UNIVERSITÀ DEGLI STUDI DI GENOVA

DIBRIS

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY,
BIOENGINEERING, ROBOTICS AND SYSTEM ENGINEERING

MODELLING AND CONTROL OF MANIPULATORS

First Assignment

Equivalent representations of orientation matrices

Author:

Surname Name
Condorelli Valentina
Meschini Marco
Piacenza Enrico

Student ID:

s4945679
s6273938
s4878469

Professors:

Enrico Simetti
Giorgio Cannata

Tutors:

Andrea Tiranti
Francesco Giovinazzo
George Kurshakov

October 30, 2023

Contents

Mathematical expression	Definition	MATLAB expression
$< w >$	World Coordinate Frame	w
${}^{a}_{b}R$	Rotation matrix of frame $< b >$ with respect to frame $< a >$	aRb
${}^{a}_{b}T$	Transformation matrix of frame $< b >$ with respect to frame $< a >$	aTb

Table 1: Nomenclature Table

1 Assignment description

The first assignment of Modelling and Control of Manipulators focuses on the geometric fundamentals and algorithmic tools underlying any robotics application. The concepts of transformation matrix, orientation matrix and the equivalent representations of orientation matrices (Equivalent angle-axis representation, Euler Angles and Quaternions) will be reviewed.

The first assignment is **mandatory** and consists of 4 different exercises. You are asked to:

- Download the .zip file called MOCOM-LAB1 from the Aulaweb page of this course.
- Implement the code to solve the exercises on MATLAB by filling the predefined files called "*main.m*", "*ComputeAngleAxis.m*", "*ComputeInverseAngleAxis.m*", and "*QuatToRot.m*".
- Write a report motivating the answers for each exercise, following the predefind format on this document.

1.1 Exercise 1 - Equivalent Angle-Axis Representation (Exponential representation)

A particularly interesting minimal representation of 3D rotation matrices is the so-called "*angle-axis representation*" or "*exponential representation*". Given two frames $< a >$ and $< b >$, initially coinciding, let's consider an applied geometric unit vector $(\mathbf{v}, O_a) = (\mathbf{v}, O_b)$, passing through the common origin of the two frames, whose initial projection on $< a >$ is the same of that on $< b >$. Then let's consider that frame $< b >$ is purely rotated around **v** of an angle θ, even negative, accordingly with the right-hand rule. We note that the axis-line defined by $(\mathbf{v}, O_a) = (\mathbf{v}, O_b)$ remains common to both the reference systems of the two frames $< a >$ and $< b >$ and we obtain that the orientation matrix constructed in the above way is said to be represented by its equivalent angle-axis representation that admits the following equivalent analytical expression, also known as Rodrigues Formula:

$$\mathbf{R}(^*\mathbf{v}, \theta) = e^{[^*\mathbf{v}\times]\theta} = e^{[\rho\times]} = \mathbf{I}_{3x3} + [^*\mathbf{v}\times]\sin(\theta) + [^*\mathbf{v}\times]^2(1 - \cos(\theta))$$

Q1.1 Given two generic frames $< a >$ and $< b >$, given the geometric unit vector $(\mathbf{v}, O_a) = (\mathbf{v}, O_b)$ and the angle θ, implement on MATLAB the Rodrigues formula, computing the rotation matrix ${}^a_bR$ of frame $< b >$ with respect to $< a >$.

Then test it for the following cases and comment the results obtained, including some sketches of the frames configurations:

- **Q1.2** $\mathbf{v} = [1, 0, 0]$ and $\theta = 45°$
- **Q1.3** $\mathbf{v} = [0, 1, 0]$ and $\theta = \pi/6$
- **Q1.4** $\mathbf{v} = [0, 0, 1]$ and $\theta = 3\pi/4$
- **Q1.5** $\mathbf{v} = [0.3202, 0.5337, 0.7827]$ and $\theta = 2.8$
- **Q1.6** $\boldsymbol{\rho} = [0, 2\pi/3, 0]$;
- **Q1.7** $\boldsymbol{\rho} = [0.25, -1.3, 0.15]$;
- **Q1.8** $\boldsymbol{\rho} = [-\pi/4, -\pi/3, \pi/6]$;

Note that ρ is $\boldsymbol{\rho} = \mathbf{v} * \theta$

1.2 Exercise 2 - Inverse Equivalent Angle-Axis Problem

Given two reference frames $< a >$ and $< b >$, referred to a common world coordinate system $< w >$, their orientation with respect to the world frame $< w >$ is expressed in Figure 1.

Q2.1 Compute the orientation matrix ${}^a_bR$, by inspection of Figure 1, without using the Rodriguez formula.
Q2.2 Solve the Inverse Equivalent Angle-Axis Problem for the orientation matrix ${}^a_bR$.
Q2.3 Given the following Transformation matrix:

$${}^w_cT = \begin{bmatrix} 0.835959 & -0.283542 & -0.46986 & 0 \\ 0.271321 & 0.957764 & -0.0952472 & -1.23 \\ 0.47703 & -0.0478627 & 0.877583 & 14 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Solve the Inverse Equivalent Angle-Axis Problem for the orientation matrix ${}^c_bR$.

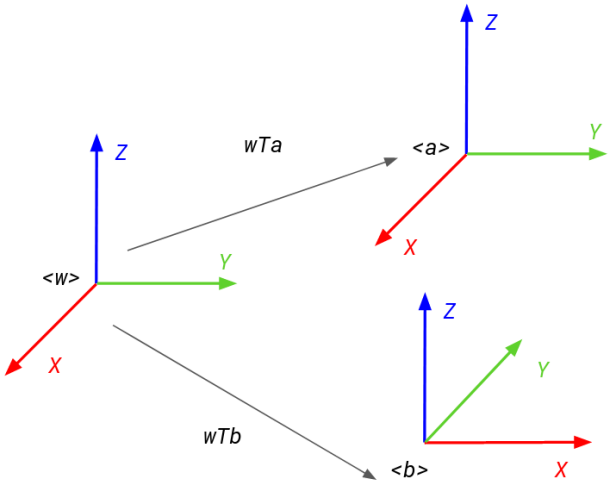


Figure 1: exercise 2 frames

1.3 Exercise 3 - Euler angles (Z-X-Z) vs Tait-Bryan angles (Yaw-Pitch-Roll)

Any orientation matrix can be expressed in terms of three elementary rotations in sequence. These can occur either about the axes of a fixed coordinate system (extrinsic rotations), or about the axes of a rotating coordinate system (intrinsic rotations) initially aligned with the fixed one. Then we can distinguish:

- Proper Euler angles: X-Z-X, Y-Z-Y, ...
- Tait-Bryan angles: Z-Y-X, X-Y-Z, ...

Q3.1 Given two generic frames $< w >$ and $< b >$, define the elementary orientation matrices for frame $< b >$ with respect to frame $< w >$, knowing that:

- $< b >$ is rotated of $45°$ around the z-axis of $< w >$
- $< b >$ is rotated of $60°$ around the y-axis of $< w >$
- $< b >$ is rotated of $-30°$ around the x-axis of $< w >$

Q3.2 Compute the equivalent angle-axis representation for each elementary rotation
Q3.3 Compute the z-y-x (yaw,pitch,roll) representation and solve the Inverse Equivalent Angle-Axis Problem for the obtained orientation matrix
Q3.4 Compute the z-x-z representation and solve the Inverse Equivalent Angle-Axis Problem for the obtained orientation matrix

1.4 Exercise 4 - Quaternions

Given the following quaternion: $q = 0.1647 + 0.31583i + 0.52639j + 0.77204k$ expressing how a reference frame $< b >$ is rotated with respect to $< a >$:
Q4.1 Compute the equivalent rotation matrix, **WITHOUT** using built-in matlab functions.
Q4.2 Solve the Inverse Equivalent Angle-Axis Problem for the obtained orientation matrix

2 Exercise 1

In the first exercise, we had to compute the orientation matrix corresponding to a set of given angle-axis representations. In order to do that, we implemented a function, called *ComputeAngleAxis()*, based on the Rodrigues formula. The function took as input two parameters:

- $\underline{v}$: the vector around which the rotation is performed;
- θ: the corresponding rotation angle (in radians).

The output of the function was the orientation matrix ${}^a_bR$ of frame $< b >$ w.r.t. frame $< a >$.

While in the first sets directly θ and $\underline{v}$ were given, in questions 1.6, 1.7 and 1.8 the vector $\underline{\rho} = \underline{v} \cdot \theta$ was given instead. Therefore, from $\underline{\rho}$, the parameter θ was first obtained as:

$$\theta = \text{norm}(\underline{\rho}) \tag{1}$$

This was possible because:

- norm$(\underline{v}) = 1$, since $\underline{v}$ is an unit vector;
- norm$(\underline{\rho}) = \sqrt{\underline{\rho}^2} = \sqrt{\theta^2 \cdot \underline{v}^2} = \theta \cdot$ norm$(\underline{v}) = \theta$

From this, given that $\underline{\rho} = \underline{v} \cdot \theta$, $\underline{v}$ was calculated as:

$$\underline{v} = \frac{\rho}{\theta} \tag{2}$$

For each angle-axis representation, the corresponding orientation matrix was computed and the rotation was plotted.

2.1 Q1.1

The function *ComputeAngleAxis()* was used to obtain the orientation matrix from the angle-axis representation. Given θ and $\underline{v}$ as inputs, the function first computed the skew-symmetric matrix associated to the vector $\underline{v}$ as an operator as follows:

$$[\underline{v}x] = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \tag{3}$$

Then, with the Rodrigues formula, the orientation matrix was computed as:

$$R = I_{3x3} + sin(\theta) \cdot [\underline{v}x] + (1 - cos(\theta)) \cdot [\underline{v}x]^2 \tag{4}$$

2.2 Q1.2

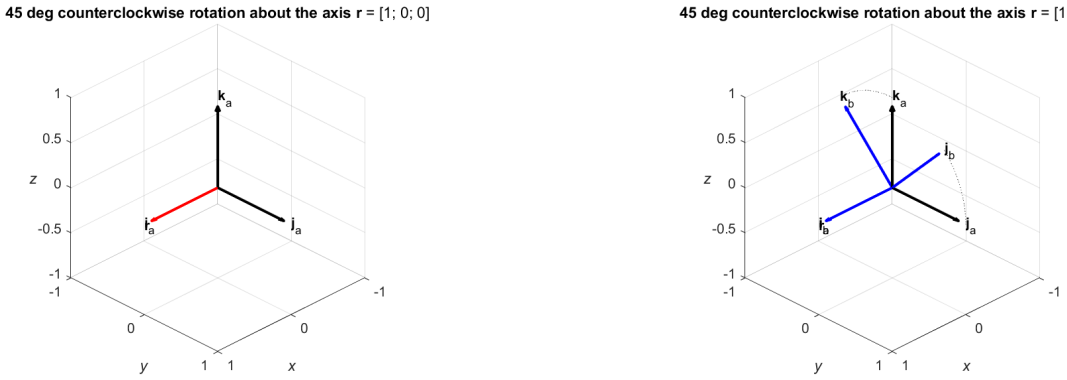
The given angle-axis representation was:

- $\underline{v}$ = [1 0 0];
- θ = 45°

First, the angle θ was converted to radians, obtaining $\theta = \pi/4$. Then, the orientation matrix ${}^a_bR$ was computed using the *ComputeAngleAxis()* function. The obtained orientation matrix is:

$${}^a_bR = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.7071 & -0.7071 \\ 0 & 0.7071 & 0.7071 \end{bmatrix} \tag{5}$$

The graphical result, with the comparison between the initial and final configuration, is shown in Figure 2.



(a) (b)

Figure 2: Initial and final configuration

2.3 Q1.3

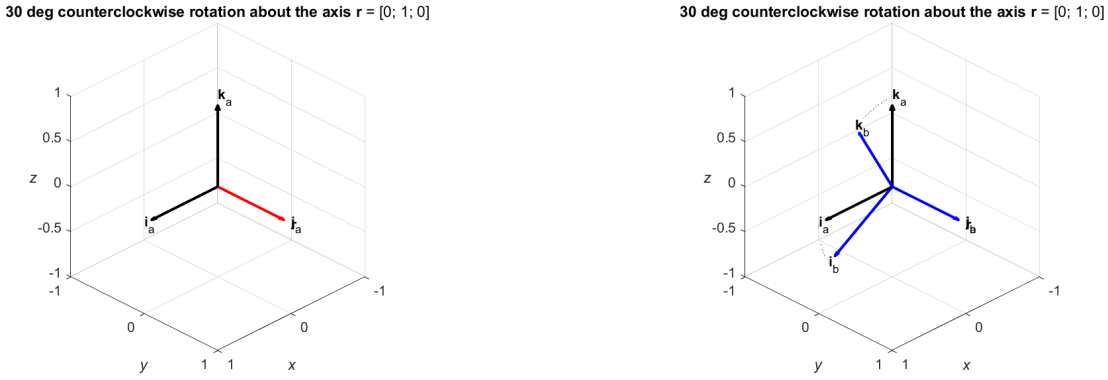
The given angle-axis representation was:

- $\underline{v}$ = [0 1 0];
- $\theta = \pi/6$

The orientation matrix ${}^a_bR$ was computed using the *ComputeAngleAxis()* function. The obtained orientation matrix is:

$$ {}^a_bR = \begin{bmatrix} 0.866 & 0 & 0.5 \\ 0 & 1 & 0 \\ -0.5 & 0 & 0.866 \end{bmatrix} \tag{6} $$

The graphical result, with the comparison between the initial and final configuration, is shown in Figure 3.



(a) (b)

Figure 3: Initial and final configuration

2.4 Q1.4

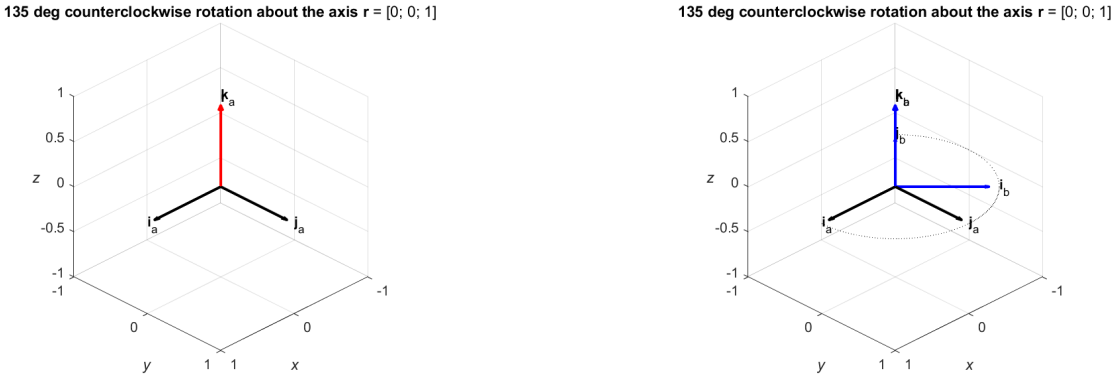
The given angle-axis representation was:

- $\underline{v}$ = [0 0 1];
- $\theta = 3\pi/4$

The orientation matrix ${}^a_bR$ was computed using the *ComputeAngleAxis()* function. The obtained orientation matrix is:

$$ {}^a_bR = \begin{bmatrix} -0.7071 & -0.7071 & 0 \\ 0.7071 & -0.7071 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7} $$

The graphical result, with the comparison between the initial and final configuration, is shown in Figure 4.



(a) (b)

Figure 4: Initial and final configuration

2.5 Q1.5

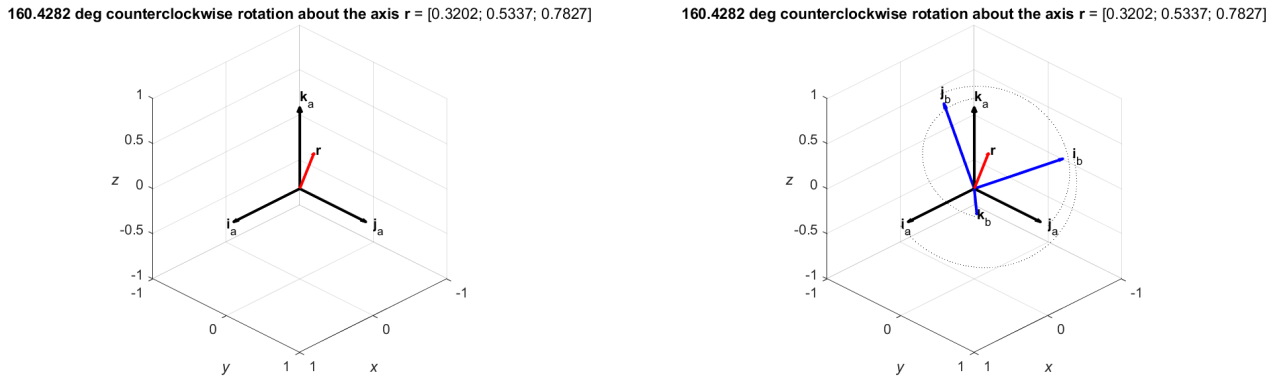
The given angle-axis representation was:

- $\underline{v}$ = [0.3202 0.5337 0.7827];
- $\theta = 2.8$

The orientation matrix ${}^{a}_{b}R$ was computed using the *ComputeAngleAxis()* function. The obtained orientation matrix is:

$$ {}^{a}_{b}R = \begin{bmatrix} -0.7431 & 0.0697 & 0.6655 \\ 0.5941 & -0.3890 & 0.7041 \\ 0.3080 & 0.9186 & 0.2477 \end{bmatrix} \tag{8} $$

The graphical result, with the comparison between the initial and final configuration, is shown in Figure 5.



(a) (b)

Figure 5: Initial and final configuration

2.6 Q1.6

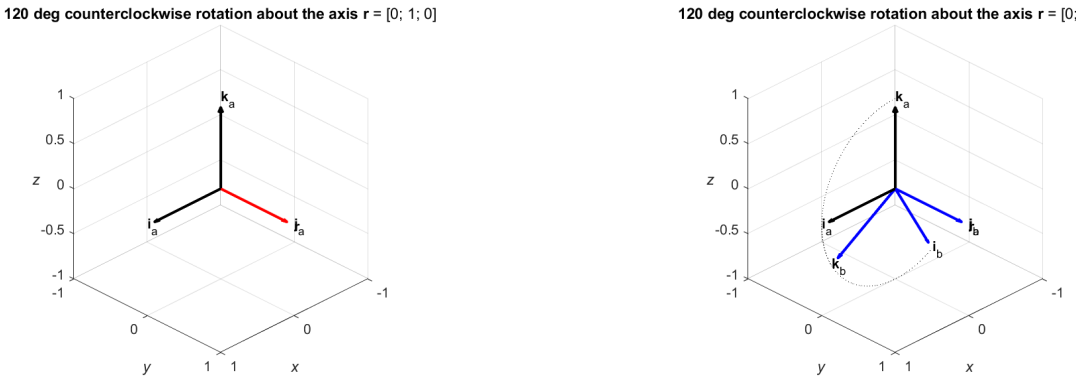
In this exercise, the vector $\rho = [0\ 2\pi/3\ 0]$ was given instead of $\underline{v}$ and θ. Using Equation 1 and Equation 2, we obtained:

- θ = 2.0944;
- $\underline{v}$ = [0 1 0]

Then, the orientation matrix ${}^a_bR$ was computed using the *ComputeAngleAxis()* function. The obtained orientation matrix is:

$$ {}^a_bR = \begin{bmatrix} -0.5 & 0 & 0.866 \\ 0 & 1 & 0 \\ -0.866 & 0 & -0.5 \end{bmatrix} \tag{9} $$

The graphical result, with the comparison between the initial and final configuration, is shown in Figure 6.



(a) (b)

Figure 6: Initial and final configuration

2.7 Q1.7

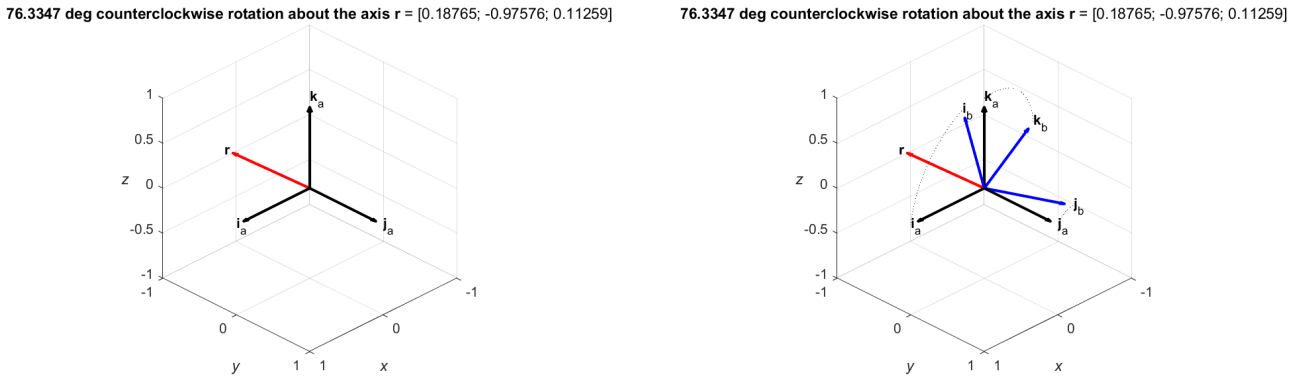
In this exercise, the vector $\rho = [0.25 \ -1.3 \ 0.15]$ was given instead of $\underline{v}$ and θ. Using Equation 1 and Equation 2, we obtained:

- θ = 1.3323;
- $\underline{v}$ = [0.1876 -0.9758 0.1126]

Then, the orientation matrix ${}^a_bR$ was computed using the *ComputeAngleAxis()* function. The obtained orientation matrix is:

$$ {}^a_bR = \begin{bmatrix} 0.2631 & -0.2492 & -0.932 \\ -0.0304 & 0.9634 & -0.2662 \\ 0.9643 & 0.0984 & 0.2459 \end{bmatrix} \tag{10} $$

The graphical result, with the comparison between the initial and final configuration, is shown in Figure 7.



(a) (b)

Figure 7: Initial and final configuration

2.8 Q1.8

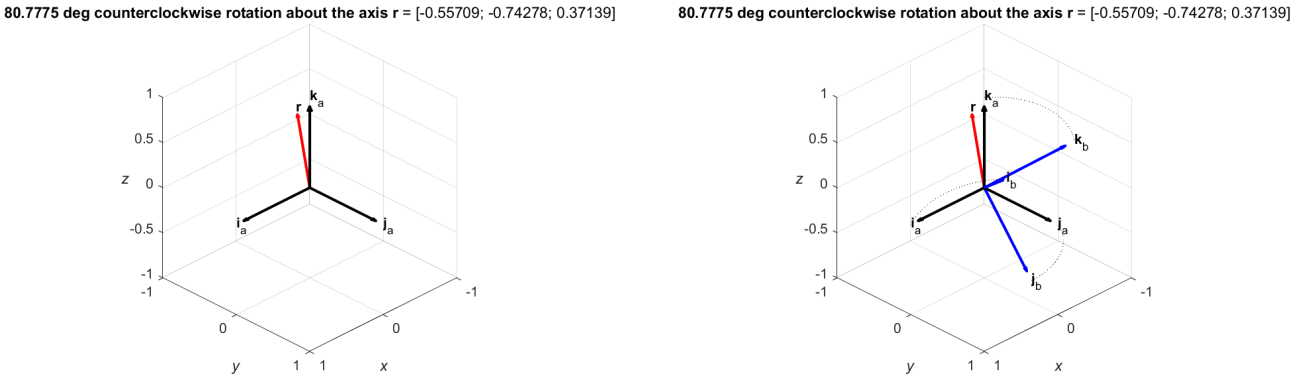
In this exercise, the vector $\rho = [-\pi/4 \ -\pi/3 \ \pi/6]$ was given instead of $\underline{v}$ and θ. Using Equation 1 and Equation 2, we obtained:

- θ = 1.4098;
- $\underline{v}$ = [-0.5571 -0.7428 0.3714]

Then, the orientation matrix ${}^a_bR$ was computed using the *ComputeAngleAxis()* function. The obtained orientation matrix is:

$$ {}^a_bR = \begin{bmatrix} 0.4209 & -0.0191 & -0.9069 \\ 0.7141 & 0.6236 & 0.3182 \\ 0.5594 & -0.7815 & 0.2761 \end{bmatrix} \tag{11} $$

The graphical result, with the comparison between the initial and final configuration, is shown in Figure 8.



(a) (b)

Figure 8: Initial and final configuration

3 Exercise 2

In the second exercise, we addressed the Inverse Equivalent Angle-Axis Problem. To this purpose, different rotation matrices of the type ${}^a_bR$ were given and the corresponding angle-axis parameters θ and $\underline{v}$ had to be computed.

3.1 Q2.1

Firstly, the frames shown in Figure 1 were considered. To compute the requested rotation matrix ${}^a_bR$, the property of multiple rotations was used as follows:

$$ {}^a_bR = {}^a_wR \cdot {}^w_bR \tag{12} $$

Moreover, it was evident that the difference between frame $< w >$ and frame $< a >$ was only a displacement of their respective origins, while the orientations were the same. Therefore, ${}^a_wR = I_{3\times3}$.
Consequently, Equation 12 could be simplified as follows:

$$ {}^a_bR = {}^w_bR \tag{13} $$

As the next step, ${}^w_bR$ was found without the Rodrigues Formula as required, but only by looking at Figure 1. We projected the axes of $< b >$ on frame $< w >$ obtaining the following result:

$$ {}^w_bR = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{14} $$

Thus, ${}^a_bR$ was found.

3.2 Q2.2

The Equivalent Angle-Axis Problem for the orientation matrix ${}^a_bR$ was solved by defining the *ComputeInverseAngleAxis()* function in Matlab. Firstly, the function verifies that the input matrix R belongs to $SO(3)$ and so if it is an orientation matrix. In order to do that, three checks were implemented:

- Check if R is a 3×3 matrix
- Check if R is orthogonal, so if $R \cdot R^T = I_{3\times3}$
- Check if $det(R) = 1$

For the last two checks, we implemented the *isequaltol()* function, which verifies if two numbers are equal with a tolerance, defined as the third input of the function. *isequaltol()* was used for all the equality checks in the *ComputeInverseAngleAxis()* function.
If the three conditions were met, θ was firstly computed as:

$$\theta = cos^{-1}(\frac{Tr(R)-1}{2}) \tag{15}$$

Before computing $\underline{v}$, the value of θ was checked to detect singularities. Particularly:

- $\theta = 0$: infinite number of solutions
- $\theta = \pi$: two possible solutions

In these two cases, the function returns an error message. Otherwise, $\underline{v}$ was computed. To this purpose, the Matlab function *eig(R)* was used to obtain the eigenvalues and eigenvectors of the orientation matrix R.

Then, $\underline{v}$ was found as the eigenvector corresponding to the eigenvalue equal to 1.
Finally, a final check was implemented to disambiguate the sign representing the direction of rotation. For this purpose, an orthogonal vector $\underline{u} \perp \underline{v}$ was found by considering the first column of the orthonormal basis associated with $\underline{v}$, computed with the *null()* MATLAB function. Then, the sign of rotation was found by applying the following formula:

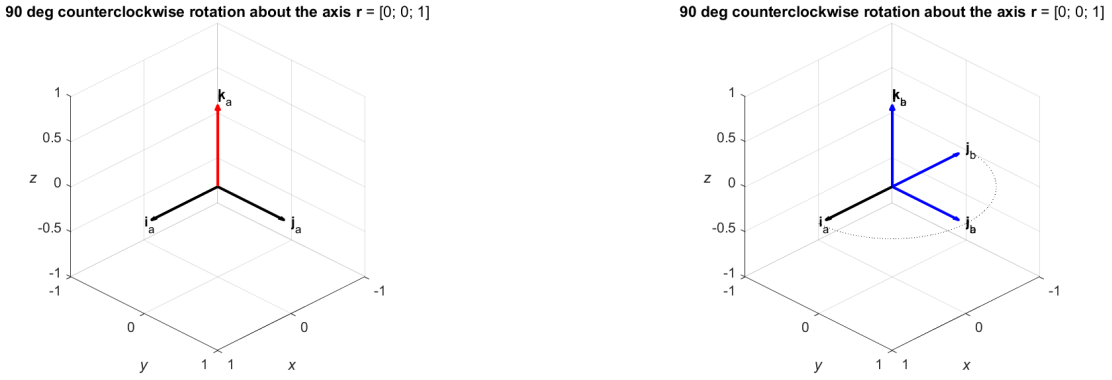
$$\underline{v} \cdot (\underline{u} \times R\underline{u}) \tag{16}$$

If Equation 16 < 0, *ComputeInverseAngleAxis()* returned $-\underline{v}$ instead of $\underline{v}$.

The finalized function *ComputeInverseAngleAxis()* was then used to solve the Inverse Equivalent Angle-Axis Problem for the orientation matrix ${}^a_bR$ defined in Equation 14. The following parameters were obtained:

- $\theta = 1.5708$
- $\underline{v}$ = [0 0 1]

The corresponding rotation is shown in Figure 9.



(a) (b)

Figure 9: Initial and final configuration

3.3 Q2.3

Firstly, from the given transformation matrix ${}^{w}_{c}T$, the orientation matrix ${}^{w}_{c}R$ was extracted:

$$ {}^{w}_{c}R = \begin{bmatrix} 0.835959 & -0.283542 & -0.46986 \\ 0.271321 & 0.957764 & -0.0952472 \\ 0.47703 & -0.0478627 & 0.877583 \end{bmatrix} \tag{17} $$

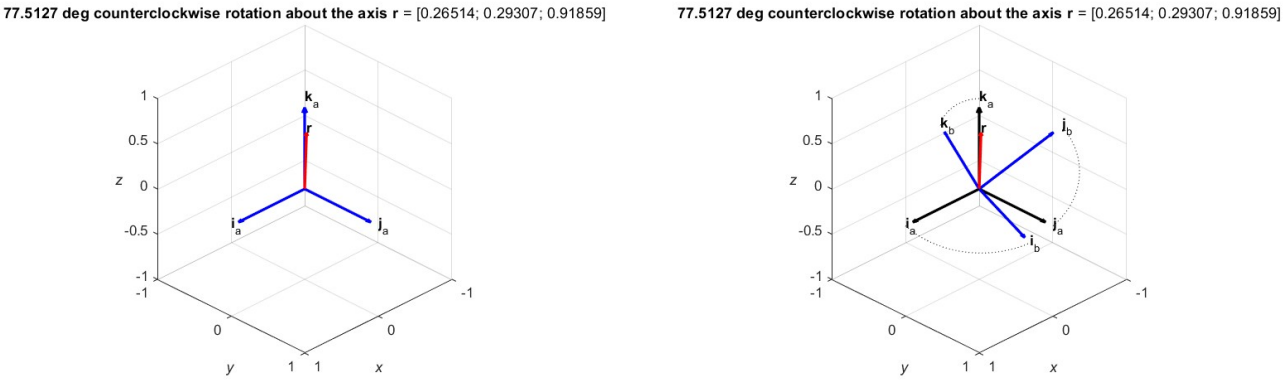
Then, the orientation matrix ${}^{c}_{b}R$ was computed with the multiple rotation property:

$$ {}^{c}_{b}R = {}^{w}_{c}R^{T} \cdot {}^{w}_{b}R = {}^{c}_{w}R \cdot {}^{w}_{b}R \tag{18} $$

Lastly, the corresponding angle-axis parameters were found using the *ComputeInverseAngleAxis()* function. The following parameters were obtained:

- $\theta = 1.3529$
- $\underline{v}$ = [0.2651 0.2931 0.9186]

The corresponding rotation is shown in Figure 10.



(a) (b)

Figure 10: Initial and final configuration

4 Exercise 3

In the third exercise, three elementary rotations from a world frame $< w >$ to a generic frame $< b >$ were given, with the requirement to define the associated orientation matrices. For this purpose, two solutions could be chosen:

1. Use the *ComputeAngleAxis()* function, since both the angle and axis of rotation were given for each elementary rotation;
2. Define the orientation matrices manually by using only the angles of rotation

Given the provided hint and the following tasks of the exercise, we opted for the second option.

Then, the computed orientation matrices had to be used to find the corresponding angle-axis representation for each elementary rotation.

Finally, the three elementary rotations had to be composed in sequence to find the:

- Tait-Bryan angles orientation z-y-x (yaw-pitch-roll sequence)
- Proper Euler angles orientation z-x-z

4.1 Q3.1

In this section, the orientation matrices corresponding to the three elementary rotation were calculated.

1. Given $\psi = \pi/4$, the corresponding orientation matrix ${}^w_b R_z$ around the z-axis was computed as:

$$\begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{19}$$

2. Given $\theta = \pi/3$, the corresponding orientation matrix ${}^w_b R_y$ around the y-axis was computed as:

$$\begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{20}$$

3. Given $\phi = -\pi/6$, the corresponding orientation matrix ${}^w_b R_x$ around x-axis was computed as:

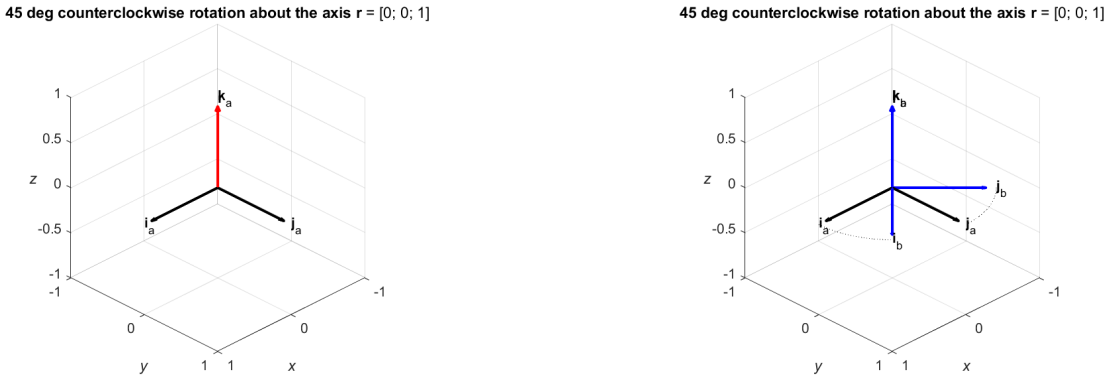
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix} \tag{21}$$

4.2 Q3.2

In this section, the three previously computed orientation matrices (namely, ${}^w_b R_z$, ${}^w_b R_y$ and ${}^w_b R_x$) were used to obtain the corresponding angle-axis representations for each one. For this purpose, the function *ComputeInverseAngleAxis()* was used.
As expected, the following results were obtained:

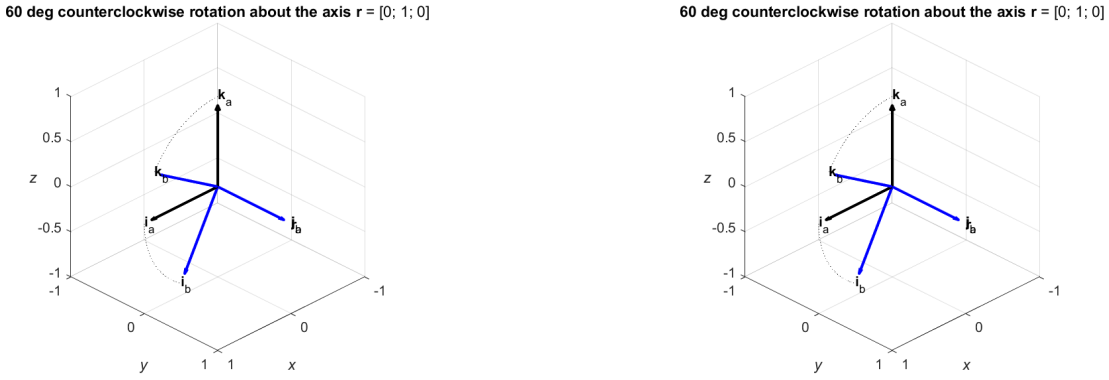
1. Elementary rotation of 45° around the z-axis (plot in Figure 11):
   - θ = $\pi/4$
   - v = [0 0 1]
2. Elementary rotation of 60° around the y-axis (plot in Figure 12):
   - θ = $\pi/3$
   - v = [0 1 0]
3. Elementary rotation of -30° around the x-axis (plot in Figure 13):
   - θ = -$\pi/6$
   - v = [1 0 0]

Please note that a clockwise rotation of a negative angle $-\theta$ around an axis $\underline{v}$ is equivalent to a counterclockwise rotation of the positive angle θ around the axis $-\underline{v}$.



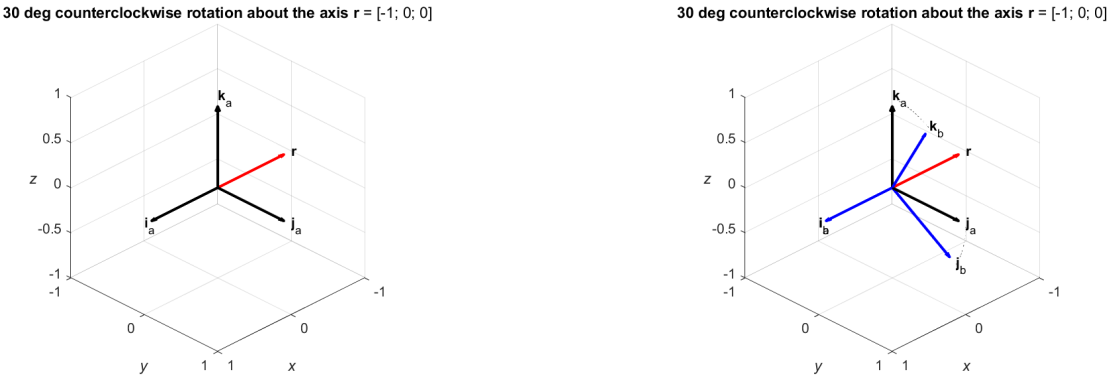
(a) (b)

Figure 11: Initial and final configuration of (1)



(a) (b)

Figure 12: Initial and final configuration of (2)



(a) (b)

Figure 13: Initial and final configuration of (3)

4.3 Q3.3

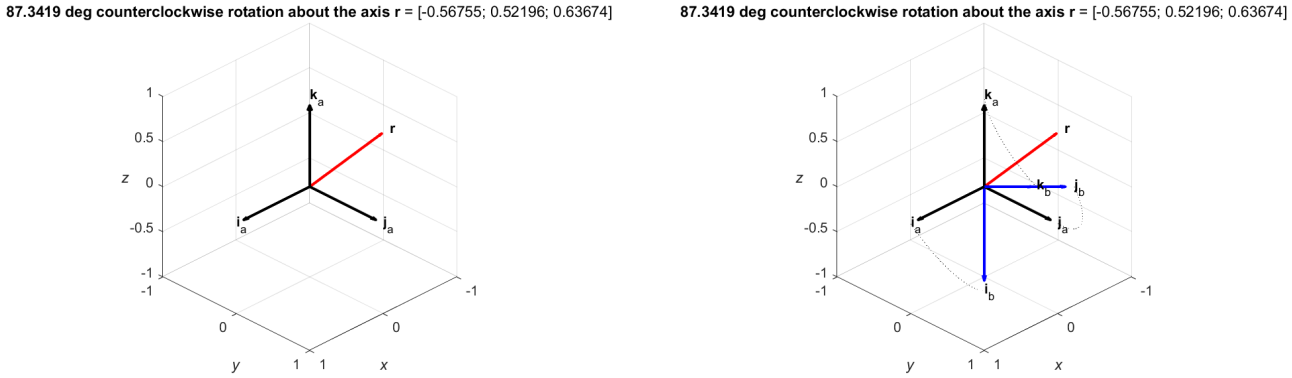
In this section, the yaw-pitch-roll (z-y-x) orientation matrix was calculated by multiplying the three following matrices, obtained in Section 4.1:

$$R_{zyx} = {}^{w}_{b}R_z \cdot {}^{w}_{b}R_y \cdot {}^{w}_{b}R_x \tag{22}$$

Consequently, R_{zyx} was obtained as:

$$R_{zyx} = \begin{bmatrix} 0.3536 & -0.9186 & 0.1768 \\ 0.3536 & 0.3062 & 0.8839 \\ -0.8660 & -0.2500 & 0.4330 \end{bmatrix} \tag{23}$$

The corresponding graphical rotation, obtained after computing the associated angle-axis representation, is shown in Figure 14.



(a) (b)

Figure 14: Initial and final configuration

4.4 Q3.4

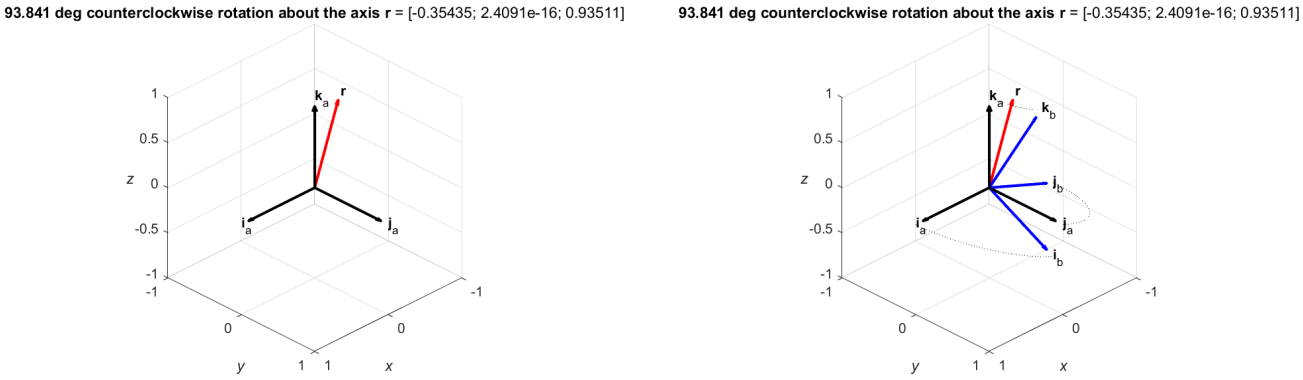
In this section, the z-x-z orientation matrix was calculated by multiplying the following matrices, obtained in Section 4.1:

$$R_{zxz} = {}^{w}_{b}R_z \cdot {}^{w}_{b}R_x \cdot {}^{w}_{b}R_z \tag{24}$$

Consequently, R_{zxz} was obtained as:

$$R_{zxz} = \begin{bmatrix} 0.0670 & -0.9330 & -0.3536 \\ 0.9330 & -0.0670 & 0.3536 \\ -0.3536 & -0.3536 & 0.8660 \end{bmatrix} \tag{25}$$

The corresponding graphical rotation, obtained after computing the associated angle-axis representation, is shown in Figure 15.



(a) (b)

Figure 15: Initial and final configuration

5 Exercise 4

In this exercise, a quaternion was given, with the requirement of creating a function *quatToRot()* to compute the associated orientation matrix. For this purpose, we took into consideration the composition of a quaternion, which is:

$$\underline{q} = \begin{bmatrix} \mu \\ \underline{\epsilon} \end{bmatrix} \tag{26}$$

Then, the Inverse Equivalent Angle-Axis problem was solved for the obtained orientation matrix by using the *ComputeInverseAngleAxis()* function.

Finally, both the quaternion and the orientation matrix were computed again with the built-in MATLAB functions *quaternion()* and *quat2rotm()* and the results were compared to verify the correctness of the *quatToRot()* function.

5.1 Q4.1

In this section, the quaternion representation $q = 0.1647 + 0.31583i + 0.52639j + 0.77204k$ was given. As the first task, the corresponding orientation matrix was computed by defining and using the function *quatToRot()*. This function took as input the quaternion parameters and returned the associated orientation matrix. In order to do that, the skew-symmetric matrix for vector ϵ was first created as:

$$[\underline{\epsilon x}] = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{27}$$

Then, considering $\mu = q_0$, the orientation matrix was computed with the Rodrigues formula:

$$R = I_{3\times 3} + 2 \cdot \mu \cdot [\underline{\epsilon x}] + 2 \cdot [\underline{\epsilon x}]^2 \tag{28}$$

By applying the function *quatToRot()* to the given quaternion, the following orientation matrix was obtained:

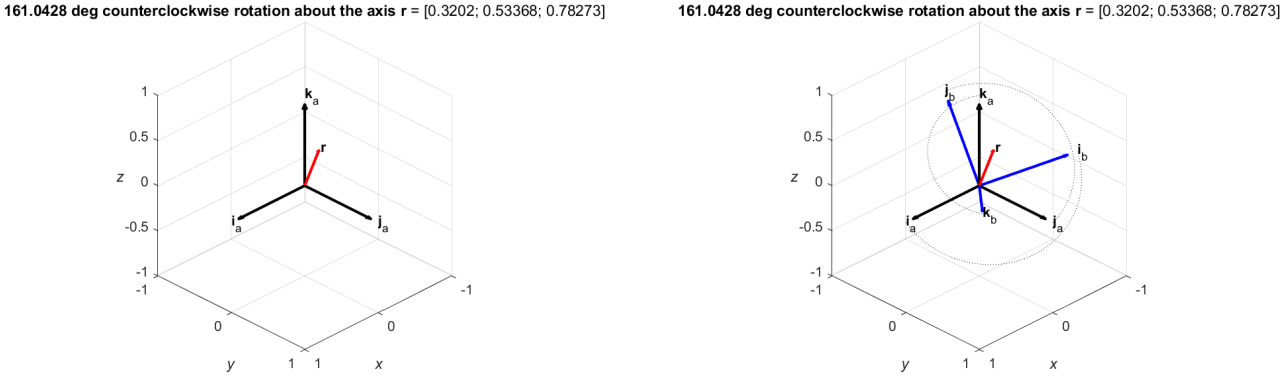
$$R = \begin{bmatrix} -0.7463 & 0.0782 & 0.6611 \\ 0.5868 & -0.3916 & 0.7088 \\ 0.3143 & 0.9168 & 0.2463 \end{bmatrix} \tag{29}$$

5.2 Q4.2

In this section, the angle-axis parameters corresponding to the orientation matrix R were found as:

- $\theta = 2.8107$
- v = [0.3202 0.5337 0.7827]

The corresponding graphical rotation is shown in Figure 16.



(a) (b)

Figure 16: Initial and final configuration

5.3 Q4.3

In this section, the built-in MATLAB functions *quaternion()* and *quat2rotm()* were used to check if the results obtained in Sections 5.1 and 5.2 were correct.

After building the quaternion vector q with the *quaternion()* function, the orientation matrix R was obtained:

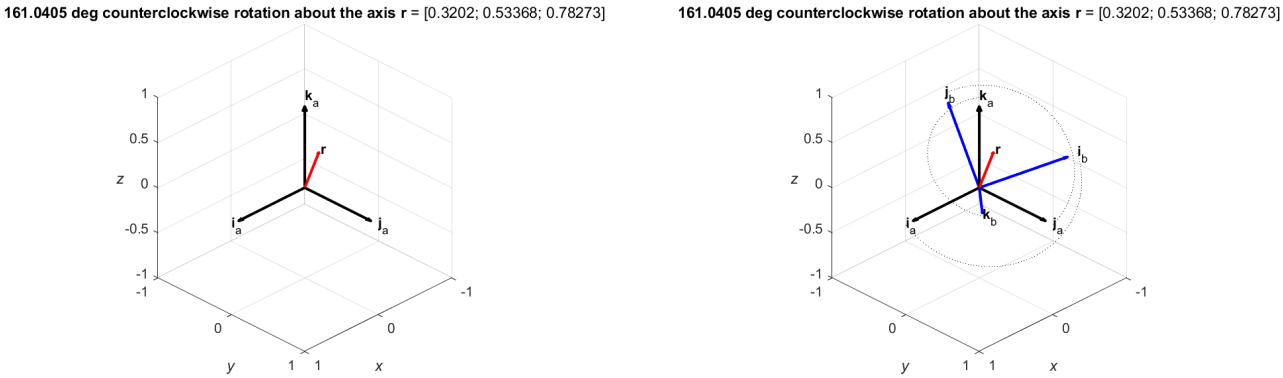
$$R = \begin{bmatrix} -0.7463 & 0.0782 & 0.6611 \\ 0.5868 & -0.3916 & 0.7087 \\ 0.3143 & 0.9168 & 0.2463 \end{bmatrix} \tag{30}$$

Only a minimum difference was shown: $R(2,3) = 0.7087$ instead of 0.7088. Therefore, the results obtained in Section 5.1 were proven right.

The corresponding angle-axis parameters were then computed and the following results were obtained:

- $\theta = 2.8107$
- v = [0.3202 0.5337 0.7827]

Since no difference was shown, the results obtained in Section 5.2 were proven right. Only in the conversion from radians to degrees the angle of rotation was slightly different, as shown in the plotted rotation in Figure 17.



(a) (b)

Figure 17: Initial and final configuration

6 Appendix

[Comment] Add here additional material (if needed)

6.1 Appendix A

6.2 Appendix B