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:

Colomba Ilaria
Ierardi Ambra
Malatesta Federico

Student ID:

s4829201
s4843302
s4803603

Professors:

Enrico Simetti
Giorgio Cannata

Tutors:

Andrea Tiranti
Francesco Giovinazzo
George Kurshakov

October 31, 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 ${}^c_bR$.

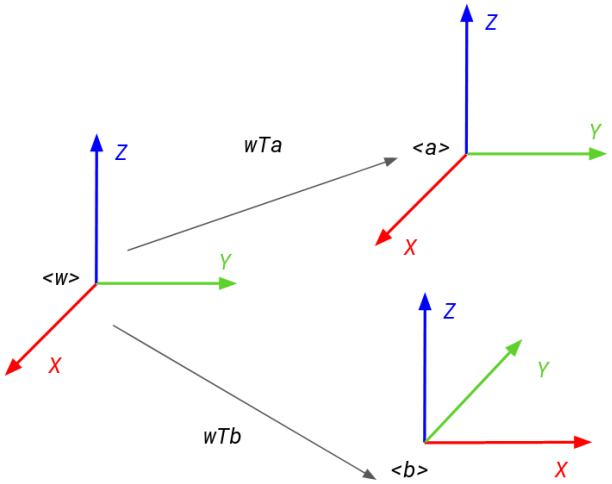


Figure 1: exercise 2 frames

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 $< w >$, in the following three cases:

- $< b >$ 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 considering a rotation of $45°$, $60°$, $-30°$ and solve the Inverse Equivalent Angle-Axis Problem for the obtained orientation matrix
Q3.4 Compute the z-x-z representation considering a rotation of $45°$, $-30°$, $45°$ 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 of this assignment the students were asked to implement and test an algorithm to obtain the **Equivalent Angle-Axis Representation**, which is also known as the **Exponential representation**.
All the following images and animations were plotted thanks to the given function **plotRotation.m**, which provides an animation of the rotation of the frame from the initial configuration to the final one.

2.1 Q1.1

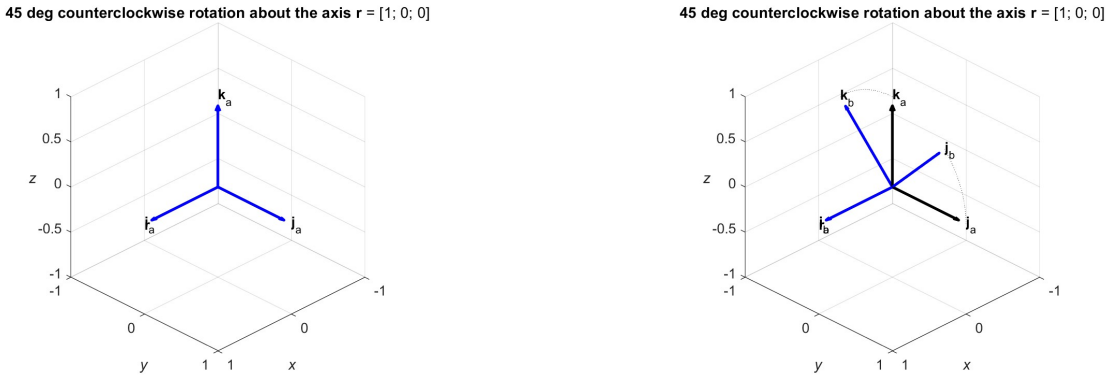
In this point it was required to implement the $Rodrigues\ formula$, to compute the **orientation matrix** of a generic frame $< b >$ with respect to another frame $< a >$. The required inputs to the **ComputeAngleAxis.m** function are θ, the angle of rotation, and **v**, which is the vector of the direction of the axis around which the rotation takes place. The $Rodrigues\ formula$ is the following one:

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

where $\mathbf{I}_{3x3}$ is the $3\ by\ 3$ identity matrix and x is the vector product operator.
This equation was written in a MATLAB code in order to obtain the desired matrix.
In the following points the developed algorithm is tested with different couples of vector **v** and angle θ.

2.2 Q1.2

The tested vector is $\mathbf{v} = [1, 0, 0]$ and the corresponding angle is $\theta = 45° = \pi/4$. The initial and the final configurations are the following ones:



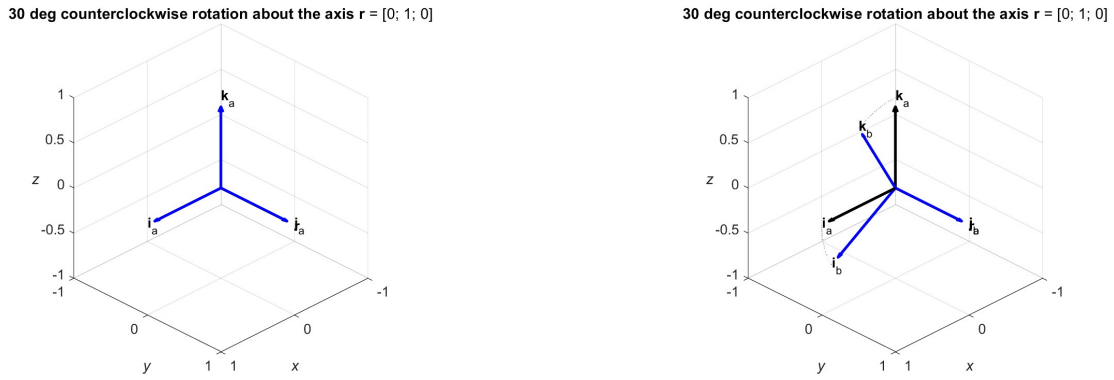
(a) (b)

Figure 2: (a) Initial configuration. (b) Final configuration.

This rotation, and all the following ones in this first exercise, was obtained by giving as input of the implemented function **ComputeAngleAxis.m** the previously defined vector and the relative angle. This way the end configuration is a frame rotated of $45°$ counterclockwise around the $x - axis$.

2.3 Q1.3

The second tested vector is $\mathbf{v} = [0, 1, 0]$ and the relative angle is $\theta = \pi/6 = 30°$.
The two frames are:



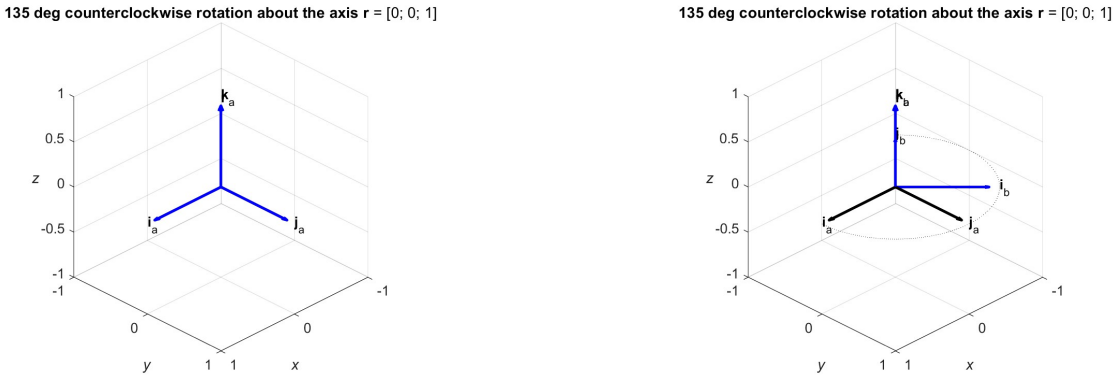
(a) (b)

Figure 3: (a) Initial configuration. (b) Final configuration.

The result of this rotation, as it can be observed in the Figure 3, is a rotation of $30°$ counterclockwise around the $y - axis$.

2.4 Q1.4

In the fourth point, the vector $\mathbf{v} = [0, 0, 1]$ and the angle $\theta = 3\pi/4$ were tested, which led to the following transformation, from the configuration $< a >$ to $< b >$:



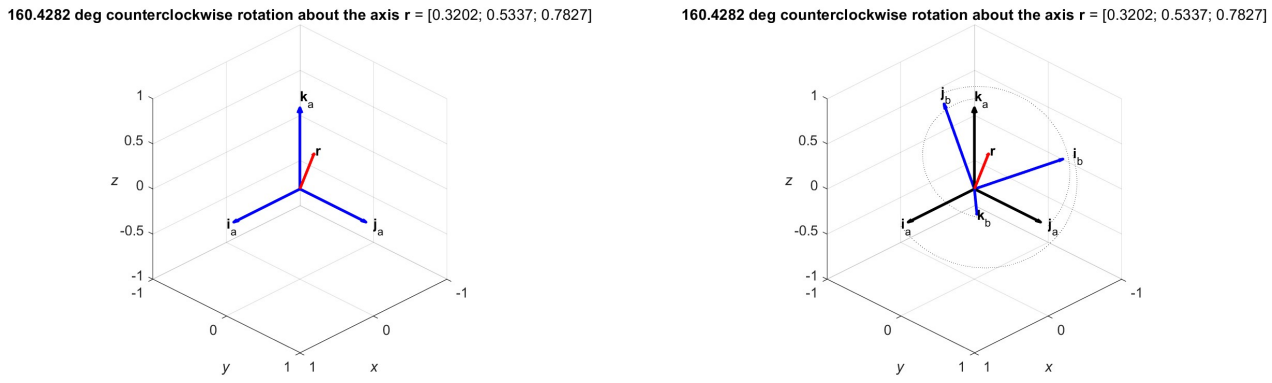
(a) (b)

Figure 4: (a) Initial configuration. (b) Final configuration.

This brought to a new orientation of the reference frame which is rotated about $3\pi/4$, or $135°$, counterclockwise around the $z - axis$.

2.5 Q1.5

In this point, the vector tested was **v** $= [0.3202, 0.5337, 0.7827]$ and the angle was $\theta = 2.8$. Note that the norm of **v** is still 1, so no change has to be done to compute the matrix.
The two configurations are:



(a) (b)

Figure 5: (a) Initial configuration. (b) Final configuration.

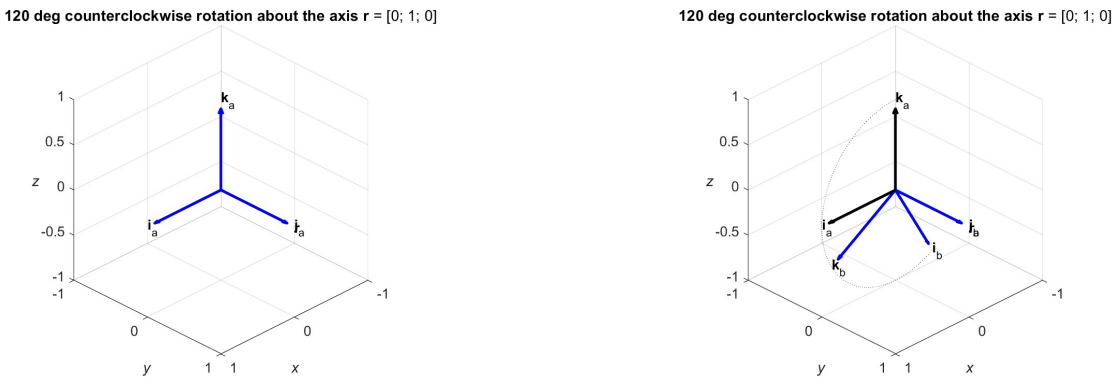
The axis around which the rotation is made is highlighted in red in Figure 5; it can be noticed that it does not correspond to any of the three fundamental axis. The angle is 2.8, which is expressed in radians. Converted in degrees with this ratio:

$$\alpha = 57,2958 \frac{\theta}{rad} \tag{2}$$

where θ is the angle in radians and α is the result in degrees, it corresponds approximately to $160.43°$. The rotation is once again counterclockwise.

2.6 Q1.6

The provided vector in this point is $\boldsymbol{\rho} = [0, 2\pi/3, 0]$;. In this case, it is necessary to consider that $\boldsymbol{\rho}$ is $\boldsymbol{\rho} = \mathbf{v} * \theta$. The vector **v** must be unitary length, so, in order to find it, $\boldsymbol{\rho}$ must be divided by the norm of itself, that is $2\pi/3$. This way, it was obtained a vector **v** which is unitary length and the angle θ is equal to the norm previously computed, that was $2\pi/3$.
At this point, it is possible to compute the new frame, starting from these data:



(a) (b)

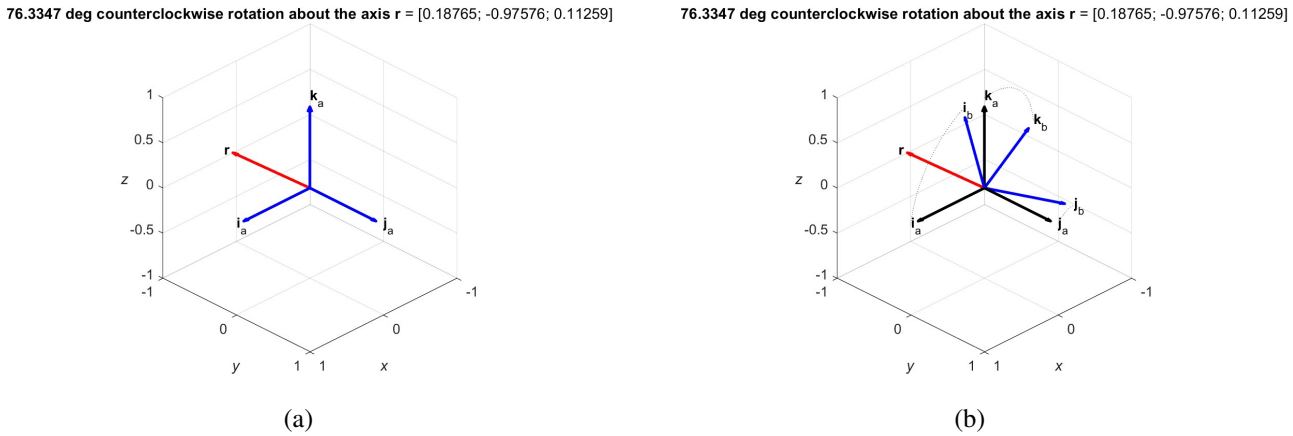
Figure 6: (a) Initial configuration. (b) Final configuration.

The rotation is around the $y - axis$, about $120°$ counterclockwise.

2.7 Q1.7

Also in this case, the provided data include the vector $\boldsymbol{\rho}$, not **v** and θ separately: $\boldsymbol{\rho} = [0.25, -1.3, 0.15]$.
The procedure is:

- first compute $\theta = \|\boldsymbol{\rho}\|$, where the norm operation consists in the square root of the sum of the squared components of the studied vector: $\|\boldsymbol{\rho}\| = \sqrt{\boldsymbol{\rho_x^2} + \boldsymbol{\rho_y^2} + \boldsymbol{\rho_z^2}}$;
- then get the **v** vector as: $\mathbf{v} = \frac{\boldsymbol{\rho}}{\theta}$



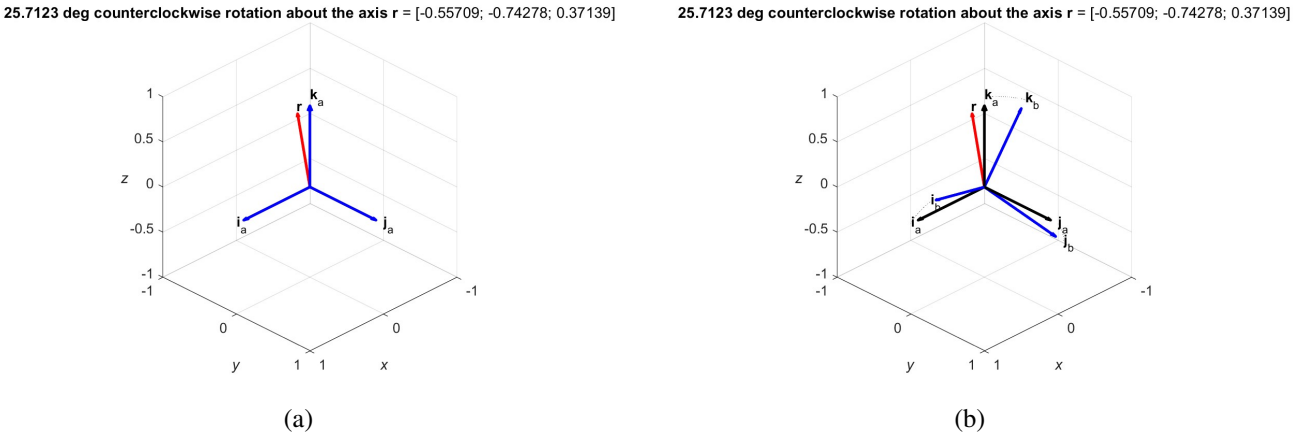
(a) (b)

Figure 7: (a) Initial configuration. (b) Final configuration.

The resulting angle $\theta = 1.3323$ and the vector is $\mathbf{v} = [0.1876, -0.9758, 0.1126]$. The rotation is about $76.33°$, once again counterclockwise.

2.8 Q1.8

In the last point the provided vector is $\boldsymbol{\rho} = [-\pi/4, -\pi/3, \pi/6]$. Computing the norm, it was found that the required values of the angle and the axis are $\theta = 0.448 = 25.71°$ and $\mathbf{v} = [-0.5571, -0.7428, 0.3714]$. The resulting transformation is:



(a) (b)

Figure 8: (a) Initial configuration. (b) Final configuration.

3 Exercise 2

The purpose of the second exercise is to compute the inverse problem: starting from an orientation matrix, in the function **ComputeInverseAngleAxis.m** it is computed the axis around which the frame rotates and the relative angle. Also, the two reference frames $< a >$ and $< b >$ are given and the frame $< w >$ is the common world frame. The scope of the task is to obtain the rotation from $< a >$ to $< b >$ considering the frame $< w >$.

3.1 Q2.1

For the first point of the exercise, the rotation matrix ${}^{w}_{a}R$, which corresponds to the rotation of frame $< a >$ with respect to $< w >$, has been initialized as an identity matrix, which has the following shape:

$$ {}^{w}_{a}R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

The shape of this orientation matrix means that the world frame is aligned to $< a >$.
Then, the rotation matrix from $< b >$ to $< w >$ has been represented by the following matrix, which consists of a rotation of $90°$ around the $z - axis$:

$$ {}^{w}_{b}R = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

At the end, it has been possible to calculate the final rotation matrix between frame $< b >$ and frame $< a >$. The result was calculated by using this formula:

$$ {}^{a}_{b}R = {}^{w}_{a}R^{T} \times {}^{w}_{b}R \tag{3} $$

in order to avoid using the Rodrigues formula, as specified in the exercise request. The obtained matrix is:

$$ {}^{a}_{b}R = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

This result is exact, since the frame $< a >$ is aligned with the world frame $< w >$, as previously mentioned, so the final rotation matrix ${}^{a}_{b}R$ has to be equal to ${}^{w}_{b}R$.

3.2 Q2.2

In the second part of this exercise, the goal is to solve the Inverse Equivalent Angle-Axis problem for the orientation matrix ${}^{a}_{b}R$. In this task, the angle θ and the vector **v** are computed by exploiting the orientation matrix, given as input of the **ComputeInverseAngleAxis.m** function. At first, it is necessary to perform some checks in order to control that:

- the size of the rotation matrix is 3x3;
- the rotation matrix is orthogonal, which means that each column and row are orthonormal, $i.e.$ the vectors are perpendiculars to each other and the magnitude of them is equal to 1, and this was verified by multiplying the rotational matrix by its transpose, and checking if the result was equal to an identity matrix;
- the determinant of the rotation matrix is equal to 1;

this way, it was verified that this matrix is **proper**, and so it is possible to proceed to solve the problem. At this point, it is possible to compute the angle with this formula:

$$ \theta = arccos((trace({}^{a}_{b}R) - 1)/2) \tag{4} $$

where the trace of the matrix is the sum of all the elements on the principal diagonal, computed thanks to the $trace$ function of Matlab; the axis are computed by using the eig Matlab function, which returns the eigenvectors and eigenvalues of the considered matrix: the rotational axis is the eigenvector associated to the eigenvalue of value equal to 1 of the rotational matrix. A further check has to be done since with this procedure there is no distinction between **v** and **-v**, or θ and -θ, because the direction along which the rotation is performed is the

same; the only thing that changes is the verse of the rotation, which can be clockwise or counterclockwise. In order to verify it, it is necessary to compute the rotational matrix starting from the computed θ and, for example, the negative axis, -**v**, and to check the result: if it matches the original orientation matrix, this means that the right angle of rotation is the negative one, and so the transformation will be clockwise; otherwise, it is the positive one, that leads to a counterclockwise rotation.
The results, for this specific case, are $\theta = 1.57rad \simeq 90°$ and $\mathbf{v} = [0, 0, 1]$, the $z - axis$.

3.3 Q2.3

The purpose of the third part of the second exercise is to compute the Inverse equivalent angle-axis problem for the orientation matrix ${}^{c}_{b}R$ given the transformation matrix:

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

Knowing from the theory that a transformation matrix is a notation used to represent both the mutual orientation between two frame and the position information, written in formula as:

$$ {}^{a}_{b}\mathbf{T} = \begin{bmatrix} {}^{a}_{b}\mathbf{R} & {}^{a}O_b \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} $$

in which ${}^{a}O_b$ is a vector containing the information of the position of the origin of the frame $< b >$ projected on the frame $< a >$.
It is possible to extract the correspondent rotation matrix which results to be:

$$ {}^{w}_{c}R = \begin{bmatrix} 0.8360 & -0.2835 & -0.4699 \\ 0.2713 & 0.9578 & -0.0952 \\ 0.4770 & -0.0479 & 0.8776 \end{bmatrix} $$

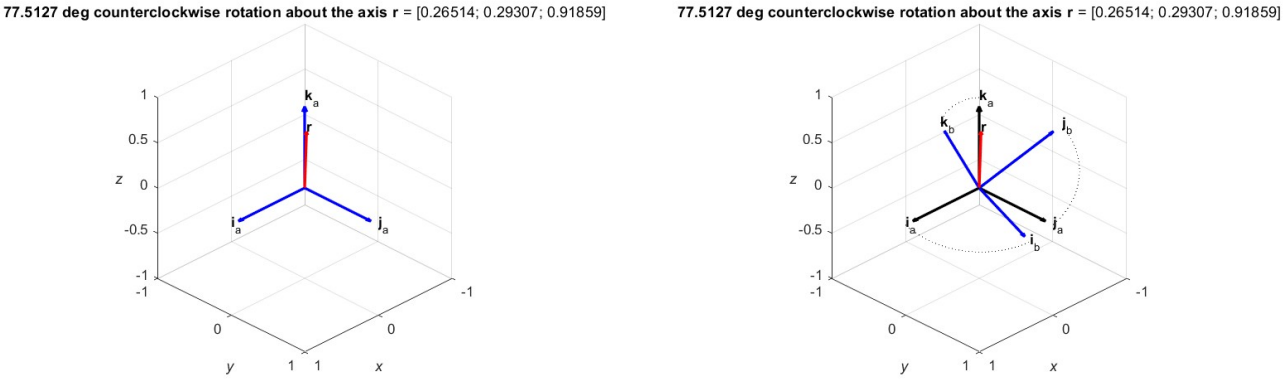
Finally the rotation matrix ${}^{c}_{b}R$ can be computed using the formula which comes from the definition of the frames in the second exercise:

$$ {}^{c}_{b}R = {}^{w}_{c}R^{T} \cdot {}^{w}_{b}R \tag{5} $$

The result expressed in numbers is as follows:

$$ {}^{c}_{b}R = \begin{bmatrix} 0.2713 & -0.8360 & 0.4770 \\ 0.9578 & 0.2835 & -0.0479 \\ -0.0952 & 0.4699 & 0.8776 \end{bmatrix} $$

The correspondent plot shown below is obtained using $\theta = 1.3529$ and $\mathbf{v} = [0.2651, 0.2931, 0.9186]$ computed from the previously described function **ComputeInverseAngleAxis.m**:



(a) (b)

Figure 9: (a) Initial configuration. (b) Final configuration.

4 Exercise 3

The purpose of the third exercise of the assignment is to see that any orientation can be achieved by composing three orientations around three different axis. So, any orientation matrix can be represented by the sequence of three elementary rotations. There are two possible choices:

- **Extrinsic orientation/rotation**:
  the rotation is composed around fixed axes of the coordinate system.
- **Intrinsic orientation/rotation**:
  the axes coordinates are not fixed. In this case the axes of a rotating coordinate system are initially aligned with the fixed ones. So there are a rotation around the $x - axis$, a rotation around y one and a rotation around z one and it is possible to choose different orders and combinations of these rotations, according to the desired goal rotation. There are different cases:
  1. **Proper Euler Angles**:
     where the first and the third rotations are repeated around the same axis Z-X-Z, Y-X-Y, X-Z-X;
  2. **Tait-Bryan Angles**:
     where all three rotations are around a different axis, for example X-Y-Z, Z-Y-X...

4.1 Q3.1

The first task of the exercise three is to define the elementary orientation matrices of frame $< b >$ with respect to the frame $< w >$. The value of rotation angles are given in the following way:

- rotation of $45°$ around the $z - axis$ from $< w >$ to $< b >$. The orientation matrix is:

$$ {}^{w}_{b}R_Z = \begin{bmatrix} \cos(45°) & -\sin(45°) & 0 \\ \sin(45°) & \cos(45°) & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

- rotation of $60°$ around the $y - axis$ from $< w$ to $< b >$. The rotation matrix is:

$$ {}^{w}_{b}R_Y = \begin{bmatrix} \cos(60°) & 0 & \sin(60°) \\ 0 & 1 & 0 \\ -\sin(60°) & 0 & \cos(60°) \end{bmatrix} $$

- rotation of $-30°$ around the $x - axis$ from $< w >$ to $< b >$. In this case the rotation matrix is:

$$ {}^{w}_{b}R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(-30°) & -\sin(-30°) \\ 0 & \sin 30°) & \cos(-30°) \end{bmatrix} $$

In $Matlab$ these rotation matrices are called respectively wRb_z, wRb_y and wRb_x. The results which are obtained are the following:

$$ wRb_z = \begin{bmatrix} 0.7071 & -0.7071 & 0 \\ 0.7071 & 0.7071 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix} $$

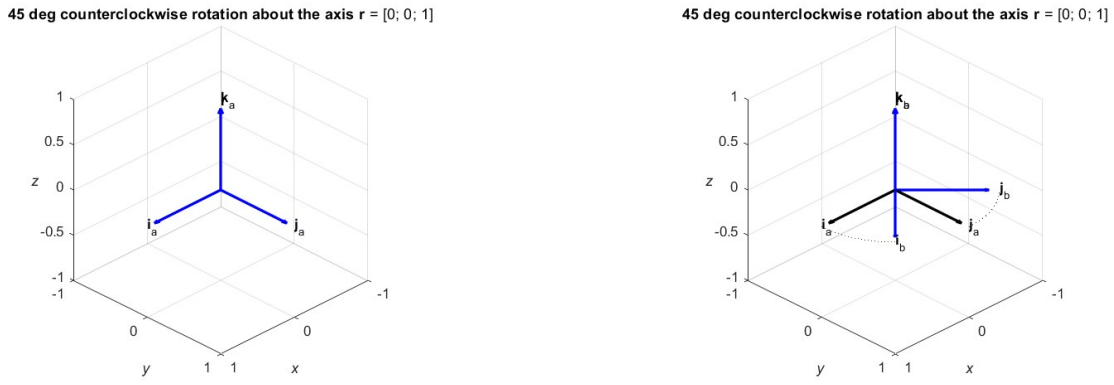
$$ wRb_y = \begin{bmatrix} 0.5000 & 0 & 0.8660 \\ 0 & 1.0000 & 0 \\ -0.8660 & 0 & 0.5000 \end{bmatrix} $$

$$ wRb_x = \begin{bmatrix} 1.0000 & 0 & 0 \\ 0 & 0.8660 & 0.5000 \\ 0 & -0.5000 & 0.8660 \end{bmatrix} $$

4.2 Q3.2

The purpose of the second part of the third exercise of the assignment is to compute the equivalent angle axis for each elementary rotation. In order to do that, the $Matlab$ code calls the function **ComputeInverseAngleAxis.m**. With this function, it is possible to compute the θ angle and the vector **v** which are the equivalent

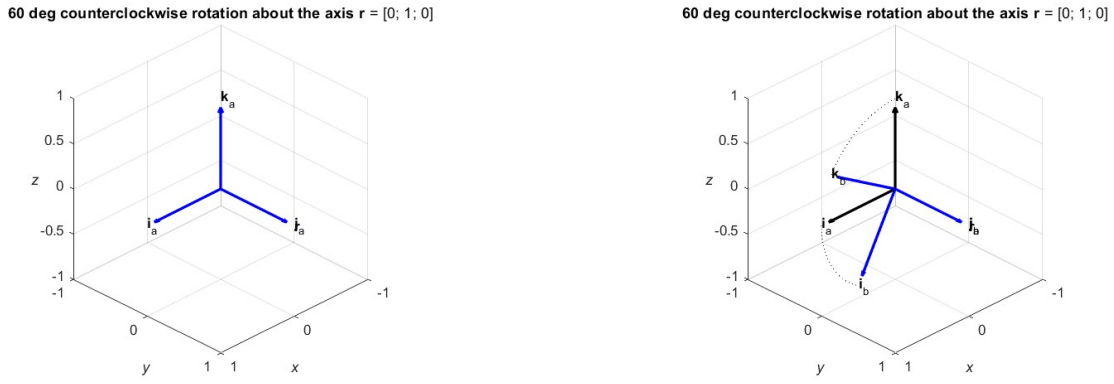
angle-axis representation elements of the rotation. The output of this $Matlab$ function is the couple of parameters θ and **v** given any orientation matrix R in input. The plots which are obtained are the following ones:



(a) (b)

Figure 10: (a) Initial configuration. (b) Final configuration.

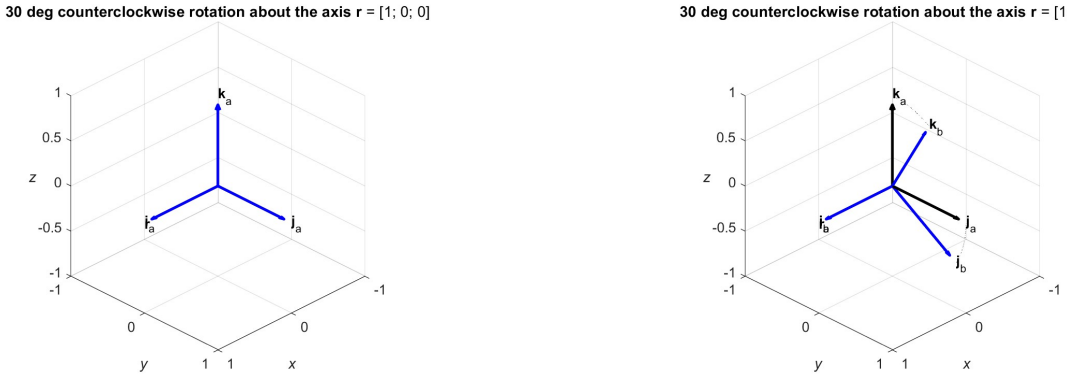
From this plot, it is possible to see a counterclockwise rotation of $45°$around the $z - axis$.



(a) (b)

Figure 11: (a) Initial configuration. (b) Final configuration.

In this plot, it is possible to see a counterclockwise rotation of $60°$ around the $y - axis$.



(a) (b)

Figure 12: (a) Initial configuration. (b) Final configuration.

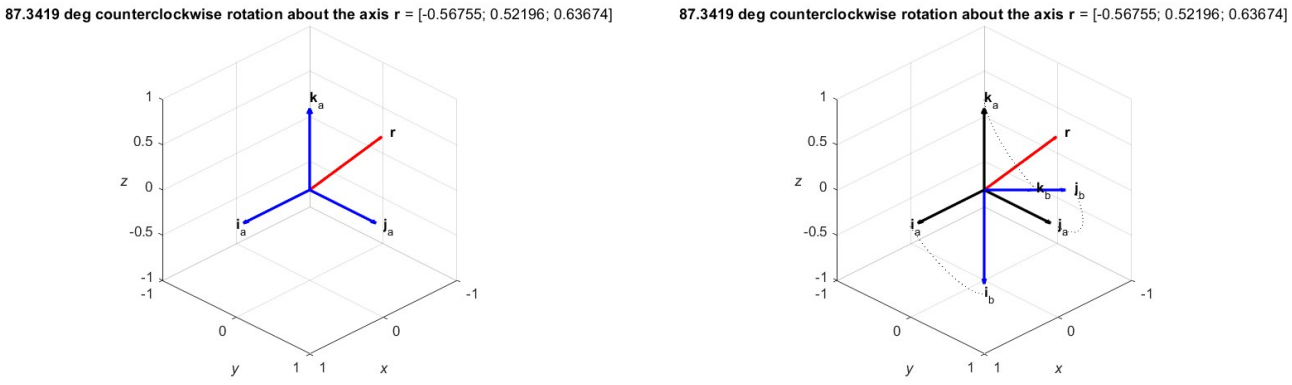
In this figure, it is possible to see a counterclockwise rotation of $-30°$around the $x - axis$.

4.3 Q3.3

The purpose of third part of the third exercise is to compute the rotation matrix using the **Tait-Bryan Angles** representation, with the sequence $z - y - x$. This rotation has been implemented using the formula:

$$R_{zyx} =_b^w R_z *_b^w R_y *_b^w R_x \tag{6}$$

Then the corresponding θ angle and vector **v** are computed by using the function **ComputeInverseAngleAxis.m**. The results obtained are presented below:



(a) (b)

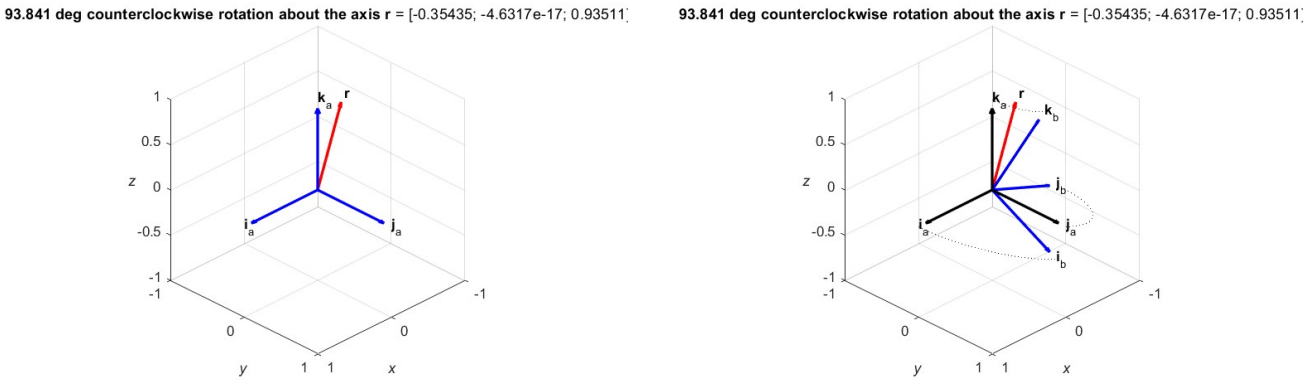
Figure 13: (a) Initial configuration. (b) Final configuration.

4.4 Q3.4

For the last task of the third exercise, the previous part has been repeated, but this time the rotation matrix is computed using the **Proper Euler Angles** with thw sequence $z-x-z$. The rotation matrix has been calculated with the following formula:

$$R_{zxz} =_b^w R_z *_b^w R_x *_b^w R_z \tag{7}$$

The results are the following:



(a) (b)

Figure 14: (a) Initial configuration. (b) Final configuration.

5 Exercise 4

In the fourth exercise the goal is to derive the rotation matrix corresponding to the given $quaternion$ (a mathematical element consisting of a scalar number, μ and a vector, ϵ). The given quaternion is the following one:

$$q = 0.1647 + 0.31583i + 0.52639j + 0.77204k \tag{8}$$

5.1 Q4.1

The first task is to define a **quatToRot.m** function which takes as input the quaternion and returns a $3x3$ matrix representing the rotation.

Given $\mathbf{q} = \begin{bmatrix} \mu \\ \epsilon \end{bmatrix}$

with μ scalar number and ϵ vector $\in \mathbb{R}^{3\times1}$, it is possible to obtain a new form of the $Rodrigues formula$, following from a series of mathematical steps, which is:

$$R = \mathbf{I}_{3\times3} + 2\mu[\boldsymbol{\epsilon}\times] + 2[\boldsymbol{\epsilon}\times]^2 \tag{9}$$

where x is again the vector product, and defining:

$$\mu = \cos\left(\frac{\theta}{2}\right) \qquad \epsilon = \sin\left(\frac{\theta}{2}\right)\mathbf{h} \tag{10}$$

where **h** is the unit length vector, of the original $Rodrigues\ formula$.
The result is the following one:

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

5.2 Q4.2

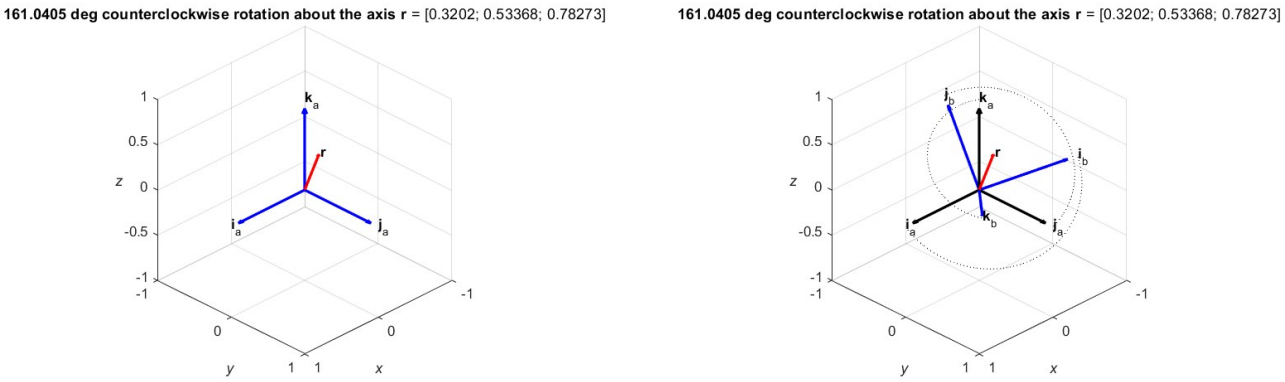
In this point, it is required to use the functions $quaternion$ and $rotmat$ available on $Matlab$ to compute the rotation matrix corresponding, in order to compare it with the one obtained using **quatToRot** and the result is:

$$ {}^{a}_{b}R = \begin{bmatrix} -0.7463 & 0.0782 & 0.6611 \\ 0.5868 & -0.3916 & 0.7088 \\ 0.3143 & 0.9168 & 0.2463 \end{bmatrix} $$

It turns out to be equal to the one calculated in the previous step.

5.3 Q4.3

The final request is to use the function **ComputeInverseAngleAxis**, with the R matrix computed at the step before in input, in order to obtain the equivalent angle-axis representation values θ and **v** which are necessary to plot the rotation using **plotRotation.m**.



(a) (b)

Figure 15: (a) Initial configuration. (b) Final configuration.

This is a complex rotation around the red axis.