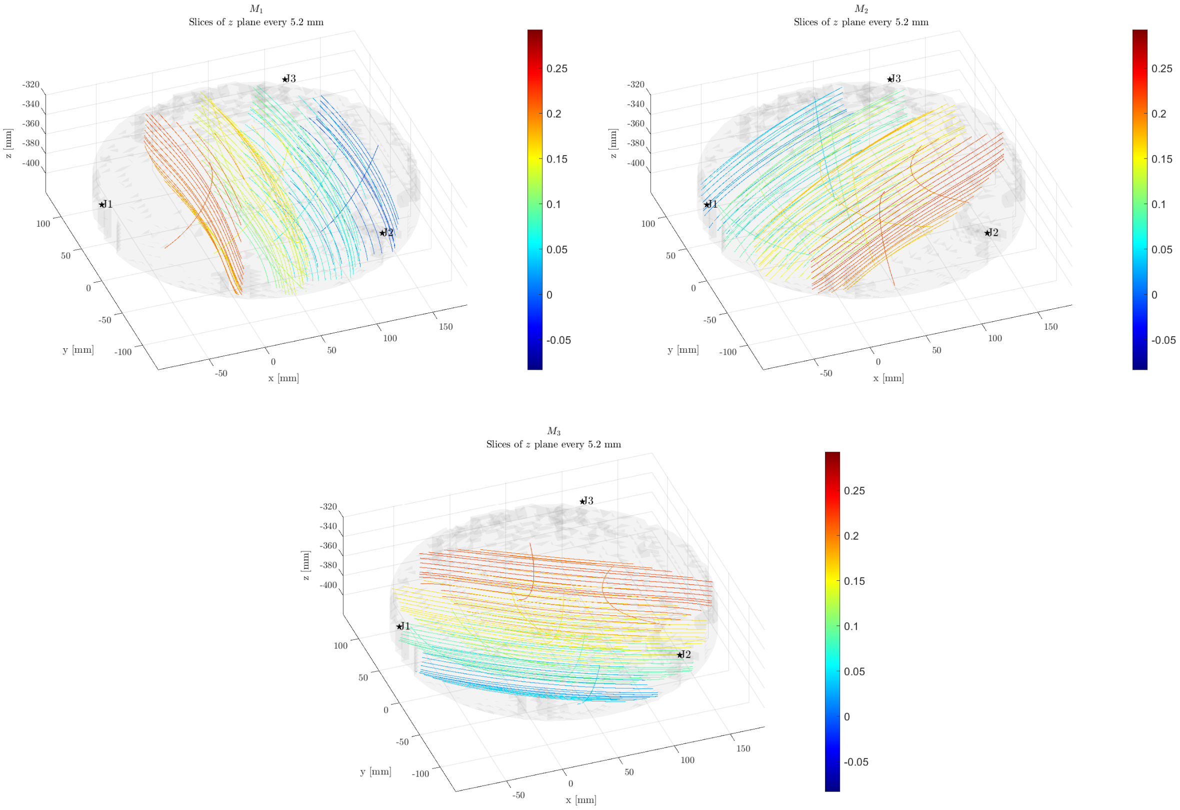
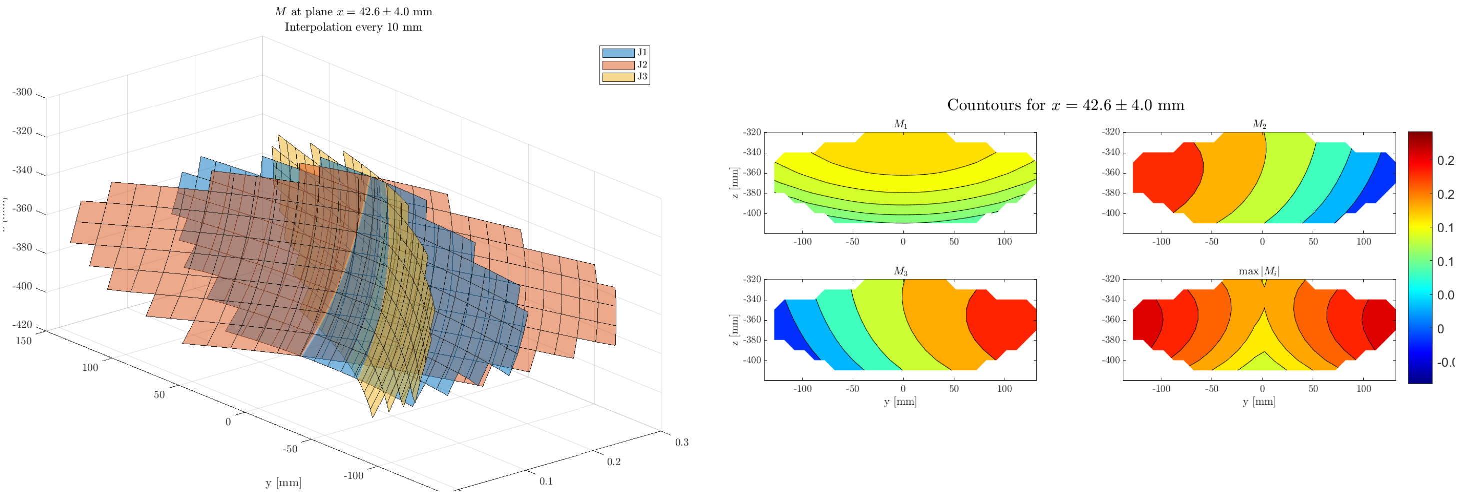


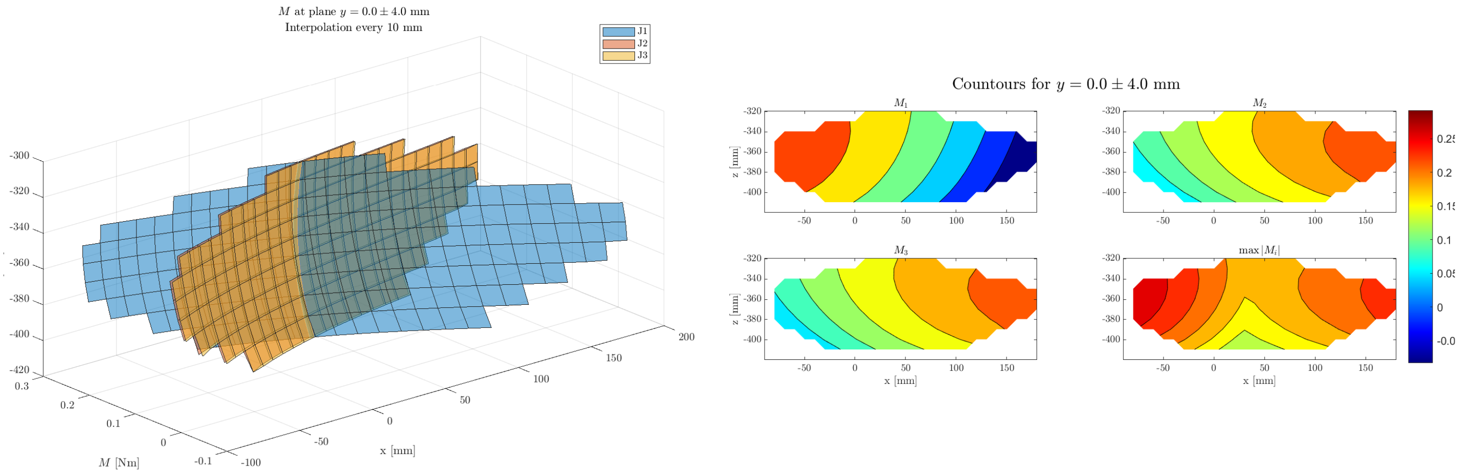
Figure B.14: Volumetric contour plot of torque in response to a vertical unitary load on the end-effector.



(a) Torque curves

(b) Torque contours

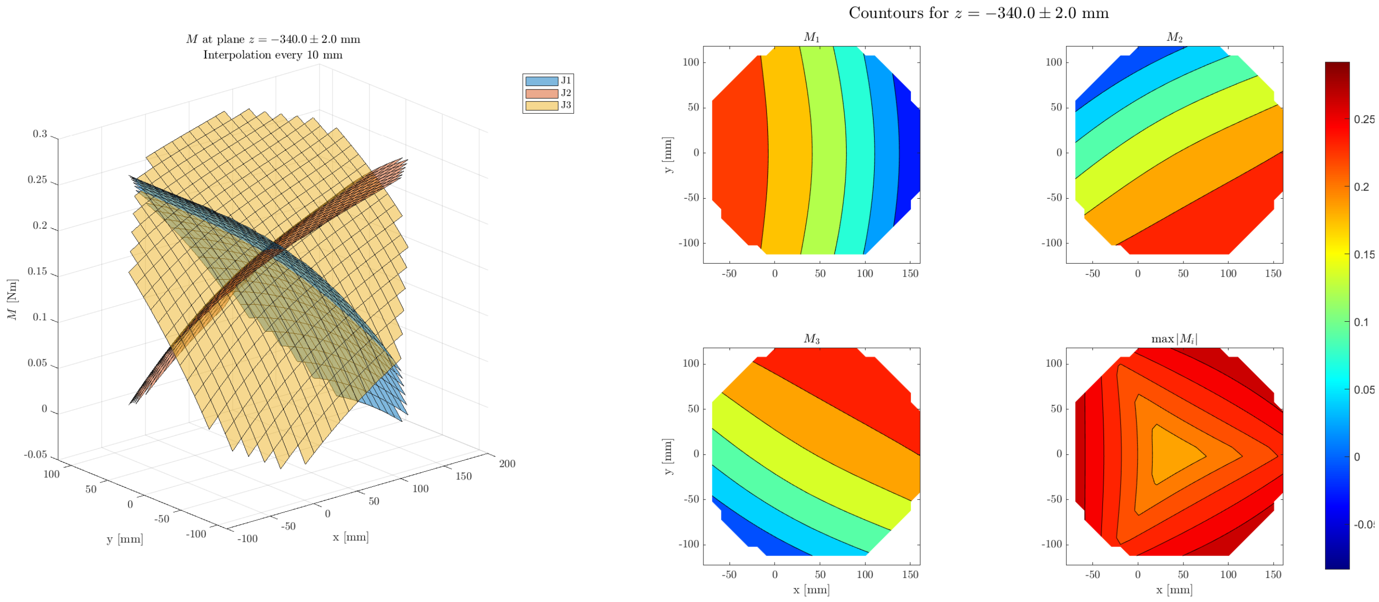
Figure B.15: Motor torque in response to a vertical unitary load when the end-effector is in the plane $x = 42.6$ mm.



(a) Torque curves

(b) Torque contours

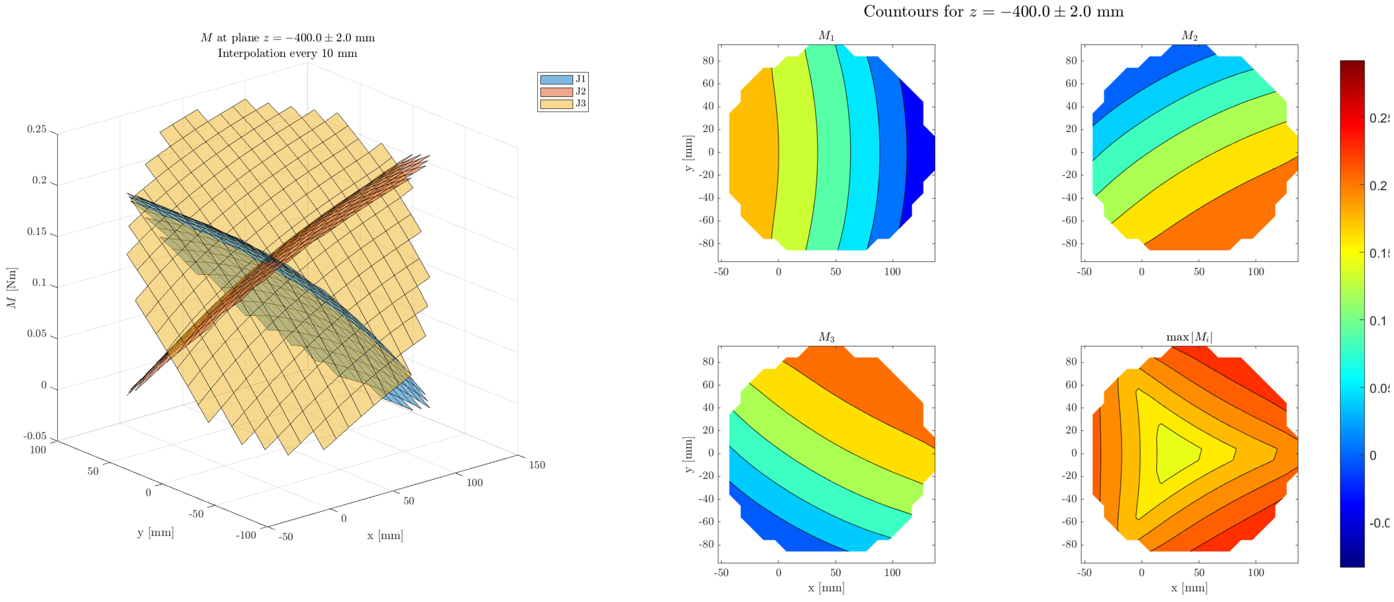
Figure B.16: Motor torque in response to a vertical unitary load when the end-effector is in the plane $y = 0$ mm.



(a) Torque curves

(b) Torque contours

Figure B.17: Motor torque in response to a vertical unitary load when the end-effector is in the plane $z = -340$ mm.



(a) Torque curves

(b) Torque contours

Figure B.18: Motor torque in response to a vertical unitary load when the end-effector is in the plane $z = -400$ mm.

## B.2. Structural Forces

### B.2.1. Radial Load

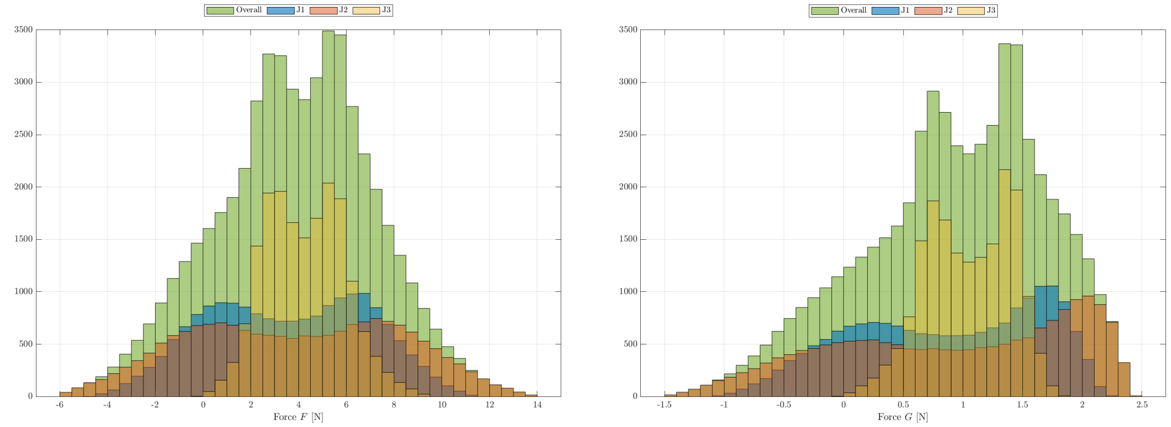


Figure B.19: Histogram of $F$ (left) and $G$ (right) with radial unitary load on the end-effector.

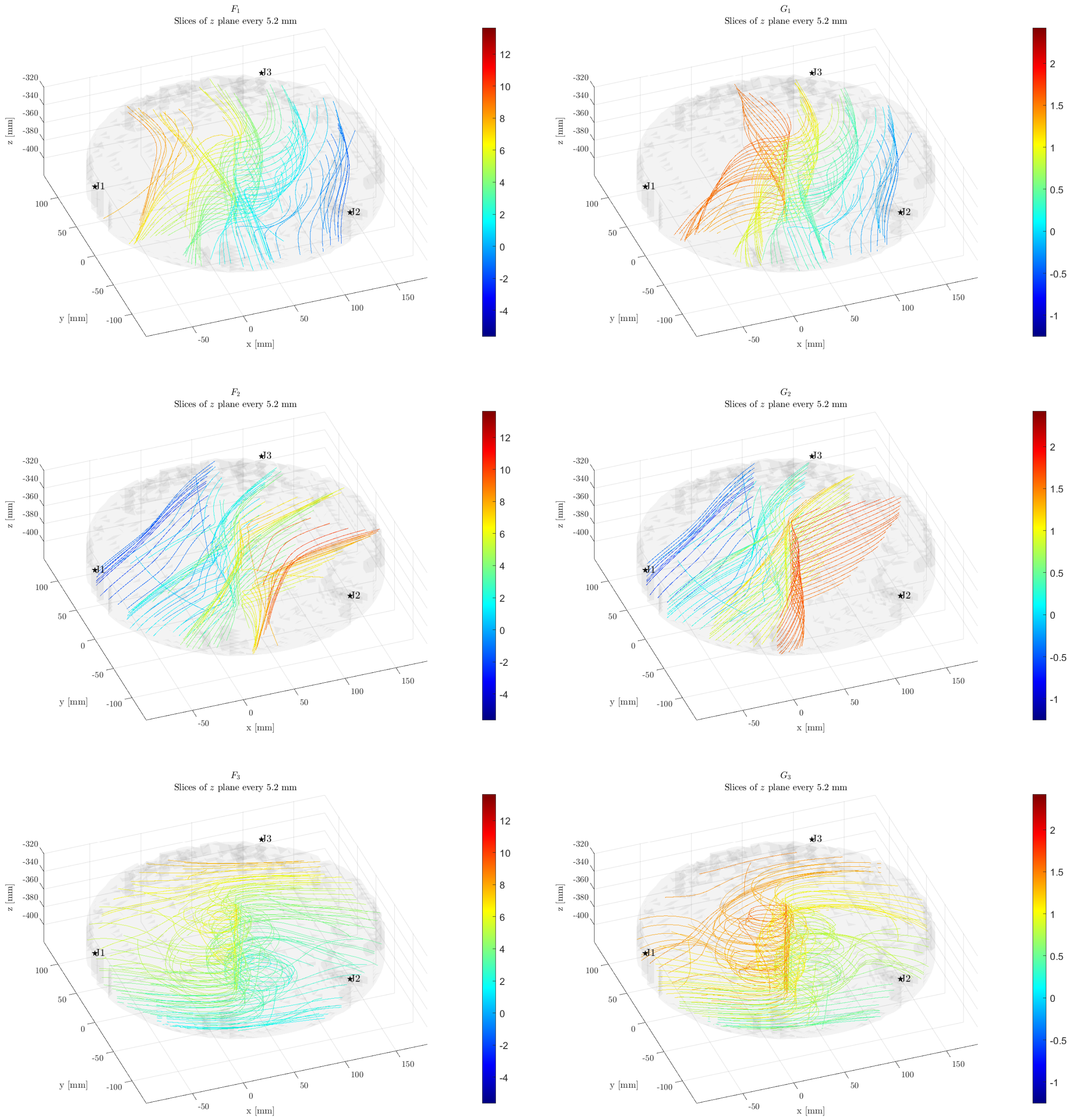
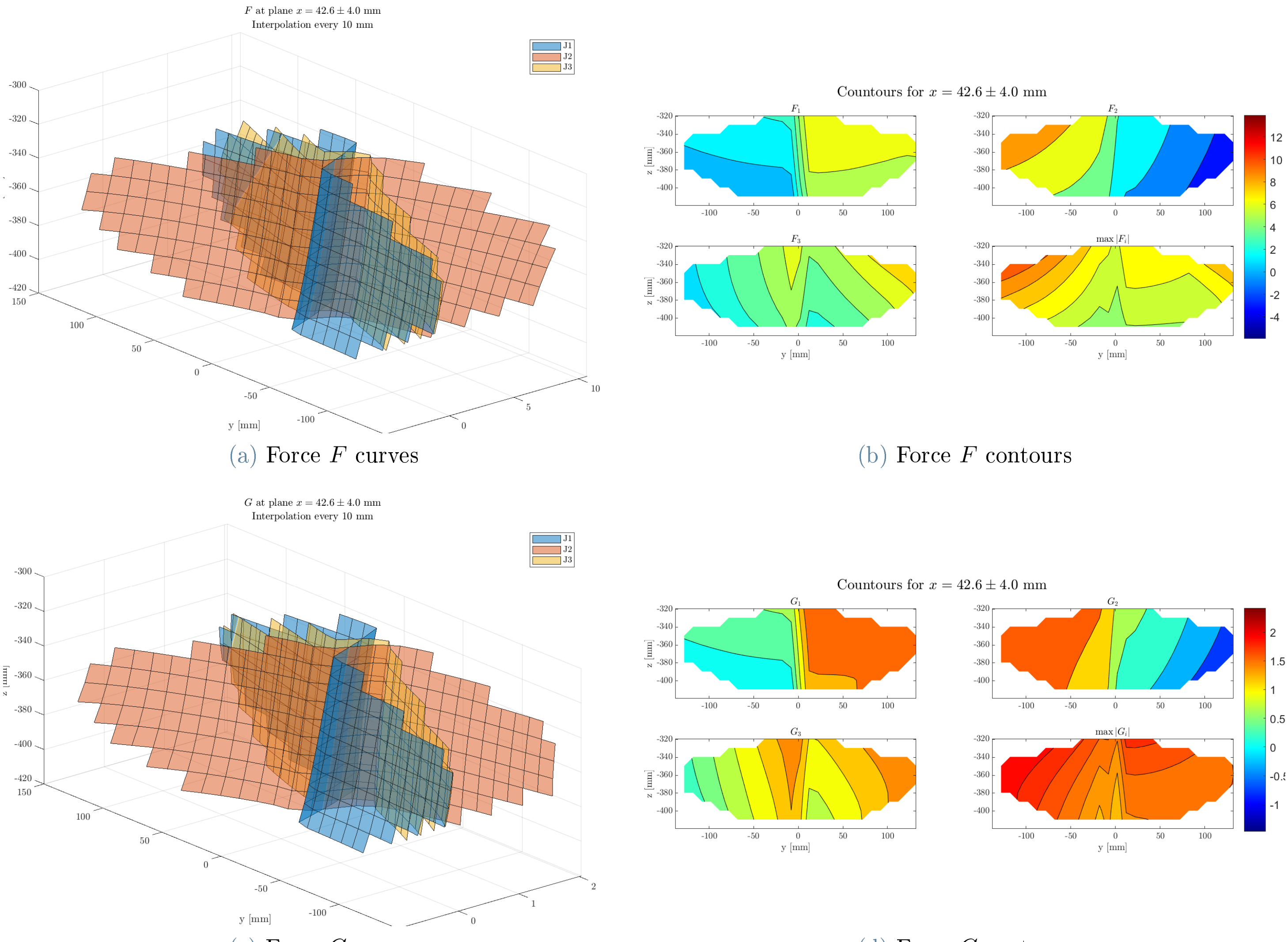


Figure B.20: Volumetric contour plot of $F$ (left) and $G$ (right) with a radial unitary load on the end-effector.



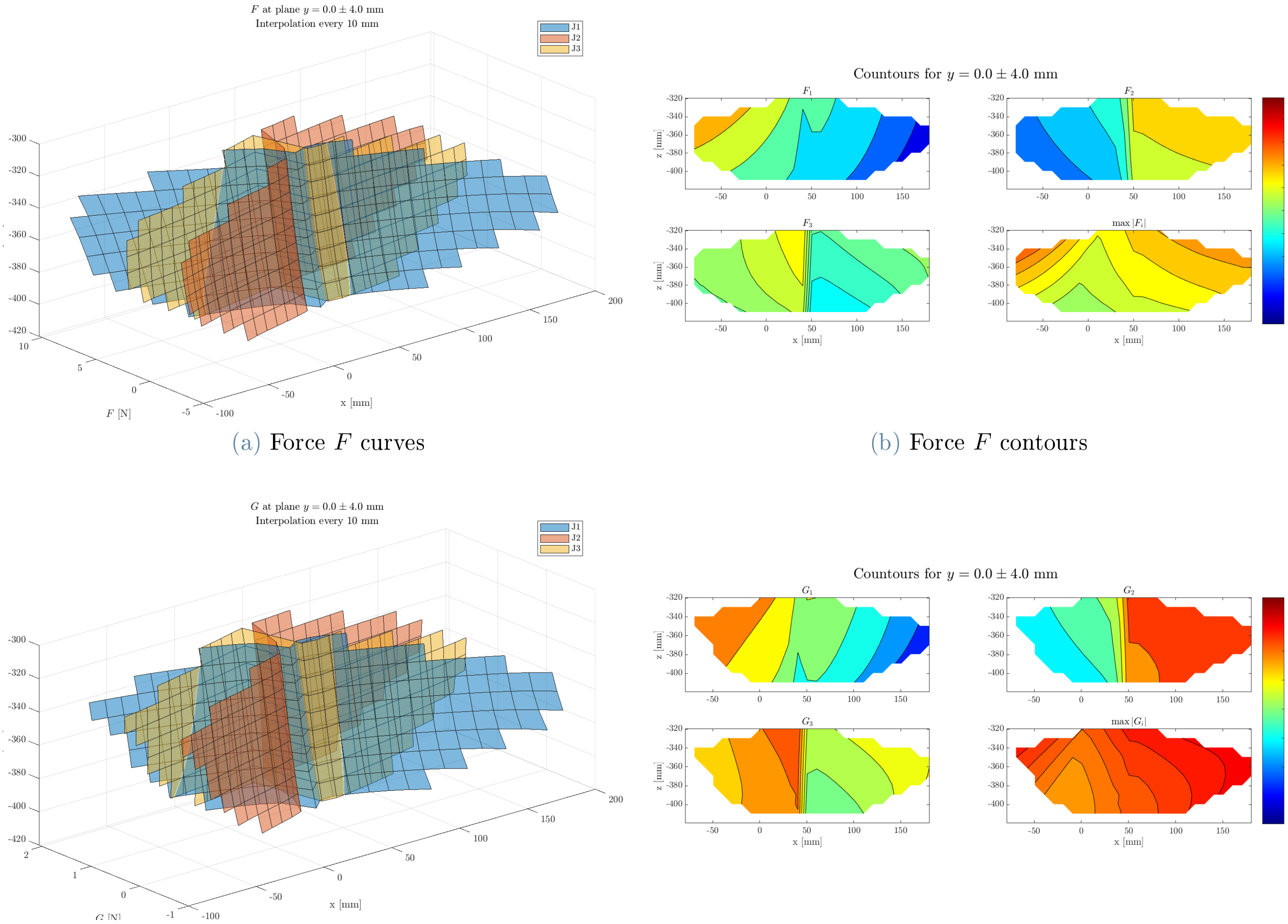
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.21: Forces $F$ and $G$ with a radial unitary load when the end-effector is in the plane $x = 42.6$ mm.



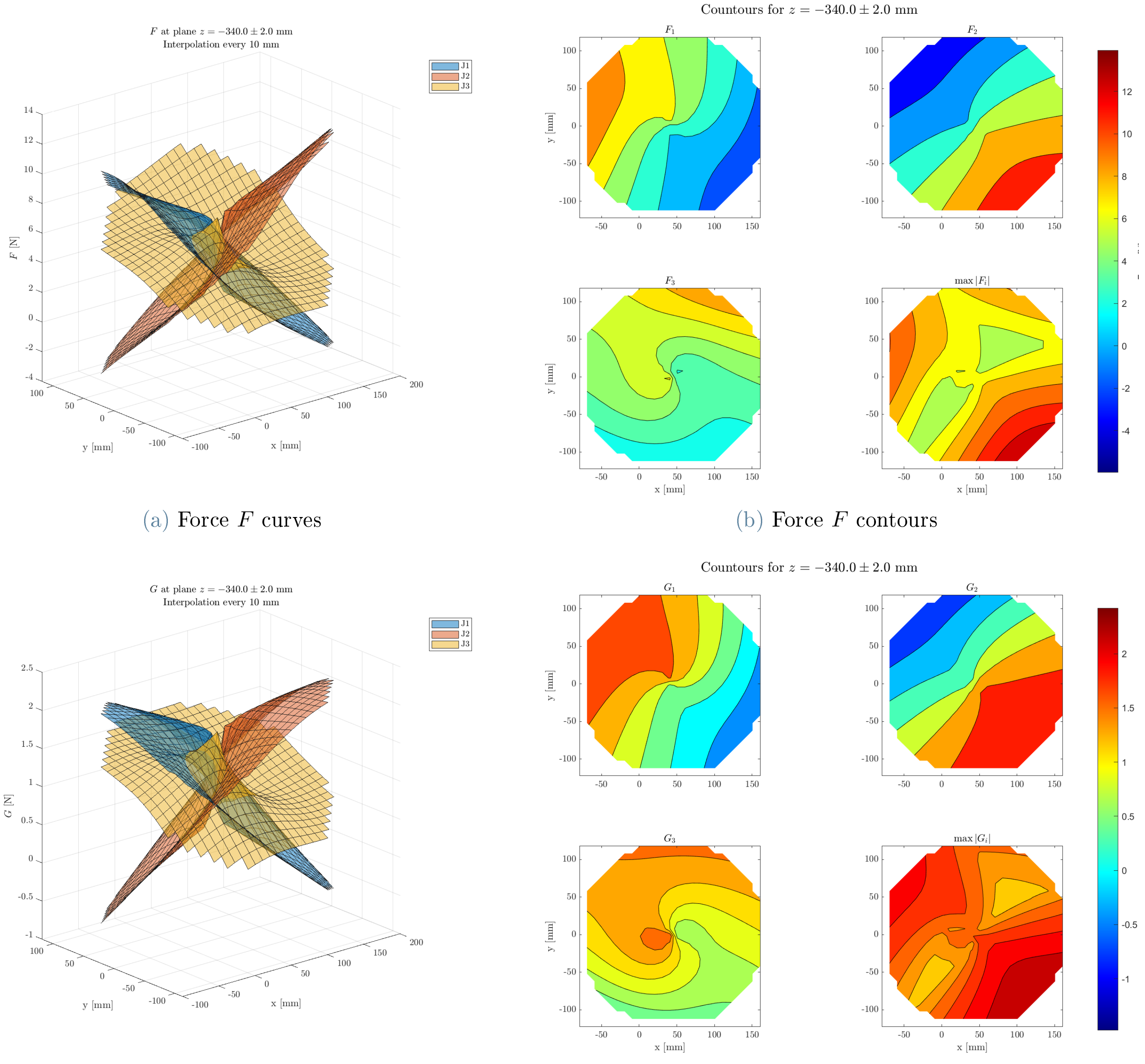
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.22: Forces $F$ and $G$ with a radial unitary load when the end-effector is in the plane $y = 0$ mm.



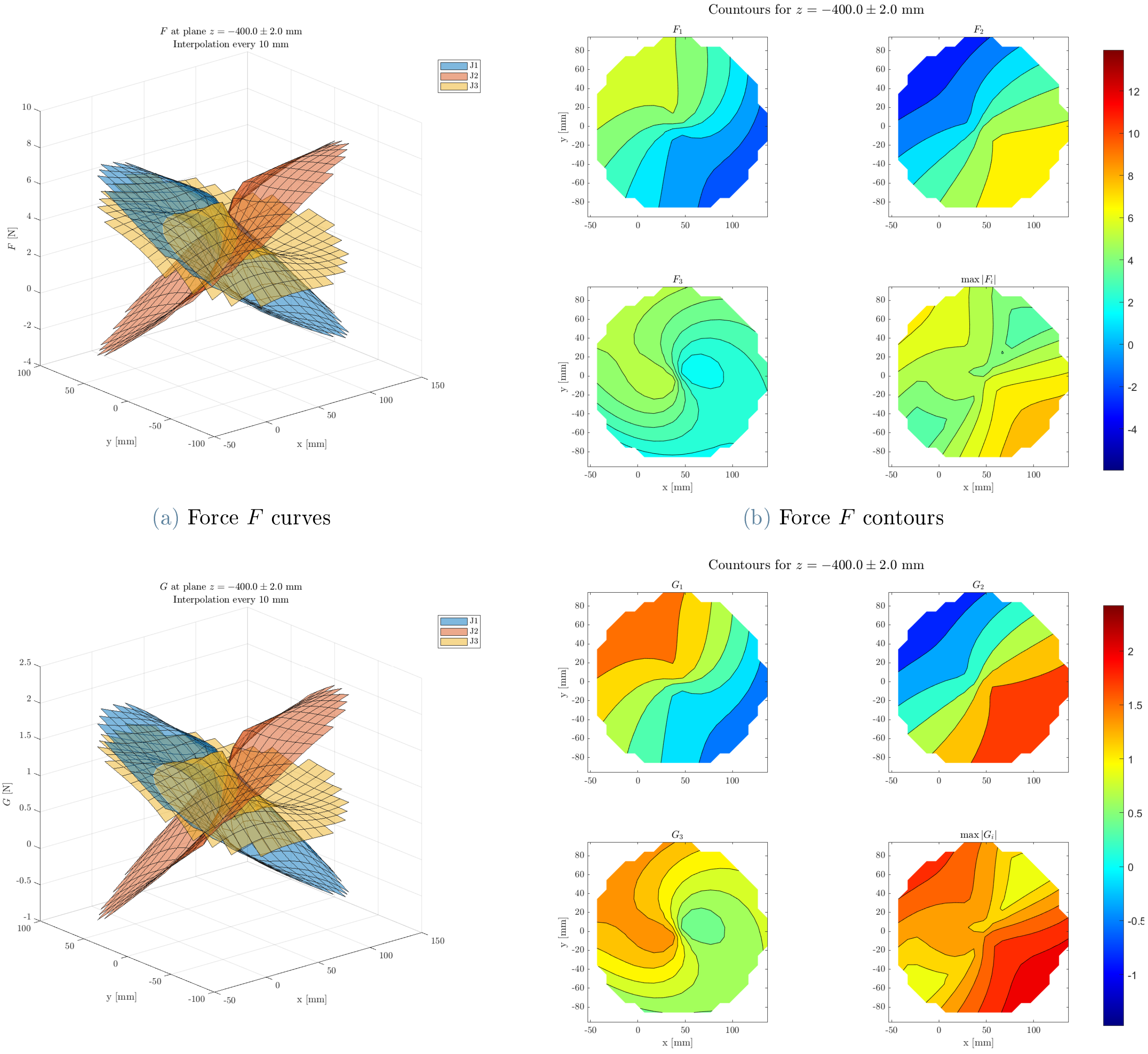
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.23: Forces $F$ and $G$ with a radial unitary load when the end-effector is in the plane $z = -340$ mm.



(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.24: Forces $F$ and $G$ with a radial unitary load when the end-effector is in the plane $z = -400$ mm.

### B.2.2. Tangential Load

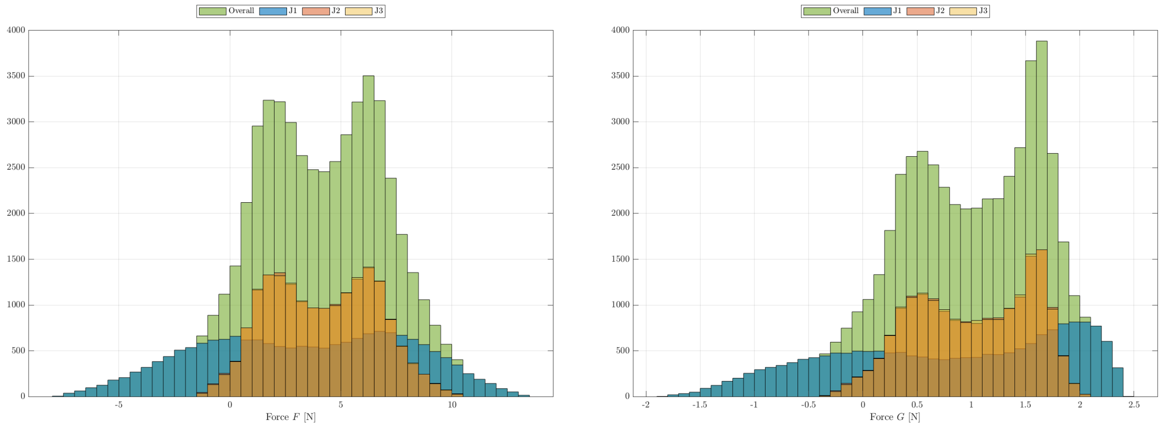


Figure B.25: Histogram of $F$ (left) and $G$ (right) with tangential unitary load on the end-effector.

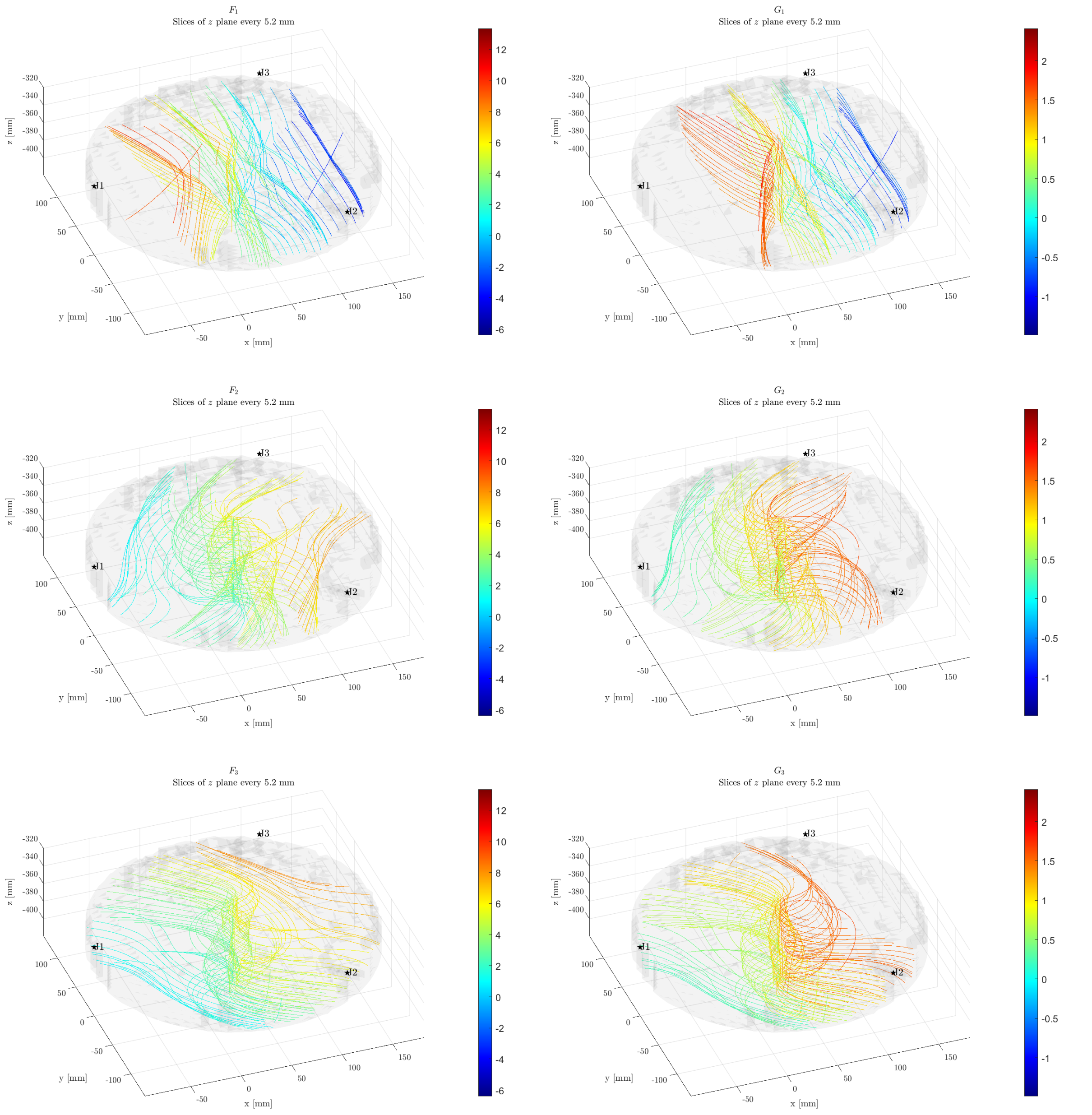
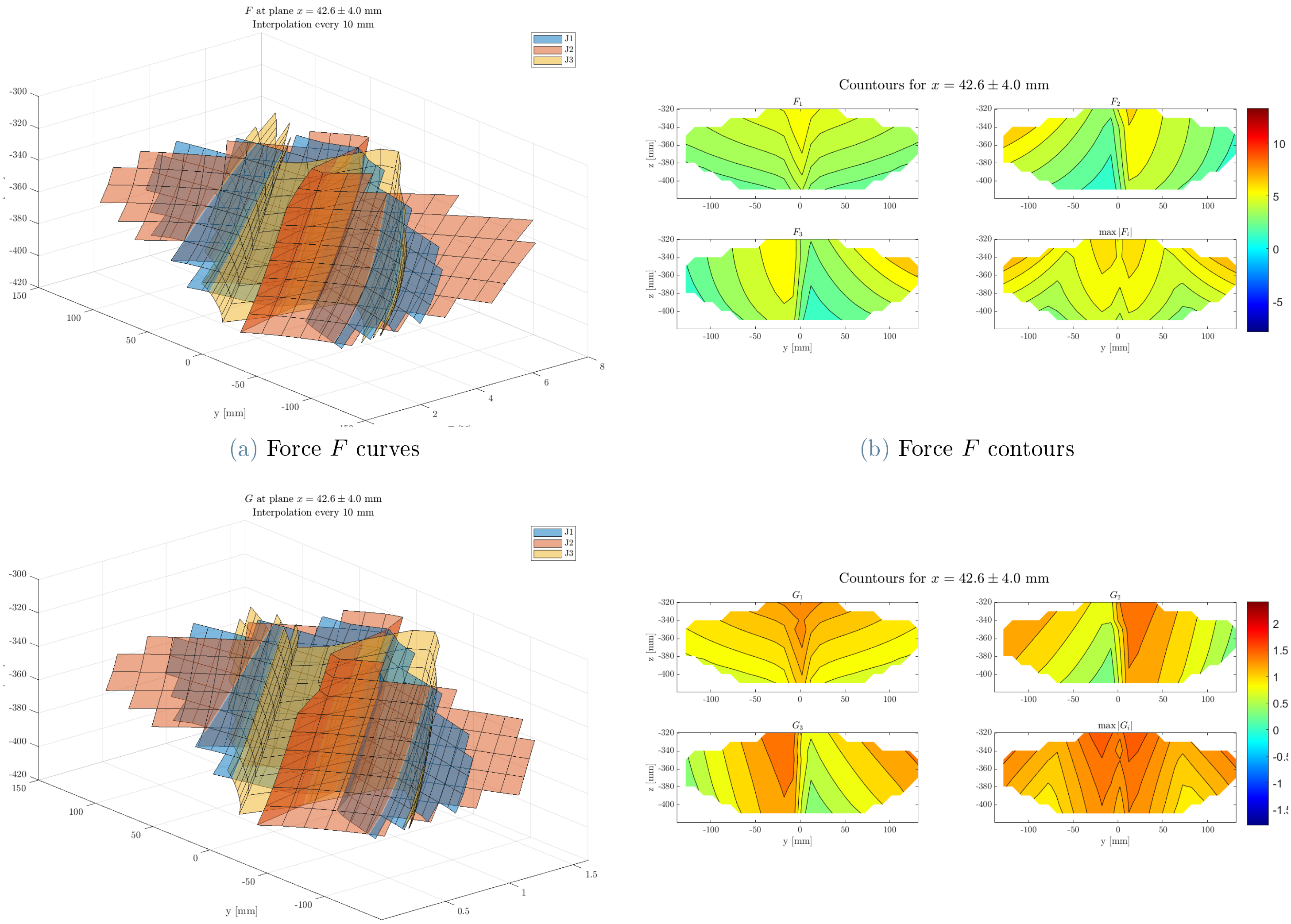


Figure B.26: Volumetric contour plot of $F$ (left) and $G$ (right) with a tangential unitary load on the end-effector.



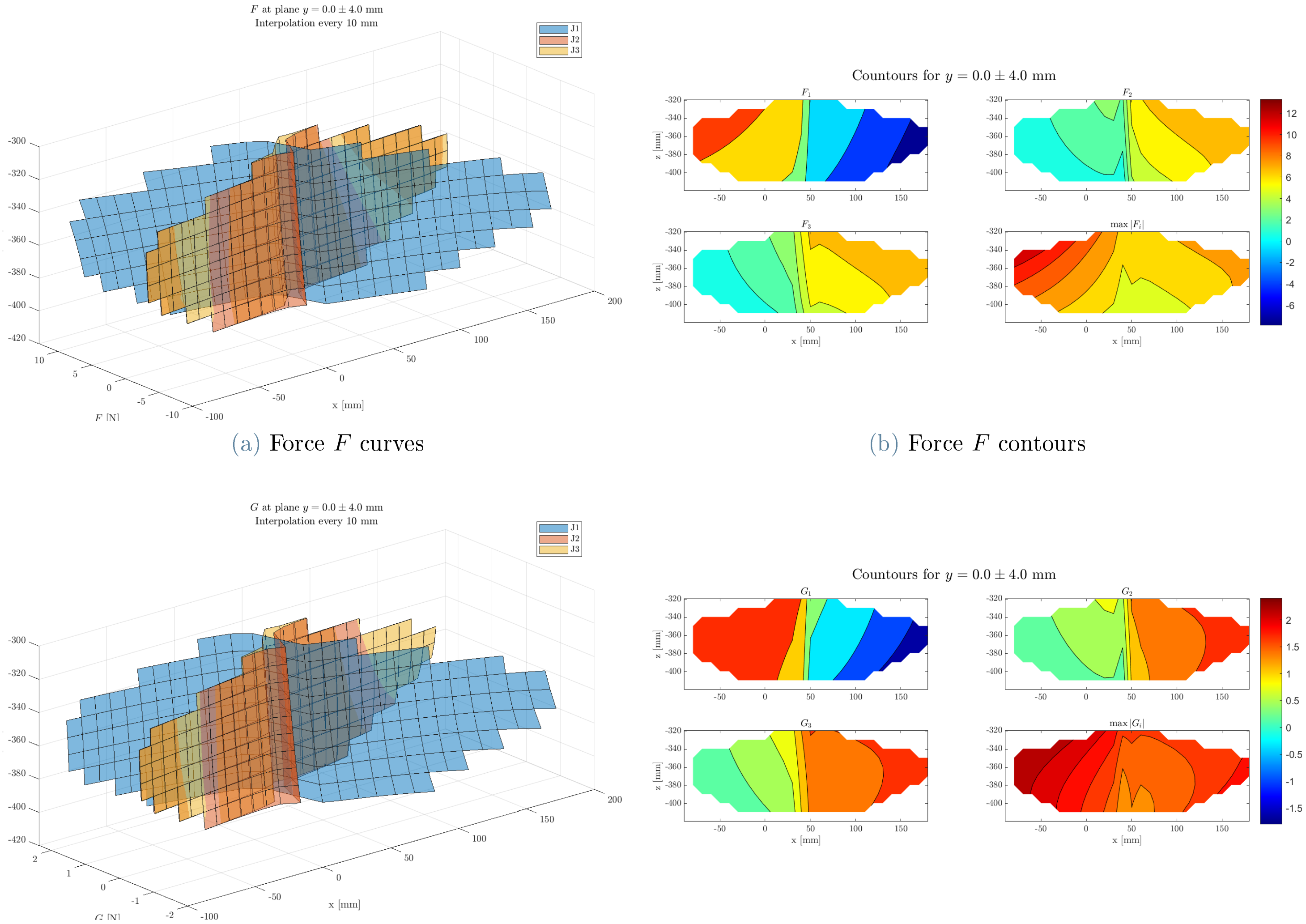
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.27: Forces $F$ and $G$ with a tangential unitary load when the end-effector is in the plane $x = 42.6$ mm.



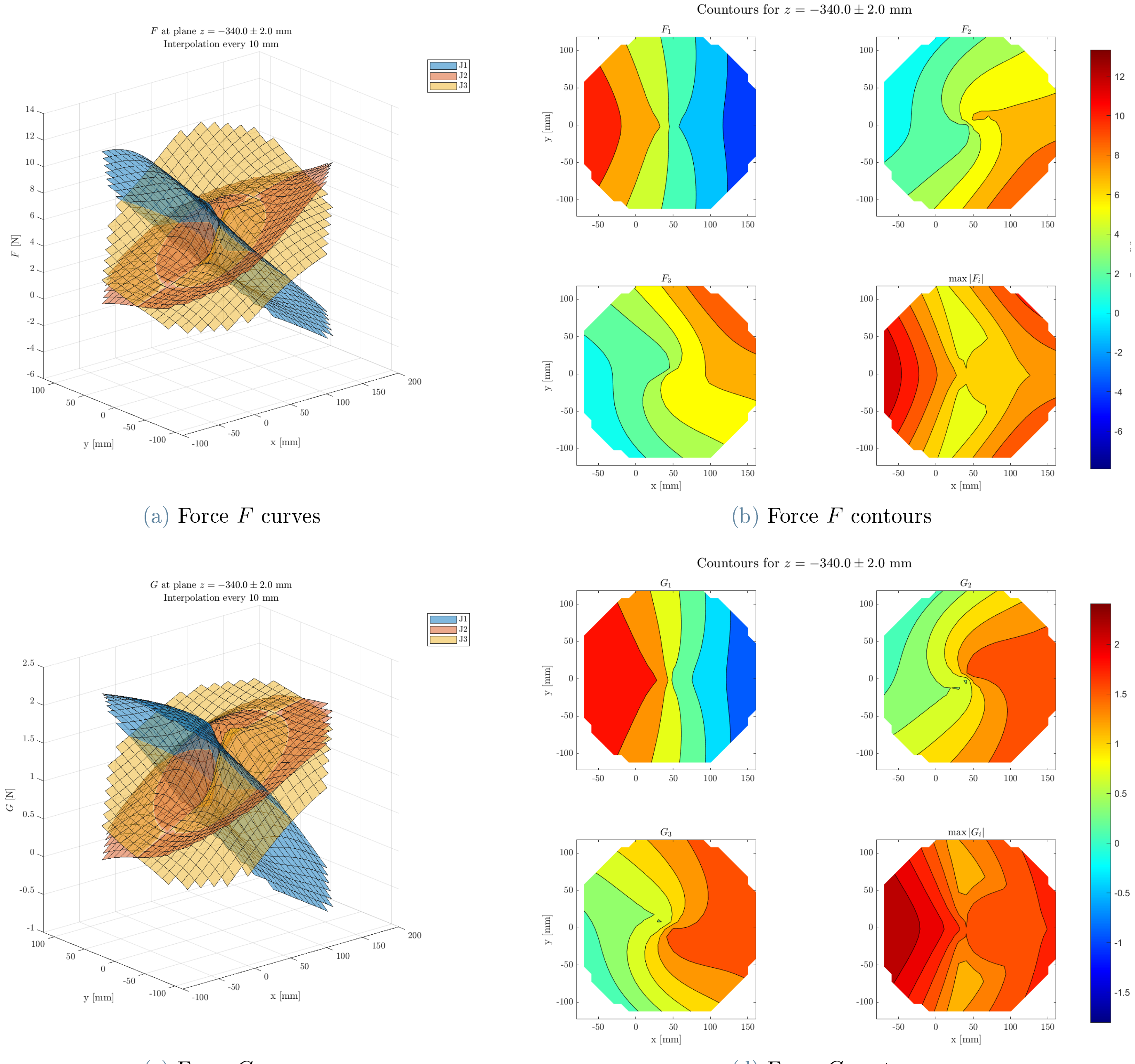
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.28: Forces $F$ and $G$ with a tangential unitary load when the end-effector is in the plane $y = 0$ mm.



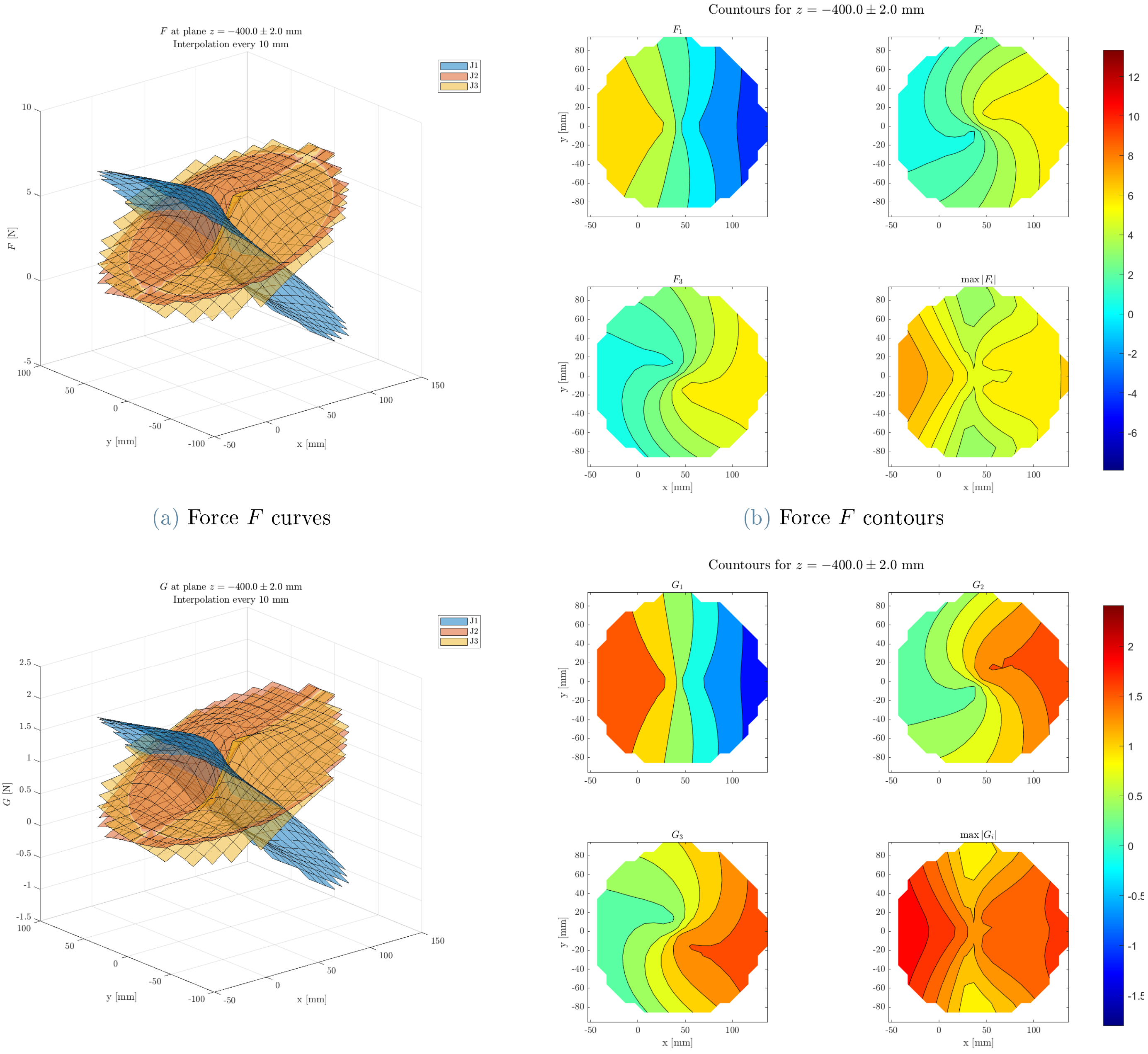
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.29: Forces $F$ and $G$ with a tangential unitary load when the end-effector is in the plane $z = -340$ mm.



(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.30: Forces $F$ and $G$ with a tangential unitary load when the end-effector is in the plane $z = -400$ mm.

### B.2.3. Vertical Load

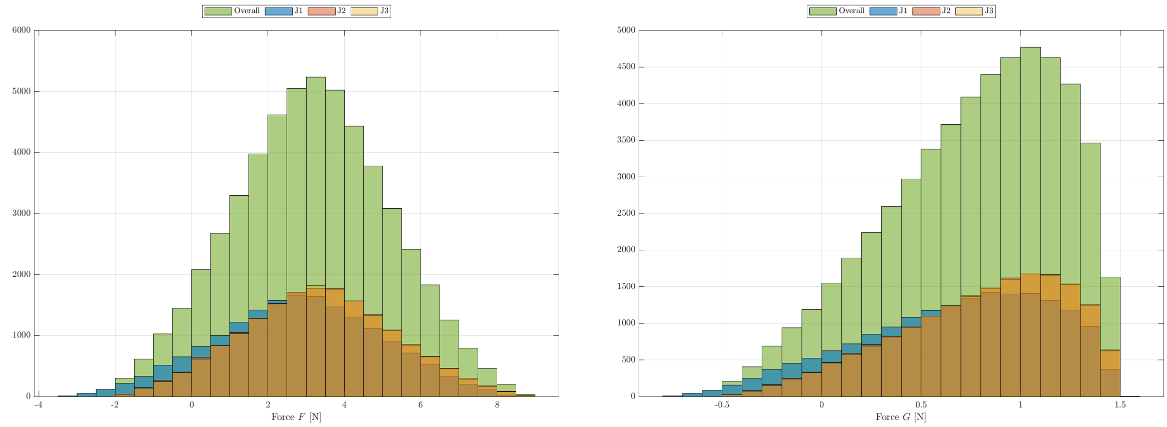


Figure B.31: Histogram of $F$ (left) and $G$ (right) with vertical unitary load on the end-effector.

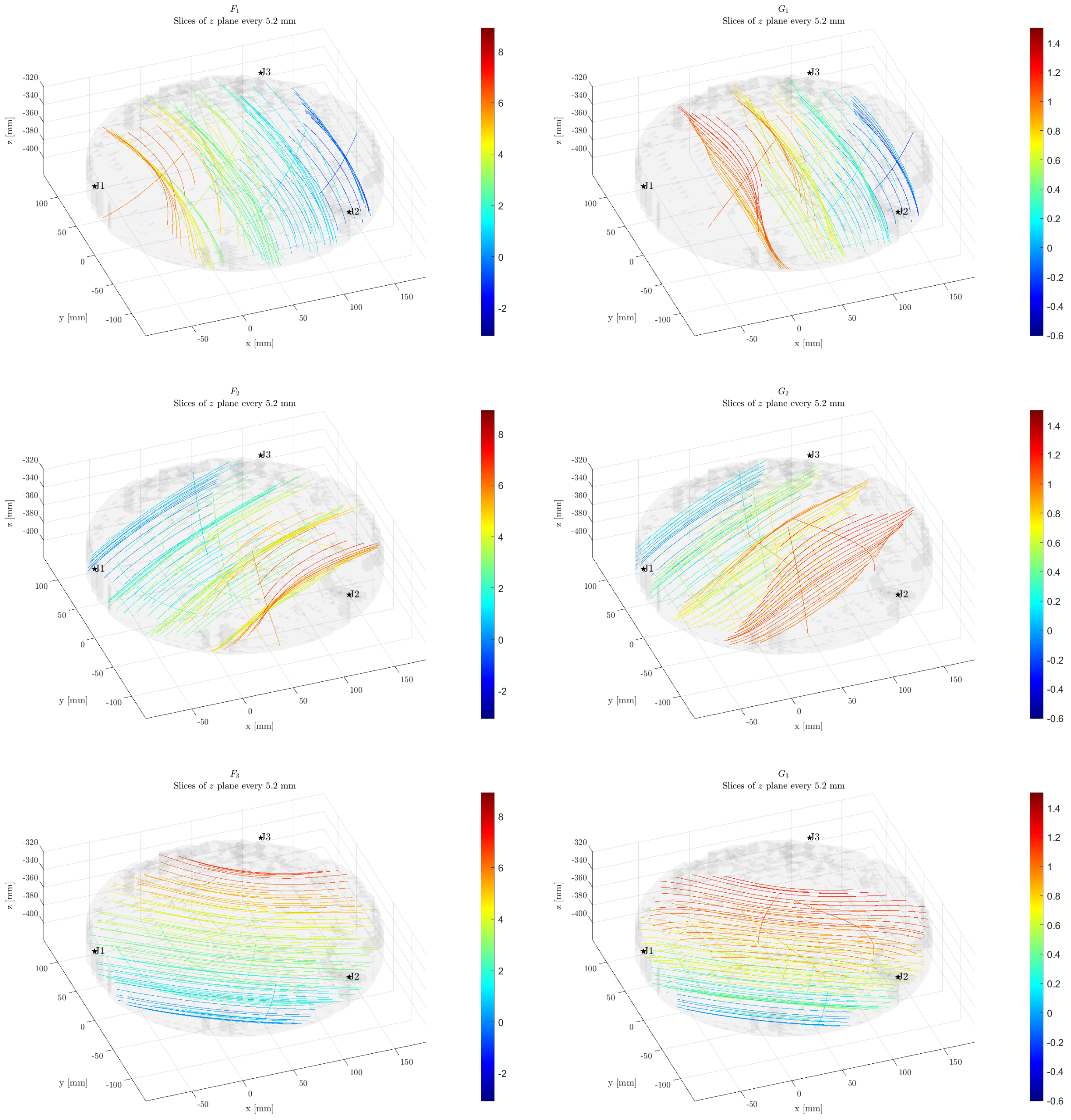
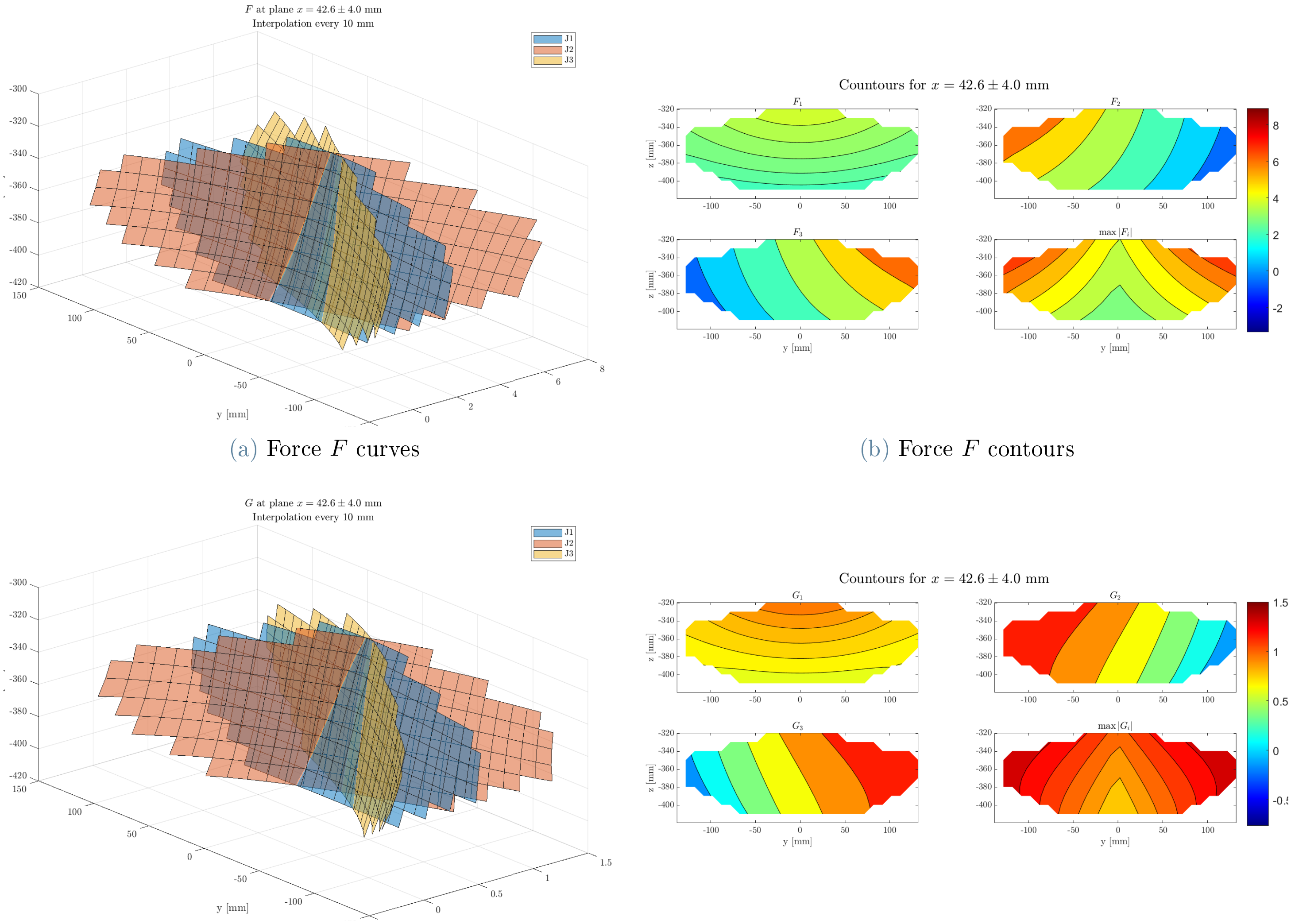


Figure B.32: Volumetric contour plot of $F$ (left) and $G$ (right) with a vertical unitary load on the end-effector.



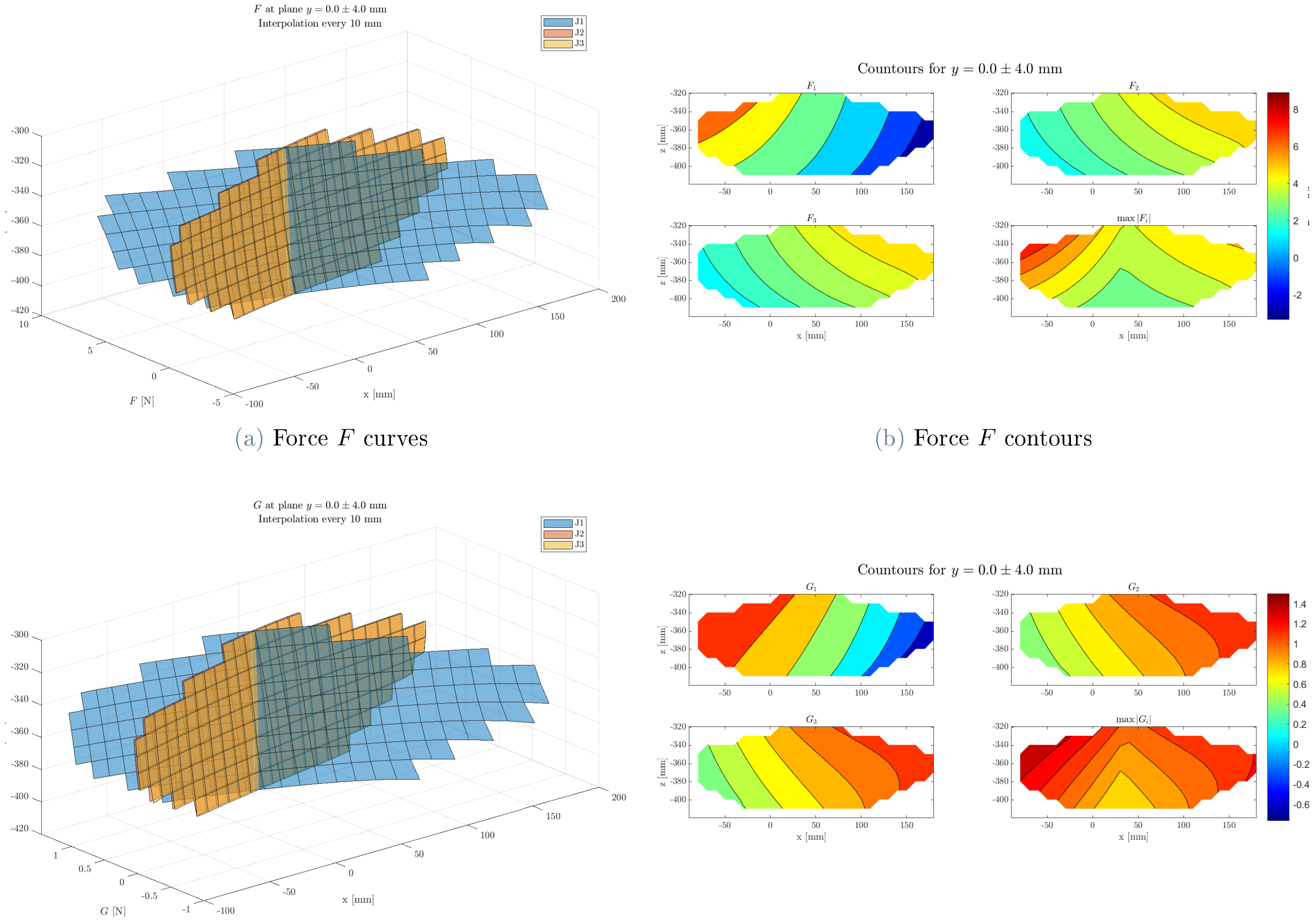
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.33: Forces $F$ and $G$ with a vertical unitary load when the end-effector is in the plane $x = 42.6$ mm.



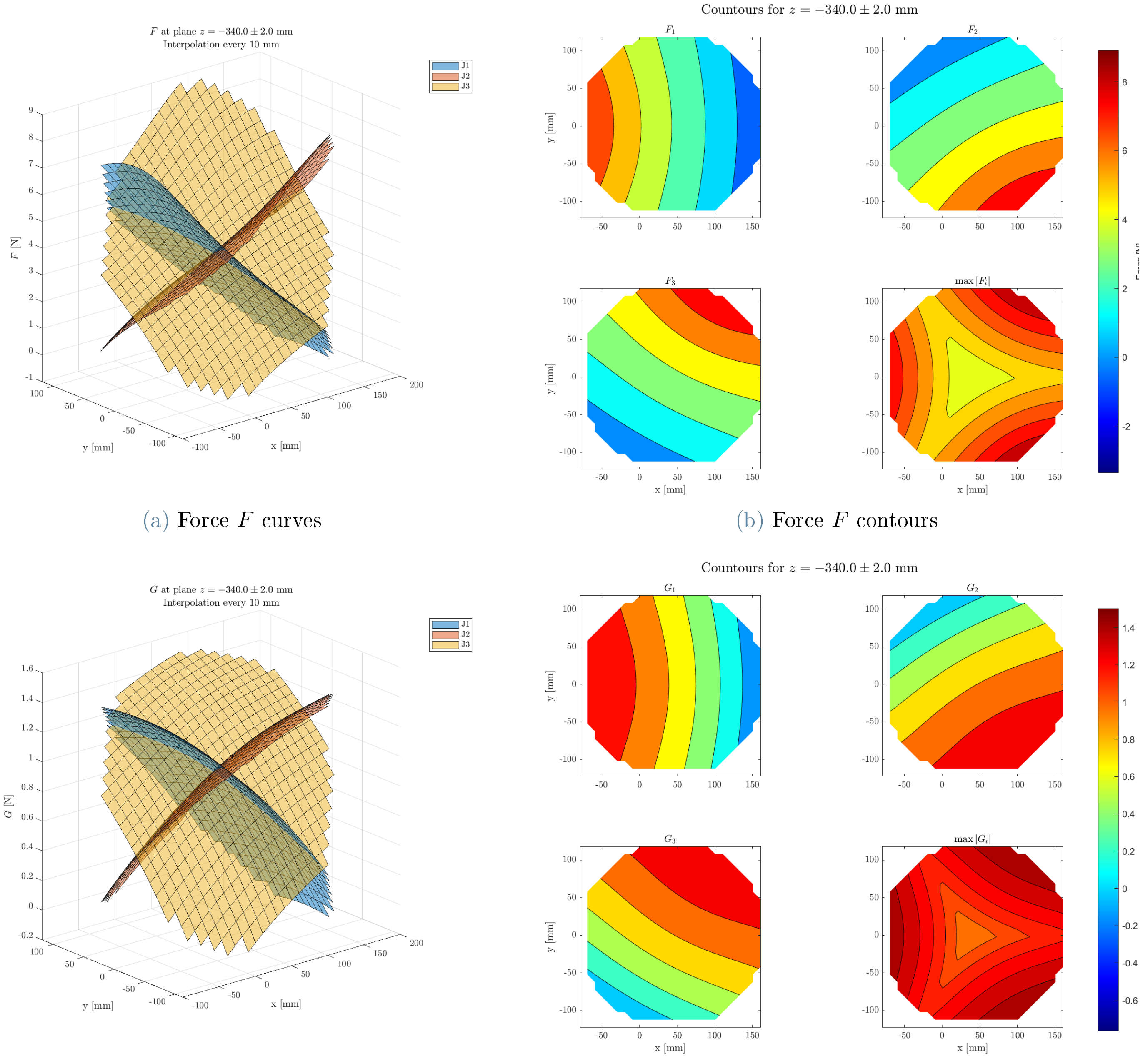
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.34: Forces $F$ and $G$ with a vertical unitary load when the end-effector is in the plane $y = 0$ mm.



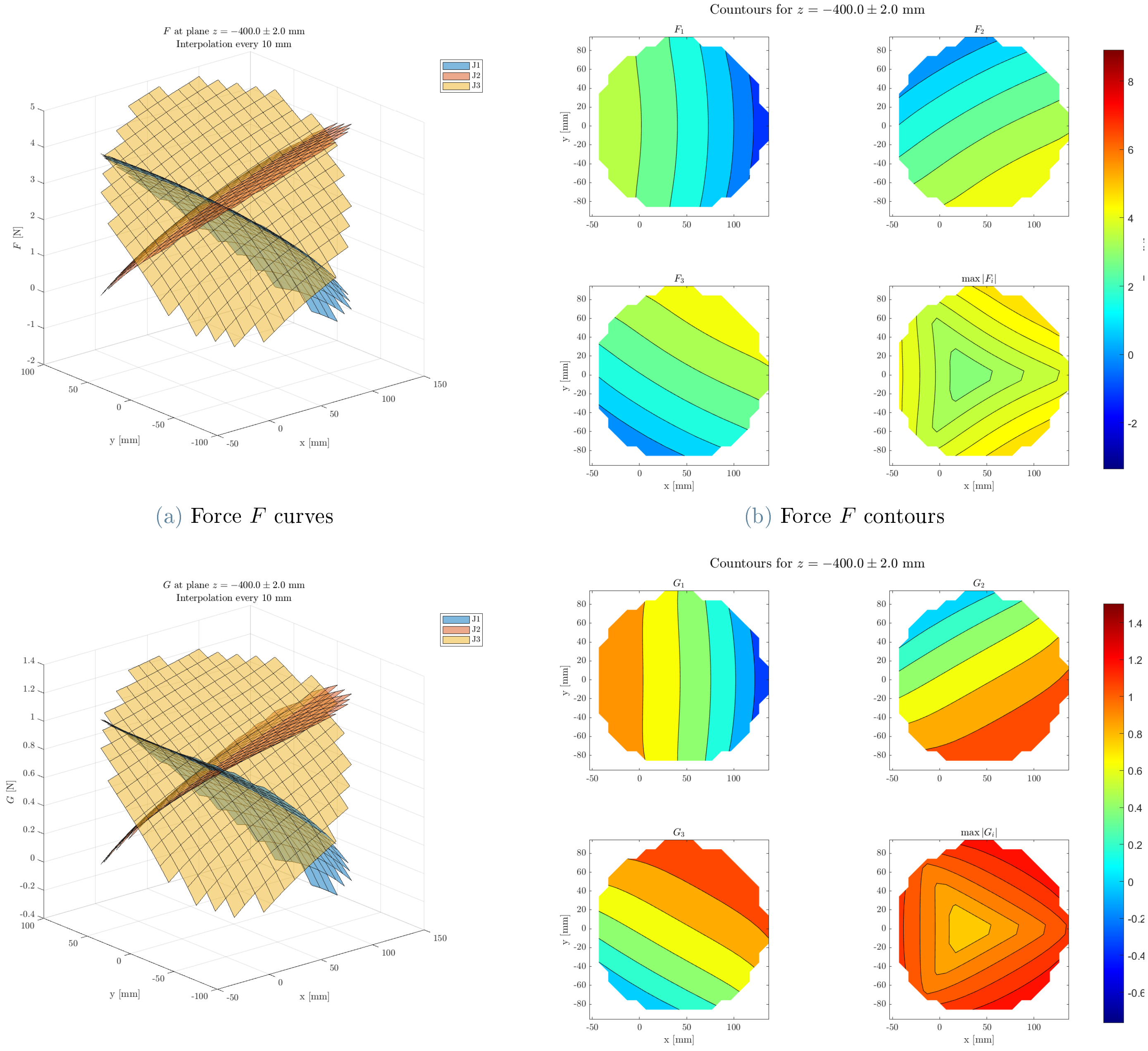
(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.35: Forces $F$ and $G$ with a vertical unitary load when the end-effector is in the plane $z = -340$ mm.



(a) Force $F$ curves

(b) Force $F$ contours

(c) Force $G$ curves

(d) Force $G$ contours

Figure B.36: Forces $F$ and $G$ with a vertical unitary load when the end-effector is in the plane $z = -400$ mm.

# List of Figures

# List of Tables

# Acknowledgements

I wish to pay my gratitude to Prof. Vladimír Bulej for accepting to supervise my thesis and proposing an interesting and rich topic despite being burdened with work. I must also thank the doctoral student Ing. Michal Bartoš, whose excellent CAD drawing of the M-1$i$A/0.5A was fundamental.

A thank-you to the Erasmus coordinators of Polimi and Uniza: Ms. Simona Benvenuto, Ms. Renata Janovčíková, and their co-workers. They supported every step of my mobility and provided valuable help to my decision to write the master thesis during an exchange period.

And finally, I am obliged to the Eramus+ student exchange project for allowing me, as well as other millions of student, experience education in another European country.