# TPK4170 - Robotics

2021-12-09

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

This report aims to present the work from the project assignment in TPK4170 Robotics. It contains the presentation and comparison of the forward kinematics computation for an open-chain manipulator, using the original Denavit-Hartenberg (DH) convention from the Siciliano et al [1], the modified version as presented in Modern Robotics [2], and the Power of Exponentials (PoE) formulation. Further, the original DH convention and the PoE formulation is used to calculate the forward kineamtics of the KUKA KR6 R900 sixx (Agilus) robot. The analytical and numerical method for finding the inverse kinematics are also presented and used for the KUKA robot. In the end, a singularity analysis is performed to find kinematic singularities and in which configurations they appear.

# Chapter 2

# Task 1

## 2.1 Task 1.1

### 2.1.1 Parameters

DH makes use of only 4 variables to describe the pose of each link. These are illustrated in figure 2.2.

- $d_i$ = the distance from $O_{i'}$ to $O_i$ along the $z_{i-1}$ axis of $O_{i-1}$
- $\vartheta_i$ = the counterclockwise angle between $x_{i-1}$ and $x_i$ about the $z_{i-1}$ axis
- $a_i$ = the distance between $O_{i'}$ and $O_i$
- $\alpha_i$ = the counterclockwise angle between $z_{i-1}$ and $z_i$ about the $x_i$ axis

$a_i$ and $\alpha_i$ are always constant and dependent on link $i$´s geometry. Depending on what kind of joint $i$ is, one of the remaining variables will be a constant and the other considered a free variable for the user to decide.

- Revolving joint: $\vartheta_i$ is the free variable.
- Prismatic joint: $d_i$ is the free variable.

### 2.1.2 Transformation matrix

To describe a single change in position and orientation of a coordinate frame, we make use of the *homogeneous transformation matrix*:

$$\mathbf{A}_i^{i-1} = \begin{bmatrix} \mathbf{R}_i^{i-1} & \mathbf{o}_i^{i-1} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{2.1}$$

Here $\mathbf{R}_i^{i-1}$ is the quadratic rotation matrix describing the change in orientation from frame $i-1$ to $i$, $\mathbf{o}_i^{i-1}$ is a column vector describing the translation, and $\mathbf{0}$ is a zero-vector of same dimension as $\mathbf{o}$.

The transformation $A_i^{i-1}$ consists of two steps; from $i-1$ to $i'$ and then from $i'$ to $i$. This is visualised in figure 2.2. First step involves a translation $d_i$ along- and a rotation $\vartheta_i$ about the $z_{i-1}$ axis, moving from frame $i-1$ to $i'$.

$$\mathbf{A}_{i'}^{i-1} = \begin{bmatrix} c_{\vartheta_i} & -s_{\vartheta_i} & 0 & 0 \\ s_{\vartheta_i} & c_{\vartheta_i} & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

Then, the second step is a translation $a_i$ along- and a rotation $\alpha_i$ about the $x_{i'}$ axis, moving from frame $i'$ to $i$

$$\mathbf{A}_{i}^{i'} = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

By cancellation of subscript by superscript we get

$$\mathbf{A}_{i}^{i-1}(q_i) = \mathbf{A}_{i'}^{i-1}\mathbf{A}_{i}^{i'} = \begin{bmatrix} c_{\vartheta_i} & -s_{\vartheta_i}c_{\alpha_i} & s_{\vartheta_i}s_{\alpha_i} & \alpha_i c_{\vartheta_i} \\ s_{\vartheta_i} & c_{\vartheta_i}c_{\alpha_i} & -c_{\vartheta_i}s_{\alpha_i} & \alpha_i s_{\vartheta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

where $q_i$ represents the free variable, either $\vartheta_i$ or $d_i$.

### 2.1.3 Direct kinematic

In order to describe the kinematics of the end effector of an open-chain manipulator with $n$ links, an expression for the positioning of the end-effector frame with respect to the space frame {s} is constructed.

$$\mathbf{T}_e^s(\mathbf{q}) = \mathbf{T}_0^s\mathbf{T}_n^0\mathbf{T}_e^n \tag{2.5}$$

$\mathbf{T}_e^s$ is a homogeneous transformation matrix, denoted by $\mathbf{T}$ to distinguish it from the individual link transformations $\mathbf{A}$. $\mathbf{T}_0^s$ describes orientation and position of the frame of the first link with respect to the base frame and $\mathbf{T}_e^n$ the end-effector frame with respect to the the frame of link $n$. $\mathbf{T}_n^0$, as with the others, describes the orientation and position of frame 0 to frame $n$ and is a function of each link parameter.

$$\mathbf{T}_n^0(\mathbf{q}) = \mathbf{A}_1^0(q_1)\mathbf{A}_0^1(q_2)...\mathbf{A}_n^{n-1}(q_n) \tag{2.6}$$

Here, $\mathbf{q}$ is a vector containing the free variables of each link. By inspection of the open-chain manipulator one can create a table of parameters for each $A_i$ as illustrated in 2.1

| Link | $a_i$ | $\alpha_i$ | $d_i$ | $\vartheta_i$ |
|---|---|---|---|---|
| 1 | $a_1$ | $\alpha_1$ | $d_1$ | $\vartheta_1$ |
| 2 | $a_2$ | $\alpha_2$ | $d_2$ | $\vartheta_2$ |
| ... | ... | ... | ... | ... |
| n | $a_n$ | $\alpha_n$ | $d_n$ | $\vartheta_n$ |

**Table 2.1:** Illustration of DH parameter table

## 2.2 Task 1.2

The difference between the original convention as presented in Siciliano and the modified convention presented in Modern Robotics, comes down to the choice of z axis and the positioning of the frame. The difference can be seen by comparing figure 2.1 and figure 2.2. Given a link $i$, joining joint $i$ and $i+1$, the original version chooses the $z_i$ axis along the $i+1$ joint axis, whereas the modified version chooses its $z_i$ axis along the $i$ joint axis. The modified version keeps the frame along joint axis i, while the original version translates the frame along the common normal onto joint axis $i+1$. The $x_i$ axis has the same orientation in both cases, colinear with the common normal, but the $y_i$ axis will differ because of the different choice of $z_i$.

The parameters $\alpha_i$, $a_i$, $d_i$ and $\phi_i$, will stay the same for both conventions:

- $\alpha_i$ is defined as the angle between joint axis $i$ and joint axis i+1, about $x_i$. Since $x_i$ is the same in both conventions, $\alpha_i$ will be the same.
- The distance $a_i$ corresponds to the length of the common normal between joint axis i and joint axis $i+1$, along the $x_i$ axis.
- $d_i$ corresponds to the distance from the common normal between joint $i-1$ and joint $i$, and the common normal between joint i and joint i+1, along joint axis i.
- $\phi_i$, or $\vartheta_i$, is the angle between $x_{i-1}$ and $x_i$, about joint axis $i$.

With the difference in positioning and orientation as described, the order of operations for calculating the relation between two frames will be different.

The original formulation calculates the relationship between frame $\{i-1\}$ and $\{i\}$ as follows:

$$T_{i-1,i} = Trans_{z_{i-1}}(d_i) \cdot Rot_{z_{i-1}}(\phi_i) \cdot Trans_{x_i}(a_i) \cdot Rot_{x_i}(\alpha_n) \tag{2.7}$$

Whereas the modified version has a different order of operations:

$$T_{i-1,i}^{mod} = Rot_{x_{i-1}}(\alpha_{i-1}) \cdot Trans_{x_{i-1}}(a_{i-1}) \cdot Trans_{z_i}(d_i) \cdot Rot_{z_i}(\phi_i) \tag{2.8}$$

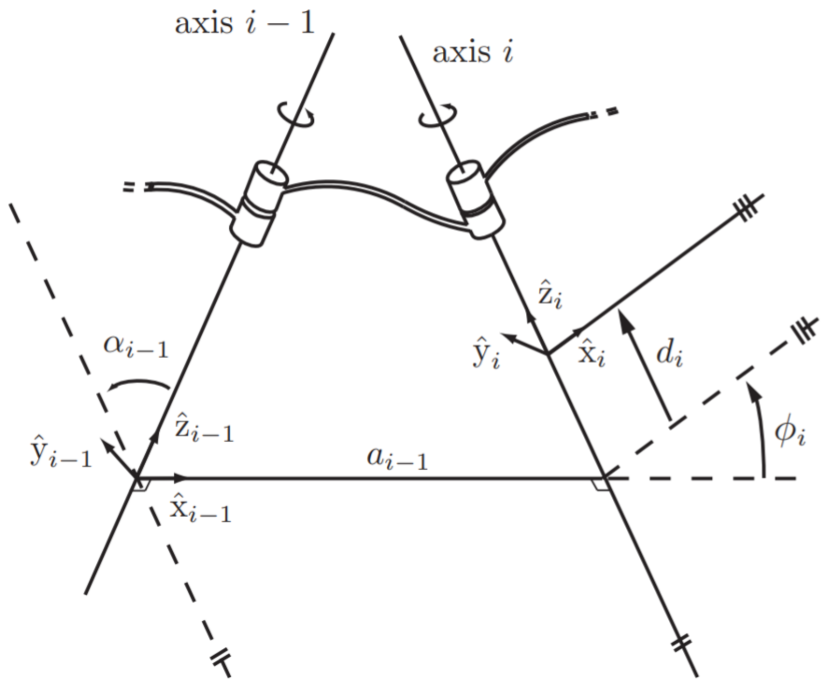


**Figure 2.1:** DH frame placement as presented in Modern Robotics.

It is important to keep in mind that these two transformation matrices, $T_{i-1,i}^{mod}$ and $T_{i-1,i}$, are not equal since frame $\{i-1\}$ and $\{i\}$ are defined differently in the two conventions.

The reason for the difference is easy to understand visually. If one was to move frame $\{i-1\}$ to frame $\{i\}$ in figure 2.2, a translation along the $z_i$ axis is necessary to reach the common normal, then a translation along $x_i$ across the common normal is done to reach joint axis $i+1$. Afterwards a rotation about $x_i$ is done to align the z-axis with the new joint axis.

In figure 2.1 the translation along the common normal can be done immediately, since the frame is already placed on and aligned with it. A translation along the new z-axis is then done to move the frame to the new common normal. In this way, the original method can be thought of as being half a step ahead of the modified method.

In terms of computation complexity and time complexity, there is no difference between the two conventions.

## 2.3 Task 1.3

The space frame PoE method finds the relation between the space frame and the body frame by regarding each joint rotation as a screw motion applied to the end-effector. This method starts at the end-effector and rotates each joint in order from tip to base. To use the PoE method for solving the forward kinematics it is necessary to define a zero-

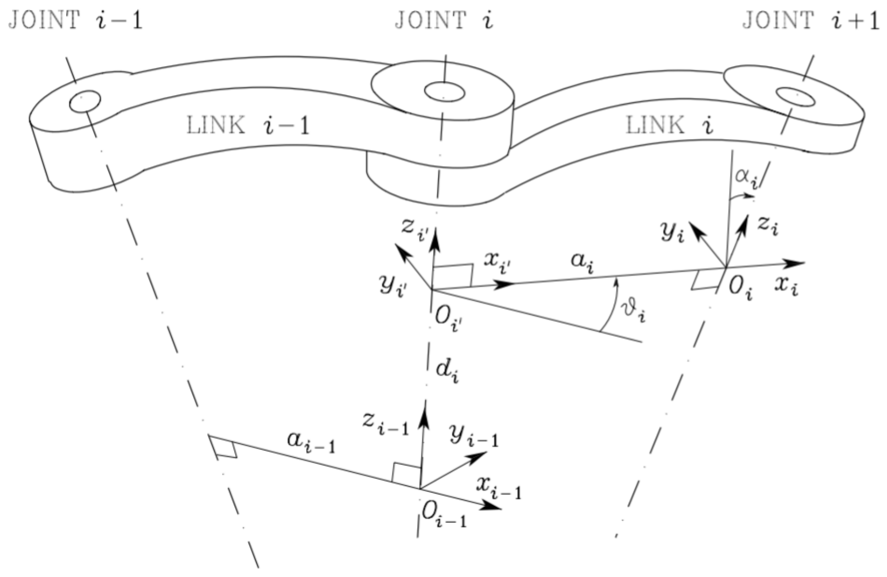


**Figure 2.2:** DH frame placement as presented in Siciliano

configuration body frame, denoted $M$, usually attached to the end-effector. The equation for $n$ joints solved by the PoE method in respect to to base frame, is as follows:

$$T(\theta) = e^{[S_1]\theta_1} \cdots e^{[S_{n-1}]\theta_{n-1}} e^{[S_n]\theta_n} M \tag{2.9}$$

The M gives the position and orientation of the frame located at the end-effector in zero-configuration. Further, $S_n$ represents the screw motion of the respective joints in fixed frame, from zero-position. $\theta_n$ represent the joint angle in joint $n$.

The PoE method has a second formulation, where the screw axis is represented in the end-effector frame, {b}. Using the matrix identity $e^{M^{-1}PM} = M^{-1}e^{P}M$, which can be rewritten to $Me^{M^{-1}PM} = e^{P}M$, and repeatedly applied to equation 2.9.

$$\begin{aligned}
&\mathrm{T}(\theta) = e^{[S_1]\theta_1} \cdots e^{[S_n]\theta_n} M \\
&= e^{[S_1]\theta_1} \cdots Me^{M^{-1}[S_n]M\theta_n} \\
&= e^{[S_1]\theta_1} \cdots Me^{M^{-1}[S_{n-1}]M\theta_{n-1}} e^{M^{-1}[S_n]M\theta_n} \\
&= Me^{M^{-1}[S_1]M\theta_1} \cdot\cdot e^{M^{-1}[S_{n-1}]M\theta_{n-1}} e^{M^{-1}[S_n]M\theta_n} \\
&= Me^{[B_1]\theta_1} \cdots e^{[B_{n-1}]\theta_{n-1}} e^{[B_n]\theta_n}
\end{aligned}$$

where $[B_i]$ is found as $M^{-1}[S_i]M$, i.e. $B_i = [Ad_{M^{-1}}]S_i$. $B_i$ is now the screw axis in reference to the end-effector frame.

## 2.4 Task 1.4

The Denavit-Hartenberg (DH) convention and the Power of Exponential (PoE) convention are both methods for solving forward kinematics of spatial kinematics chains or robot manipulators. The forward kinematics is used to calculate the position and orientation of its end-effector from the joint coordinates.

### Relation between the DH-convention and the PoE method

In the DH-convention, a coordinate frame for each link is positioned according to a set of rules. Then the DH parameters are found, describing the relation between the frames. From these parameters the transformation describing the end link is constructed as described in 2.1.1.

There is a strong relationship between the two methods seen as the PoE formula directly can be derived from the DH-representation. If joint $i$ is a revolute joint, the first three matrices in equation (2.8) are constant. Furthermore, the revolute joint variable is defined as $\theta_i = \phi_i$ and $Rot(\hat{x}, \theta_i)$ rewritten as the matrix exponential

$$Rot(\hat{x}, \theta_i) = e^{[A_i]\theta_i} \tag{2.10}$$

where

$$\boldsymbol{A} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.11}$$

Together with

$$M_i = Rot(\hat{x}, \alpha_{i-1})Trans(\hat{x}, \alpha_{i-1})Trans(\hat{z}, d_i) \tag{2.12}$$

makes it possible for equation 2.8 to be rewritten to

$$T_{i-1,i} = M_i e^{[A_i]\theta_i} \tag{2.13}$$

In the case where joint $i$ is a prismatic joint, $d_i$ is defined as the joint variable and $\phi_i$ is now constant. The order of $Trans(\hat{z}, d_i)$ and $Rot(\hat{z}, \phi_i)$ in equation 2.8 is reversed as the motion becomes the same. This result is the same as in equation 2.13, with $\theta_i = d_i$, $M_i$ given as

$$M_i = Rot(\hat{x}, \alpha_{i-1})Trans(\hat{x}, \alpha_{i-1})Rot(\hat{z}, \vartheta_i) \tag{2.14}$$

and $A$ is now given as

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.15}$$

The forward kinematic for a open chain using DH-parameters can now be written as

$$T_{0,n} = M_1 e^{[A_1]\theta_1} M_2 e^{[A_2]\theta_2} \cdots M_n e^{[A_n]\theta_n} \tag{2.16}$$

By repeatedly applying the identity $Me^P = e^{MPM^{-1}}M$, equation 2.16 can be written as

$$T_{0,n} = e^{[S_1]\theta_1} \cdots e^{[S_{n-1}]\theta_{n-1}} e^{[S_n]\theta_n} M \tag{2.17}$$

where

$$[S_i] = (M_1 \cdots M_i)[A_i](M_1 \cdots M_i)^{-1} \tag{2.18}$$

$$M = M_1 M_2 \cdots M_n \tag{2.19}$$

### 2.4.1 Advantages and disadvantages

There are advantages and disadvantages with both methods. The DH method uses the minimal number of required parameters to map the chains, which are four parameters, to make for a simple representation and compact computation. However, it does this by carefully selecting frames for all links in order to enable subscript cancellations. Because of this the frames can not be chosen arbitrarily. There are multiple methods for choosing the frames, as previously discussed in 2.2, which makes for a none standard method, which again can lead to confusion.

The PoE convention only uses two frames of reference, a base frame and a frame for the end-effector home configuration. The frames can be placed arbitrarily and with no restrictions. There is no need to place reference frames for all joints, which again makes for a simpler construct. Instead the PoE formulation uses the zero-configuration screw axis of each joint, which is easier to find. An additional advantage with the PoE convention over the DH ones is that revolute and prismatic joints are treated the same.

There is also the case that the DH parameters have singularities, meaning they can vary drastically with only small changes to the robots geometry. This is the case when adjacent joint axes are parallel, or close to parallel. In that case the location of the common normal between the two axes is either not uniquely defined, or can change drastically with small variations in the axes' orientation. Unlike DH, PoE does not suffer from this. The joint parameters, i.e. screws, vary smoothly with changes to joint location or orientation.

To summarize, the main disadvantage of the PoE compared to the DH-convention is that PoE uses more parameters. An important advantage of PoE is that the parameters do not suffer from singularities.

# Chapter 3

# Task 2

## 3.1 Task 2.1

The pose of the robot determines the value of the each $\alpha_i$, so different poses result in different DH tables. This can be useful when facing robots with complicated structures, as one can modify the pose to easier identify the parameters. Then, when the table is complete, the angles can be modified so that they once again represent the zero configuration of the robot. This is the case with the Agilus robot, as there is a small shift in height from joint axes 3 and 4. Thus, to simplify the process of obtaining the DH parameters, $\theta_3$ is rotated by 90 degrees, as given by the hint in the assignment. This has the effect of orienting the common normal between joint axes 3 and 4 in the same direction as the previous ones, the $x_s$ direction, eliminating the need to rotate about the z axis before translating along it. To compensate for this modification, the joint angle added is simply subtracted from the joint variable after the DH-table is completed. In more complicated robots, or robots with more offsets, this trick of aligning common normals can significantly simplify the process.

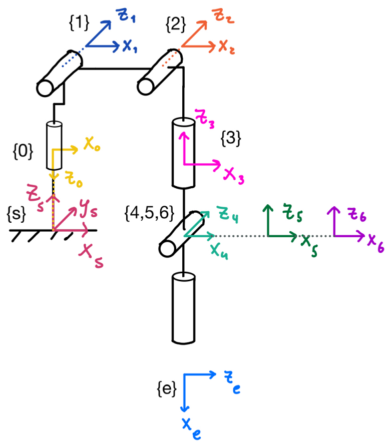


**Figure 3.1:** Simplified model for DH analysis with $\theta_3$ rotated by 90 degrees. The frames are shifted along the doted lines.

To identify the parameters, the procedure from section 2.2 was used:

- To begin with, the space frame {s} is defined as illustrated in figure 3.1.
- The axes of rotation for each joint $i$ is set to be $z_i$, which is defined as in figure 3.3.
- $x$ is drawn up between each $z$. If $z_{i-1}$ is perpendicular to $z_i$, $x_i$ is set to be perpendicular to the plane created by the pair and in this case away from {s}.
- Using the right-hand-rule and the definition in 2.1.1, $\alpha_i$ is obtained by measuring the angle from $z_{i-1}$ to $z_i$.
- As with the previous point, $\vartheta_i$ is obtained by measuring the angle angle from $x_{i-1}$ to $x_i$.
- For joint $n$, $x_n$ is chosen to point in the same direction as $x_s$

Note that the frames {4}, {5} and {6} in figure 3.1 are gathered in joint 5, and that frame {0} is placed with frame {s}. This is allowed for frames with intersecting z axes, in this case a point in the origin of frame {5}. This simplifies both the DH parameter table and the computation of the Jacobian.

| Link | $a_i$ | $\alpha_i$ | $d_i$ | $\vartheta_i$ |
|---|---|---|---|---|
| 1 | 25 | $\frac{\pi}{2}$ | -400 | $\vartheta_1$ |
| 2 | 455 | 0 | 0 | $\vartheta_2$ |
| 3 | 35 | $\frac{\pi}{2}$ | 0 | $\vartheta_3 - \frac{\pi}{2}$ |
| 4 | 0 | $-\frac{\pi}{2}$ | -420 | $\vartheta_4$ |
| 5 | 0 | $\frac{\pi}{2}$ | 0 | $\vartheta_5$ |
| 6 | 0 | 0 | 0 | $\vartheta_6$ |

**Table 3.1:** DH parameter table for Aglius robot when $\theta_3$ is rotated by 90 degrees. All distances are given in [mm] and all angles given in radians

## 3.2 Task 2.2

The procedure to calculate $M$ involves the DH parameters from 3.1. $M$ is a special case of the transformation matrix $\mathbf{T}_e^s$ from equation 2.5, and is calculated using the same procedure as introduced in section 2.1.1. It is important to add an additional rotation to joint 3 of -90 degrees to revert the modifications made under the DH analysis. By using equation 2.6 we get

$$\mathbf{T}_6^0 = \begin{bmatrix} 0 & 0 & -1 & 900 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & -435 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

and for $\mathbf{T}_1^s$ and $\mathbf{T}_e^n$ decide that

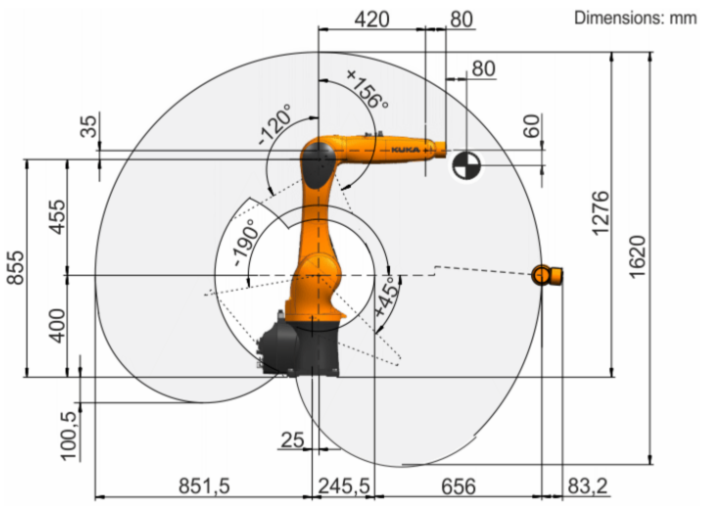


**Figure 3.2:** Dimensions of the KUKA KR6 R900 sixx (Agilus) robot.

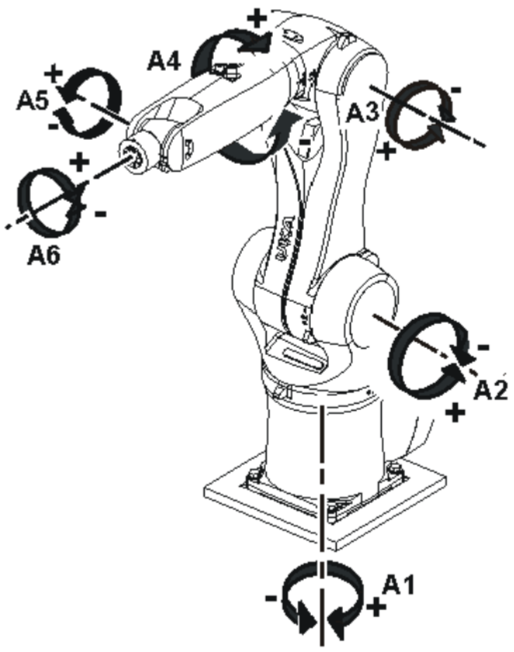


**Figure 3.3:** Direction of robot joint axes for the KUKA KR6 R900 sixx (Agilus) robot.

$$\mathbf{T}_0^s = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

$$\mathbf{T}_e^n = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -80 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

and we get

$$\mathbf{M} = \mathbf{T}_1^s \mathbf{T}_6^1 \mathbf{T}_e^6 = \begin{bmatrix} 1 & 0 & 0 & 980 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 435 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

## 3.3 Task 2.3

The space-frame screw axis $S_i$ was determined by visual inspection.

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 0 & -1 & 0 & -1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -400 & -400 & 0 & -435 & 0 \\ 0 & 0 & 0 & -435 & 0 & -435 \\ 0 & 25 & 480 & 0 & 900 & 0 \end{bmatrix} \tag{3.5}$$

## 3.4 Task 2.4

The body-frame screw axes $B_i$ was calculated using $B_i = [Ad_M^- 1]S_i$.

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & -1 & 0 & -1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 35 & 35 & 0 & 0 & 0 \\ -900 & 0 & 0 & 0 & 0 & 0 \\ 0 & -875 & -420 & 0 & 0 & 0 \end{bmatrix} \tag{3.6}$$

## 3.5 Task 2.5

To visualize the robot a custom class in the The big coordinate system being the s-frame and joint-frames oriented with z-axis along joint screw-axis. Joint 6 is coloured red Axis colours: -Blue -Z axis and joint screw-axis -Red -X axis -Green -Y axis

## 3.6 Task 2.6

To confirm that the forward kinematic solution using the DH-convention and PoE methods is the same, some transformation matrices $\boldsymbol{T_{sb}}$ are calculated using both methods, with the same joint angles, and then compared. To find the $\boldsymbol{T_{sb}^{PoE}}$, the Modern Robotics library is imported, and the function FKinSpace() is used. The function takes in three parameters: S list 3.5, M matrix 3.7 and a list of joint angles.

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 900 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 435 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

The functions used to calculate the $\boldsymbol{T_{sb}^{DH}}$ can be found in task 2 in the code. It takes in the DH-parameters (3.1) found in 3.1 and a list of configuration angles in compliance with the DH method. The same set of angles are used as in the PoE, and the results are compared in the function *PoeDHCompare* using allclose from the Numpy library.

The result from the comparison shows that the $\boldsymbol{T_{sb}}$ matrices match, and therefore gives the same end-effector position and orientation. A sample of three comparisons with different joint angles is shown below.

$$\boldsymbol{\theta_1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.8}$$

$$\boldsymbol{\theta_2} = \begin{bmatrix} 0 & 0 & -\frac{\pi}{2} & \frac{\pi}{2} & 0 & 0 \end{bmatrix} \tag{3.9}$$

$$\boldsymbol{\theta_3} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{3.10}$$

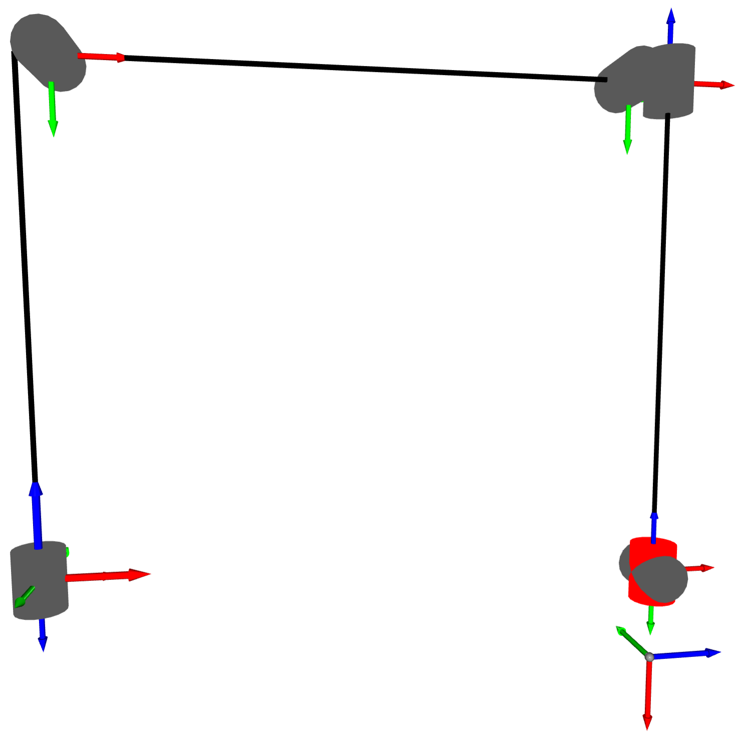


**Figure 3.4:** The DH-frames of the Agilus robot. A cylinder representing a joint $i$ is drawn on frame $i-1$ and aligned with its z-axis. This is done to visualise the placement of the frames relative to the joints as defined in the original DH convention. Note that the placement of the cylinders will not exactly correspond to the actual robots joints, but functionally they give the same behavior. All three wrist frames({3},{4},{5}) are co-located in the wrist(shown in red). The end-effector frame is translated $-80mm$ along axis $z_5$ to get its correct placement.

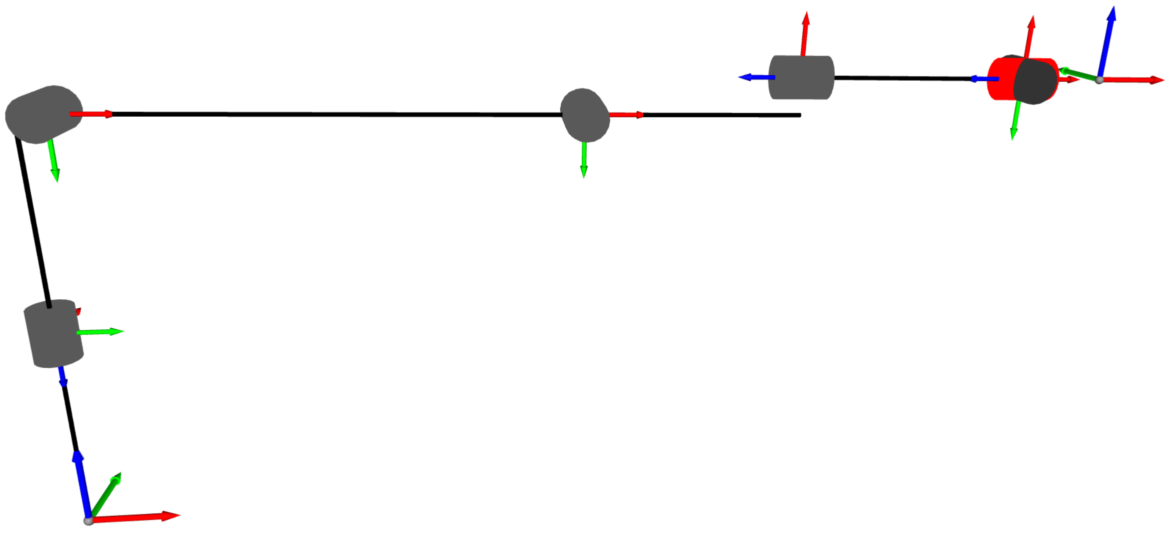
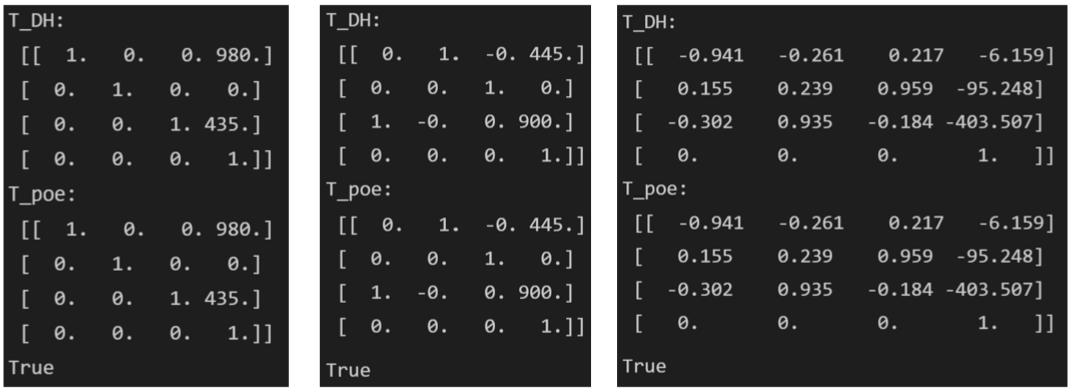


**Figure 3.5:** Visualization of Kuka robot in zero-configuration. Showing Me, Si and Bi: Me is the chains last frame (without grey joint), screw axes as the joint frames' z-axis (both Bi and Si). {4} is drawn outside of the wrist to show functionality better. {b} is located inside the wrist(red/grey) and oriented as {e}



```
T_DH:
 [[  1.   0.   0. 980.]
 [  0.   1.   0.   0.]
 [  0.   0.   1. 435.]
 [  0.   0.   0.   1.]]
T_poe:
 [[  1.   0.   0. 980.]
 [  0.   1.   0.   0.]
 [  0.   0.   1. 435.]
 [  0.   0.   0.   1.]]
True
```

```
T_DH:
 [[  0.   1.  -0. 445.]
 [  0.   0.   1.   0.]
 [  1.  -0.   0. 900.]
 [  0.   0.   0.   1.]]
T_poe:
 [[  0.   1.  -0. 445.]
 [  0.   0.   1.   0.]
 [  1.  -0.   0. 900.]
 [  0.   0.   0.   1.]]
True
```

```
T_DH:
 [[  -0.941   -0.261    0.217   -6.159]
 [   0.155    0.239    0.959  -95.248]
 [  -0.302    0.935   -0.184 -403.507]
 [   0.       0.       0.       1.   ]]
T_poe:
 [[  -0.941   -0.261    0.217   -6.159]
 [   0.155    0.239    0.959  -95.248]
 [  -0.302    0.935   -0.184 -403.507]
 [   0.       0.       0.       1.   ]]
True
```

**Figure 3.6:** Result when comparing $\boldsymbol{T}$ matrices, using the configuration described in (3.8), (3.9) and (3.10) respectively

# Chapter 4

# Task 3

## 4.1 Task 3.1

It is possible to find the inverse kinematics of an open-chain manipulator using both a numerical- and analytic method. By solving the inverse kinematics, the parameters needed to move the end-effector to the desired location is found. Meaning the inverse kinematic solution finds $\boldsymbol{\theta}$ such that $\boldsymbol{T}(\boldsymbol{\theta}) = \boldsymbol{X}$, given $\boldsymbol{X} \in SE(3)$,

It is more complex to solve the inverse kinematic problems than the forward kinematic solution. This is because when calculating the forward kinematics, the end-effector pose is explicitly given by the joint variables. This is not the case the other way around. In fact, there are often multiple or even infinite solutions. In the case where a solution does exist, it is necessary to consider the physical constraints of the robot. The end-effector orientation and position must be within the open-chains workspace.

The analytic method uses the function *atan2(x,y)* and the law of cosines to derive a relationship between the angles and the end-effector position. Since the last three joint axes of the open-chain intersect at a common point, a decoupling of the problem is possible. The problem is dissected into an inverse position and inverse orientation problem. This is explained more thoroughly in section 4.2

The numerical method used to solve the inverse kinematics is a iterative numerical algorithm called the Newton-Raphson method. We want to find $\boldsymbol{\theta}$ such that $\boldsymbol{x_d} - f(\boldsymbol{\theta}) = \boldsymbol{0}$. Where $\boldsymbol{x_d}$ describes the desired end effector configuration, and $f(\boldsymbol{\theta})$ is the forward kinematics of the robot with a given set of angles. This is done by these two steps:

1. Given $\boldsymbol{x_d} \in \boldsymbol{R^6}$ describing a desired end-effector pose, and an initial guess $\boldsymbol{\theta}^0 \in \boldsymbol{R^n}$, set $i = 0$

2. Set $\boldsymbol{e} = \boldsymbol{x_d} - f(\boldsymbol{\theta}^i)$. While $||\ \boldsymbol{e}\ || > \epsilon$ for some acceptable error $\epsilon$: Set $\theta^{i+1} = \theta^i + J^\dagger(\theta^i)\boldsymbol{e}$. Where $J^\dagger(\theta^i)$ is the pseudo-inverse of the Jacobian matrix.

Then increment $i$

There are advantages and disadvantages with both methods. The numerical method is great for implementation in computers and provides great accuracy. It can also be used for redundant open-chain manipulators. In this case, the method will give us the solution which requires the least change in theta. One of the main advantages with Newton-Raphson is that unlike the analytic method, in the case where no solution exists, we can find the closest solution possible. However, the numerical method requires an initial guess to compute, which must be close enough to the solution for the algorithm to converge.

The analytic method gives all the inverse kinematic solutions and will determine whether no solution exist. The equations needed in this method can be cumbersome and tedious to derive, but when this is done the solutions are fast to compute. Unlike the numerical method, the analytic one does not require an initial guess or solution parameters. This method can only be used with nonredundant robots, and the equations must be derived independently for robots with different kinematic structures.

## 4.2 Task 3.2

Directly finding the analytical solution for a general 6R robot is difficult since there are many variables to consider. However, in the case of most commercially used 6R robots, like the KUKA KR6 R900 sixx (Agilus) considered in this project, the joints are oriented in such a way that decoupling of the problem is possible. A simplified drawing of the robot is shown in figure 4.1. Since the last three joints have intersecting joint axes, a point $P$ at the intersection can be found. This point is located inside the fifth joint and has the important properties of being entirely given by the desired end-effector configuration, and only being affected by the three first joint angles. Since $P$ has these properties, the first three joints must be responsible for reaching that point, giving a much easier sub-problem. The last three joint angles are sufficient to get any orientation on the end-effector, making the decoupling possible. They are defined by the remaining rotation required to get the desired orientation. These angles are found by using Euler-angles.

In this section the space frame and the desired end-effector frame will be denoted as s and d respectively. In figure 4.1 the space frame and the zero-configuration end-effector frame are shown. The notation used will be in accordance with Modern Robotics, where $T_{xy}$ describes a frame $y$ based on the frame $x$.

First the point $P$ must be described in the space frame, denoted $P_s$. The point $P_s$ is found by describing $P$ in the desired end-effector frame, then changing its base frame using the following formula:

$$P_s = T_{sd}^{-1} P_d \tag{4.1}$$

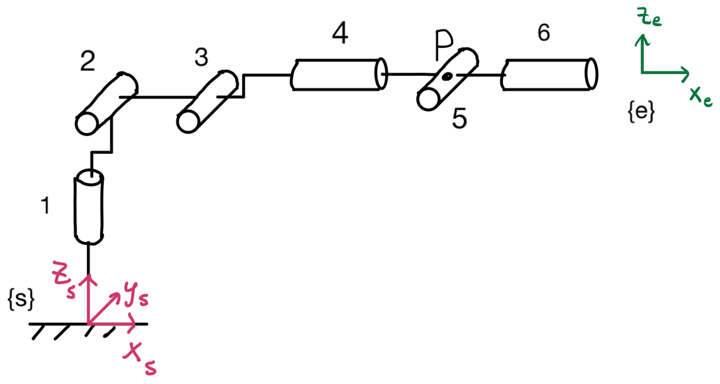


**Figure 4.1:** A simplified figure of the robot considered in this project, illustrating the position of P. The space frame and end-effector frame are drawn.

Finding a solution for the first joint variable, $\theta_1$, is simple. Only the first of the non-wrist joints is able to give any rotation about the z-axis. Therefore it must be responsible for pointing the robot in the direction of $P_s$ projected onto the xy-plane, giving

$$\theta_1 = -\arctan\frac{P_{s,y}}{P_{s,x}} \tag{4.2}$$

or, as is more useful for our application:

$$\theta_1 = -atan2(P_{s,x}, P_{s,y}) \tag{4.3}$$

Where $atan2$ is a function defined in the Python package Sympy. This function is the same as a normal arctan, but gives the angle in the correct quadrant, which is not generally the case for arctan. The minus sign is added to compensate for the fact that joint axis 1 is defined in the negative z direction.

The joint angle $\theta_1$ could also point in the opposite direction, requiring the robot to "bend over backwards", giving a different set of joint values. However, these solutions are not considered in this analytical solution.

Finding the second and third joint angles requires use of the law of cosines. A triangle based in joint 2, 3 and $P$ can be constructed, as shown in figure 4.2. The $35mm$ offset in joint 3 complicates this process slightly, since the links do not by themselves form a triangle. Therefore an imaginary line must be drawn from link 3 to $P$ to complete the triangle, shown as edge $a$ in figure 4.2. Because of this, the angle $\beta$ deviates from joint 3's angle by a constant $\psi$. This needs to be taken into consideration when calculating the joint angles.

First the lengths $a$, $b$ and $c$ must be calculated. They are given as:

$$a = \sqrt{420^2 + 35^2} \tag{4.4}$$

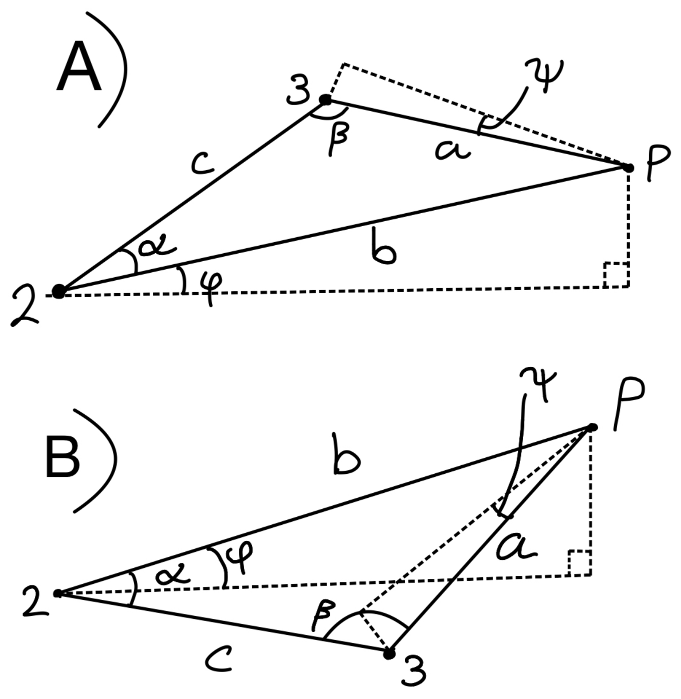


**Figure 4.2:** An elbow up(A) and elbow down(B) solution for a general point P, viewed from the $-y$ direction. The dotted lines forming a right angle from point 2 to P show the projection of P onto the xy-plane. The dotted line from point 3 to P shows the physical link of the robot.

$$c = 455 \tag{4.5}$$

$$b = \sqrt{(\sqrt{P_{s,x}^2 + P_{s,y}^2} - 25)^2 + (P_{s,z} - 400)^2} \tag{4.6}$$

Lengths $a$ and $c$ are constant and given by the robot links, but $b$ will change based on $P$. Length $b$ is calculated using the Pythagoras theorem on the right triangle constructed by the projection of $P$. The complicated expression for $b$ is caused by the offset of $25mm$ from $\{s\}$ to link 1. The $-400$ term must be added since $P_s$ is based in the space frame located $400mm$ beneath joint 2, the origin of our triangle.

With these lengths, all four angles shown in figure 4.2 can be calculated:

$$\psi = \arccos \frac{420}{a} \tag{4.7}$$

$$\phi = atan2(P_{s,z}, \sqrt{P_{s,x}^2 + P_{s,y}^2} - 25) \tag{4.8}$$

$$\alpha = \arccos \frac{b^2 + c^2 - a^2}{2bc} \tag{4.9}$$

$$\beta = \arccos \frac{a^2 + c^2 - b^2}{2ac} \tag{4.10}$$

The term 420 in equation 4.7 is the length of the adjacent side to $\psi$.

With these angles, joint angles $\theta_2$ and $\theta_3$ can be calculated for the elbow up case:

$$\theta_2 = -(\alpha + \phi) \tag{4.11}$$

$$\theta_3 = \pi - (\beta - \psi) \tag{4.12}$$

And for the elbow down case:

$$\theta_2 = -(\phi - \alpha) \tag{4.13}$$

$$\theta_3 = -\pi + \beta + \psi \tag{4.14}$$

Note that the joint axes of joints 2 and 3 are in the y direction, meaning the theta values are defined as positive in the clockwise direction in figure 4.2. The $\pi$ in equation 4.12 and 4.14 are added to compensate for the defined zero configuration of the joint. The angle $\psi$ is added to compensate for the $35mm$ offset slightly changing the angle needed to reach the point P.

In order to find the remaining angles, Euler angles are used. The remaining rotation required can be described using the power of exponentials formulation:

$$e^{[S_1]\theta_1}e^{[S_2]\theta_2}e^{[S_3]\theta_3}e^{[S_4]\theta_4}e^{[S_5]\theta_5}e^{[S_6]\theta_6}M = T_{sd} \tag{4.15}$$

By isolating the three unknown angles on the left side, an expression for the remaining rotation based on the space frame is found:

$$e^{[S_4]\theta_4}e^{[S_5]\theta_5}e^{[S_6]\theta_6} = e^{-[S_3]\theta_3}e^{-[S_2]\theta_2}e^{-[S_1]\theta_1}T_{sd}M^{-1} = R \tag{4.16}$$

In the case of the robot considered in this project, and with the space frame as defined in figure 4.1, the last three joints correspond to XYX Euler angles.

The general matrix obtained by rotating about the $x$ axis, then about the $y$ axis and finally about the $x$ axis is given as follows:

$$R_e = \begin{bmatrix} c_2 & s_2s_3 & c_3s_2 \\ s_1s_2 & c_1c_2 - c_2s_1s_3 & -c_1s_3 - c_2c_3s_1 \\ -c_1s_2 & c_3s_1 + c_1c_2s_3 & c_1c_2c_3 - s_1s_3 \end{bmatrix} \tag{4.17}$$

Where $c$ and $s$ represent the cosine and sine functions, and their subscript refer to the which rotation angle they take as input.

These two rotation matrices are set equal to each other:

$$R_e = R \tag{4.18}$$

By exploiting the indices of $R_e$, an expression for each of the rotations can be found. The three angles can be obtained as follows:

$$\theta_4 = -atan2(R_{21}, -R_{31}) \tag{4.19}$$

$$\theta_5 = atan2(\sqrt{1 - R_{11}}, R_{11}) \tag{4.20}$$

$$\theta_6 = -atan2(R_{12}, R_{13}) \tag{4.21}$$

The first minus sign in equation 4.19 and 4.21 is added since the corresponding joint axes point in the $-x$ direction.

There is another set of solutions possible since the arctan function always has two solutions on opposite sides of the unit circle. The second solution can be found by setting $\theta_5$ negative and subtracting $\pi$ from $\theta_4$ and $\theta_6$. Only the solution listed in the above equations will be used in this analytical solution.

With these steps two analytical solutions are found, differing in joint angles 2 and 3, giving an elbow up and elbow down solution. There are, however, in total eight possible solutions to the general problem, all of which can be found using this method. There is an elbow up and elbow down solution for each of the two choices of $\theta_1$. For each of those four choices, there are two solutions to the remaining three joint angles, giving in total eight possible solutions. Some of these solutions may coincide depending on the desired configuration, and some may only be theoretically possible, depending on the joint limits of the physical robot.

## 4.3 Task 3.3

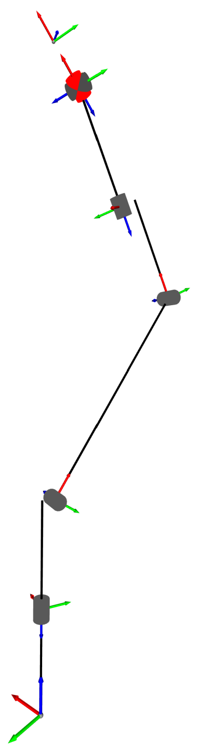
The function IKinSpace is used to calculate a numerical solution. This is a function from the Python library provided by Modern Robotics. This function uses the iterative Newton-Rhapson root-finding method to solve an inverse kinematics problem within a given margin of error. The function takes as input the desired body frame $T_{sd}$, a list of spacial screws, the M matrix, and an initial guess $\theta_0$.

The analytical elbow down, analytical elbow up and the numerical solution always give the correct desired body frame, meaning they are all equally correct solutions.

| Cases | $\theta_1$ | $\theta_2$ | $\theta_3$ | $\theta_4$ | $\theta_5$ | $\theta_6$ |
|---|---|---|---|---|---|---|
| Elbow up | 2.000 | -2.037 | 1.166 | 2.796 | 1.627 | -0.573 |
| Elbow down | 2.000 | -1.000 | -1.000 | 2.500 | 0.600 | 0.000 |
| Numerical A | 2.000 | -1.000 | -1.000 | -0.642 | -0.600 | 3.142 |
| Numerical B | -1.142 | -2.199 | 1.147 | -0.732 | 0.530 | 0.107 |

**Table 4.1:** An example of four valid solutions to a randomly chosen $T_{sd}$. The elbow up and elbow down solutions are analytical. The two numerical solutions show the result of two different choices for $\theta_1$.

As discussed in section 4.2, there are generally eight possible solutions to a given $T_{sd}$. Which one of these is chosen by the numerical method is entirely dependent on the initial guess $\theta_0$. The numerical algorithm generally chooses the closest solution to the initial guess. By "*rigging*" the numerical solver by guessing very close to one of the analytical


[Numerical solution A]

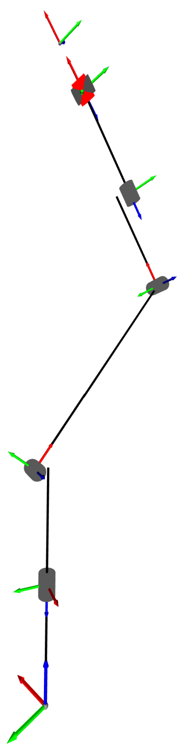

[Numerical solution B]

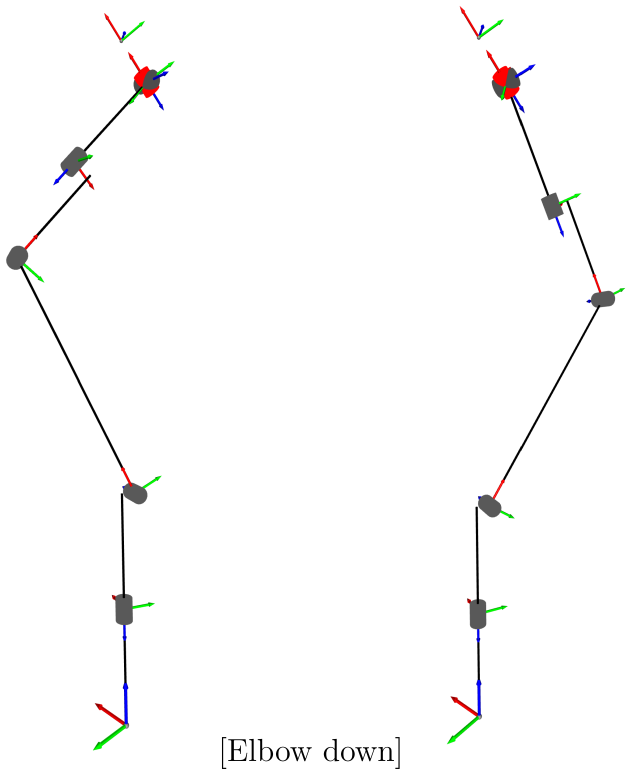
**Figure 4.3:** The two numerical solutions listed in table 4.1. By looking at the offset of link 1, it is clear that the Numerical B configuration "bends over backwards".

solutions, the joint values will be identical.

When the numerical solution chooses the first joint angle as 180° opposite to the analytical solutions, the solutions deviate a lot. This is exemplified by the Numerical B solution in table 4.1. If, however, the first joint angle is the same, the joint angles in joint 2 and 3 will be identical to either the elbow up or elbow down analytical solutions. In the case of the Numerical A solution shown in table 4.1, even though the first three angles are identical to the analytical elbow down solution, the last three angles are different. This is due to the two possible solutions to equation 4.17 discussed in section 4.2, meaning the numerical solution is not always identical to one of the analytical solutions even if $\theta_1$ is the same.

It can be concluded that the two analytical solutions found are also possible numerical solutions, but are not always chosen by the algorithm since there are seven other possible solutions.

## 4.4 Task 3.4



[Elbow up] [Elbow down]

**Figure 4.4:** The two analytical solutions listed in table 4.1.

# Chapter 5

# Task 4

## 5.1 Singularities

A singularity is a configuration of an open chain in which the end-effector loses the ability to instantaneously move in one or more directions. There are two kinds of singularities; boundary and internal. The boundary singularities represent the outer limits of the robots workspace, which can be seen in 3.2, as the robot is prohibited from reaching any further by the physical system. Unlike the internal singularities they cannot be avoided and are therefore of little interest. The internal singularities occur within the workspace, which can disrupt operations and it is therefore necessary to identify the configurations where these occur. When moving over these singularities, small changes in the cartesian space induce infinite velocities in the joint space.

The relation between the end-effectors velocity and the joint velocities is given by the Jacobian matrix, as shown in equation 5.1. Any achievable motion is therefore a linear combination of the columns of the Jacobian. The general expressions for these columns are shown in equation 5.2

$$\boldsymbol{v} = \boldsymbol{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{5.1}$$

$$\boldsymbol{J} = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 & \omega_5 & \omega_6 \\ -\omega_1 \times q_1 & -\omega_2 \times q_2 & -\omega_3 \times q_3 & -\omega_4 \times q_4 & -\omega_5 \times q_5 & -\omega_6 \times q_6 \end{bmatrix} \tag{5.2}$$

Where $\omega_i$ is the direction of joint axis i, and $q_i$ is an arbitrary point on that axis.

When six of the columns are linearly independent, any end-effector velocity vector can be achieved. However, if the matrix has a lower rank than 6, there exists vectors that are outside the Jacobians span. Since the Jacobian in this project is a 6x6 matrix, this is equivalent with the Jacobian being singular. A singularity can therefore be identified as the joint values giving a singular Jacobian. That is, the thetas solving equation 5.3.

$$det(\boldsymbol{J}(\boldsymbol{\theta})) = 0 \tag{5.3}$$

This equation is very cumbersome to compute for a 6x6 matrix, but can be simplified significantly by decoupling.

## 5.2 Singularity Decoupling

To find the singularities of the Agilus robot, one can decouple the body Jacobian into wrist and arm singularities. Wrist singularities are the result of motion in the *last* three joints, i.e. the wrist. Arm singularities are the result of movement in the *first* three joints. As there are six columns in the jacobian, it can be divided in two; the arm and the wrist section. These again can be divided in two; the top half of the jacobian represent the angular velocity of the joints and the bottom half represent the linear velocity of the end-effector induced by rotation in the respective joint. This results in a jacobian of the form illustrated in 5.4.

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_{11} & \boldsymbol{J}_{12} \\ \boldsymbol{J}_{21} & \boldsymbol{J}_{22} \end{bmatrix} \tag{5.4}$$

Note that the Jacobian deviates from the notation in *Siciliano*, where the linear velocities are placed in the upper half and the screw axes are in the lower half.

One can use either the {s} of {b} frame to make use of decoupling as long as the corresponding Jacobian is used. In the following sections, the {b} frame and the body Jacobian will be used to identify the singularities.

By placing the end-effector frame on the axis intersecting with the wrist axes, the following is obtained 5.5.

$$\boldsymbol{J}_{22} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.5}$$

Now, the determinant of 5.4 is reduced to the product of the remaining diagonal, as shown in 5.6

$$det(\boldsymbol{J}) = det(\boldsymbol{J}_{12})det(\boldsymbol{J}_{21}) \tag{5.6}$$

This simplifies the expression to 5.7.

$$\mathbf{J}_{12} = 0 \vee \mathbf{J}_{21} = 0 \tag{5.7}$$

The first condition of 5.7 represent the arm singularities, and the second the wrist singularities.

Even with these simplifications, the calculations are cumbersome to calculate even for a computer. The singularities are therefore found through an analytical dissection of the problem. Since singularities are independent of what reference frame is used, the space frame can be placed wherever it makes the calculations easiest. By strategic placing of the space frame to simplify the Jacobian, *Modern Robotics* identifies six common cases of singularities, listed below. By utilizing these, and by comparing them to the geometry of the Agilus, five singularities can be found.

- **Case 1**: Two collinear revolute joints axes.
- **Case 2**: Three Coplanar and paralell revolute joint axes.
- **Case 3**: Four revolute axes intersecting at a common point.
- **Case 4**: Four coplanar revolute joints.
- **Case 5**: Six revolute joints intersecting a common line.

| Cases | $\theta_1$ | $\theta_2$ | $\theta_3$ | $\theta_4$ | $\theta_5$ | $\theta_6$ |
|---|---|---|---|---|---|---|
| Case 1 | $\theta_1$ | $\theta_2$ | $\theta_3$ | $\theta_4$ | 0 | $\theta_6$ |
| Case 2, 5 | $\theta_1$ | $\theta_2$ | 0.083 | $\theta_4$ | $\theta_5$ | $\theta_6$ |
| Case 3 | $\theta_1$ | -2.701 | 2.379 | $\theta_4$ | 1.892 | $\theta_6$ |
| Case 4 | $\theta_1$ | $\theta_2$ | 0.083 | $\theta_4$ | -0.083 | $\theta_6$ |

**Table 5.1:** Angles representing singularities. Angles that are arbitrary for the given case are not assigned any value.

**Case 1** is almost exclusively found in the wrist. When $\theta_5 = 0$, the axes of joints 4 and 6 align, meaning $\omega_4 = \pm\omega_6$. Therefore $\mathbf{J}_{12}$ is obviously singular, resulting in the whole Jacobian being singular. The configuration is shown in figure 5.1

Case 1 can also be achieved during a shoulder singularity as shown in figure 5.2, where axes 6 and 1 are co-linear. This is a very specific and rare configuration and is avoided automatically by avoiding the shoulder singularity. It is therefore of little interest.

**Case 2** is the elbow singularity and is reachable by the Agilus robot. It is a boundary singularity, meaning the robots end-effector is at the boundary of its workspace. When joint axes 2, 3 and 5 are parallel and coplanar, the robot arm is completely stretched out, disallowing for any further movement outward. This singularity is only dependent on joint axis 3, the elbow, and the corresponding joint value is presented in table 5.1. The robots configuration in this singularity is shown in figure 5.1. Boundary singularities such as this one are not as interesting as internal singularities since the robot can never cross this singularity, meaning high joint velocities are never induced. They are also impossible to avoid, unless the robot is disallowed from stretching out completely by joint limits.

**Case 3** is also a possible singularity for the Agilus. This is an internal shoulder singularity. Because of the robots geometry, joint axes 2 and 3 can never intersect with each other, and can only ever intersect with two of the other joint axes. Therefore the only possibility is joints 1, 4, 5 and 6 intersecting. This is achievable when the wrist is positioned directly above the shoulder joint. By positioning the space frame on the intersection between the four axes, all $v$ terms are set to zero. This results in four columns where the bottom three indices are zero, meaning they can not be linearly independent. This is shown in equation 5.8. In order to continue a linear movement over this singularity, the robots shoulder has to rotate $180^{\text{deg}}$ infinitely fast. An example of joint values for this singularity is shown in table 3.1.

The offsets in link 1 and 3 ensure that this singularity does not occur when the robot is in its home configuration.

$$J = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 & \dots \\ 0 & 0 & 0 & 0 & \dots \end{bmatrix} \tag{5.8}$$

**Case 4** is possible by simply rotating joint 5 slightly in the configuration shown in fig. (5.1), such that joint axis 6 is coincident with the orange line. This is only possible in the same pose as in **case 2**.

**Case 5** is also only possible in the same pose as in **case 2**. Since joint axes 2, 3 and 5 are parallel, they cannot intersect the same line without being coplanar. As long as joint axes 2, 3 and 5 are coplanar, because of the robots geometries, all joint axes will intersect a common line thus satisfying the condition.

In conclusion, there are only three unique singularities that can happen independent of each other, one of which is a redundant boundary singularity. The two internal singularities, shoulder and wrist, need to be considered when using the robot.

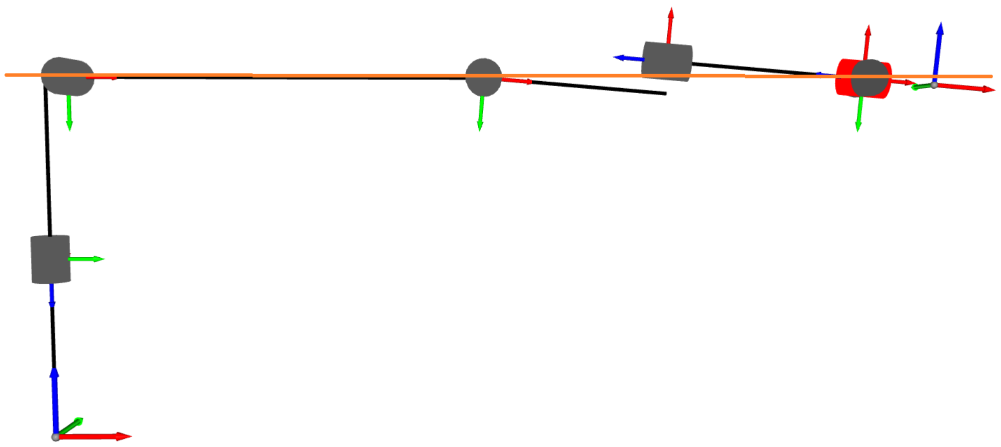


**Figure 5.1:** Singularities for case 1, 2 and 5. The orange line represent a section of the plane spanned by joint 2, 3 and 5. This is also the line which all joint axes intersect, satisfying the conditions for case 5.

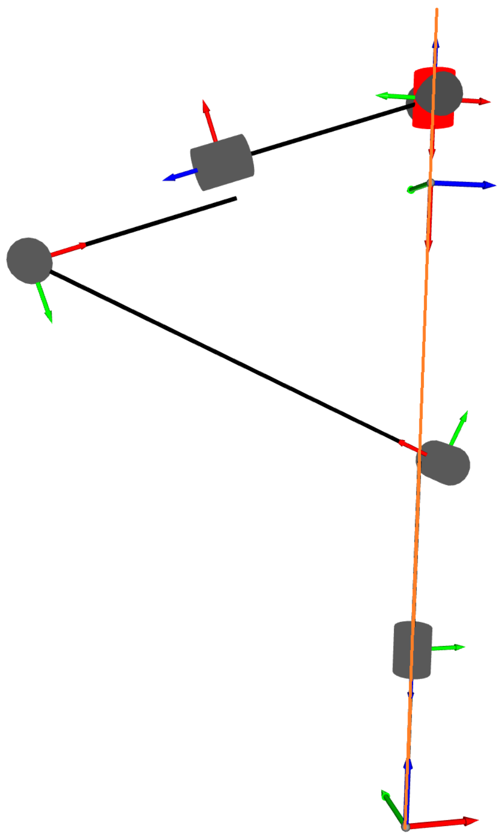


**Figure 5.2:** Singularity for case 3. The orange line represent the axis of joint 1 intersecting the wrist axes.

# References

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. SpringerVerlag, 2010, ISBN: 978-1-84628-641-4.

[2] K. M. Lynch and F. C. Park, *Modern Robotics*. Cambridge, 2021, ISBN: 978-1-107-15630-2.

# Chapter 6

# Code