Practical Work 1

Rigid Transformation

This practical work is designed to illustrate rigid transformation, consisting of a rotation and a translation in a 3D Euclidean space in Cartesian coordinates.

Preparation

We're using the Python language and the MatPlotLib library. The MatPlotLib library can be installed with the command:

```
pip install matplotlib
```

or for conda installation:

```
conda install -c conda-forge matplotlib
```

The MatPlotLib library displays geometric rendering in an independent interactive window. Depending on the Python environment, this window may be rendered as a frozen image, preventing user interaction. To correct this problem, here are a few solutions:

Jupyter Notebook:

Execute following code within the Jupyter Notebook:

```
%matplotlib qt
```

PyCharm

Go to `Settings / Tool / Python Plot` and uncheck the option `Show plots in tool windows`.

Spyder

Go to `Tools / Preferences / IPython console / Graphics / Backend:Inline` and change "`Inline`" to "`Automatic`". Click OK button and restart the IDE.

Getting started with MatPlotLib

The MatPlotLib library enables to display 2D or 3D geometries. The following code display an empty 3D rendering with

```
import matplotlib.pyplot as plt


def main():
    # Initialize a new Plotting window
    plt.figure(figsize=(10, 10))

    # Initializing 3D capabilities
    axes = plt.axes(projection="3d", proj_type='ortho')

    # Setting axis properties
    axes.set_xlim(-10, 10)  # X Axis graduation
    axes.set_ylim(-10, 10)  # Y Axis graduation
    axes.set_zlim(-10, 10)  # Z Axis graduation

    axes.set_xlabel('X')  # X Axis label
    axes.set_ylabel('Y')  # Y Axis label
    axes.set_zlabel('Z')  # Z Axis label

    axes.xaxis.label.set_color('red')    # X Axis color
    axes.yaxis.label.set_color('green')  # Y Axis color
    axes.zaxis.label.set_color('blue')   # Z Axis color

    axes.tick_params(axis='x', colors='red')   # X Axis graduation color
    axes.tick_params(axis='y', colors='green') # Y Axis graduation color
    axes.tick_params(axis='z', colors='blue')  # Z Axis graduation color

    # Display the 3D plotting window
    plt.show()


if __name__ == "__main__":
    main()
```

Exercise 1

Using the code above, check that the MatPlotLib successfully display an empty 3D chart.

Julien SEINTURIER
http://www.seinturier.fr / julien.seinturier@univ-tln.fr

Drawing points and lines

MatPlotLib enable to draw simple primitives like points and lines.

Point

The drawing of a point with (x, y, z) coordinates is made by the code:

```
plt.plot(x, y, z, marker='m', color='c')
```

where:

- `m` is the type of the marker (`'o'` for round, `'+'` for right cross, `'x'` for cross)
- c is the point color (`'red'`, `'greed'`, `'blue'`, …)

Line

The drawing of a line between two points (x_1, y_1, z_1) and (x_2, y_2, z_2) is made by the code:

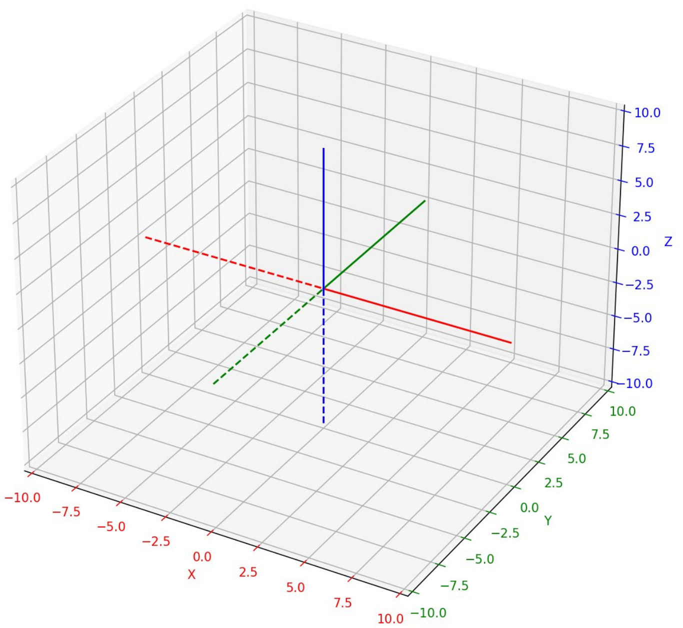
```
plt.plot([x1 , x2],[ y1, y2],[ z1, z2], color='c', linestyle='s')
```

where :

- c is the line color (`'red'`, `'greed'`, `'blue'`, …)
- s is the line style (`'solid', 'dashed', 'dashdot' or 'dotted'`)

Exercise 2

Using point and line drawing functions, draw a 3D referential like the figure below. The referential lines respectively have to be red for X axis, green for Y axis and blue for Z axis. Positive demi axis have to be represented with solid lines, negative demi axis have to be represented with dashed lines.



Julien SEINTURIER
http://www.seinturier.fr / julien.seinturier@univ-tln.fr

Working with points

For the rest of the work, 3D points are represented as Python tuples `(x, y, z)`.

```
p = (x, y, z)
```

create a point p where:

```
p[0] = x
p[1] = y
p[2] = z
```

We are now focusing on 3D point transformation implementation and display.

Translation

Let $P = (x, y, z)$ a 3D point. A translation is an application, denoted $T(\alpha, \beta, \gamma)(P)$ that add values α, β and γ to the coordinates of P. More formally:

$$T(\alpha, \beta, \gamma)(P) = (x + \alpha, y + \beta, z + \gamma)$$

Exercise 3

Write a function `translate_point(point, alpha, beta, gamma)` that takes in parameter a point represented by a tuple `(x, y, z)` and that return a tuple that represents the translated point along vector (α, β, γ).

Test the function translate by displaying the point `(4.0, 3.0, 2.0)` and by displaying the result of the translation along vector `(0.0; 1.0, 1.0)`.

Rotation

Let $P = (x, y, z)$ a 3D point. A rotation is an application, denoted $R_i(\theta)(P)$ that rotate the point P around the axis i by an angle θ. More formally, for a 3D Euclidean space:

The rotation $R_x(\omega)(P)$ around X axis by an angle ω is defined such as:

$$P_r = R_x(\omega)(P) = (x_r, y_r, z_r) \; with \begin{cases} x_r = x \\ y_r = y\cos(\omega) - z\sin(\omega) \\ z_r = y\,sin(\omega) + z\cos(\omega) \end{cases}$$

The rotation $R_y(\varphi)(P)$ around Y axis by an angle φ is defined such as:

$$P_r = R_y(\varphi)(P) = (x_r, y_r, z_r) \; with \begin{cases} x_r = x\cos(\varphi) + z\sin(\varphi) \\ y_r = y \\ z_r = z\cos(\varphi) - x\sin(\varphi) \end{cases}$$

The rotation $R_z(\kappa)(P)$ around Z axis by an angle κ is defined such as:

$$P_r = R_z(\kappa)(P) = (x_r, y_r, z_r) \; with \begin{cases} x_r = x\cos(\kappa) - y\sin(\kappa) \\ y_r = x\sin(\kappa) + ycos(\kappa) \\ z_r = z \end{cases}$$

Exercise 4

Write a function `rot_x_point(point, omega)` that takes in parameter a point represented by a tuple `(x, y, z)` and returns the tuple `(xr, yr, zr)` that represents the rotated point around X axis by an angle `omega`.

Test the function by displaying the point `(4.0, 4.0, 4.0)` as a black circle and displaying the result of its rotation by an angle of $\frac{\pi}{4}$ as a red circle.

Exercise 5

Write a function `rot_y_point(point, phi)` that takes in parameter a point represented by a tuple `(x, y, z)` and returns the tuple `(xr, yr, zr)` that represents the rotated point around Y axis by an angle `phi`.

Test the function by displaying the point `(4.0, 4.0, 4.0)` as a black circle and displaying the result of its rotation by an angle of $\frac{\pi}{4}$ as a green circle.

Exercise 6

Write a function `rot_z_point(point, kappa)` that takes in parameter a point represented by a tuple `(x, y, z)` and returns the tuple `(xr, yr, zr)` that represents the rotated point around Z axis by an angle `kappa`.

Test the function by displaying the point `(4.0, 4.0, 4.0)` as a black circle and displaying the result of its rotation by an angle of $\frac{\pi}{4}$ as a blue circle.

The global rotation of a point within a 3D space is obtained by applying the three rotations around the X, Y and Z axis. Let $P = (x, y, z)$ a 3D point, the rotation of P around the X, Y and Z axis by the angles ω, φ, κ is such that:

$$P_r = R_x(\omega)R_y(\varphi)R_z(\kappa)(P) = (x_r, y_r, z_r)$$

With:

$$x_r = x\cos(\varphi)\cos(\kappa) + y(\mathit{sin}(\omega)\sin(\varphi)\cos(\kappa) - \mathit{cos}(\omega)\sin(\kappa)) \\ + z(\cos(\omega)\sin(\varphi)\cos(\kappa) + \sin(\omega)\sin(\kappa))$$

$$y_r = x\cos(\varphi)\sin(\kappa) + y(\mathit{sin}(\omega)\sin(\varphi)\sin(\kappa) + \cos(\omega)\mathit{cos}(\kappa)) \\ + z(\cos(\omega)\sin(\varphi)\sin(\kappa) - \sin(\omega)\,\mathit{cos}(\kappa))$$

$$z_r = -x\sin(\varphi) + y\,\mathit{sin}(\omega)\cos(\varphi) + z\cos(\omega)\cos(\varphi)$$

Exercise 7

Write a function `rot_point(point, omega, phi, kappa)` that takes in parameter a point represented by a tuple `(x, y, z)` and returns the tuple `(xr, yr, zr)` that represents the rotated point around X, Y and Z axis by the angles `omega`, `phi`, `kappa` respectively.

Test the function by displaying the point `(4.0, 4.0, 4.0)` as a black circle and displaying the result of its rotation by three angles of $\frac{\pi}{4}$ as an orange right cross.

Ensure that when using only one angle value (by setting others to 0), the behavior of `rot_point` is the same as `rot_x_point`, `rot_y_point` and rot_z-point.

Julien SEINTURIER
http://www.seinturier.fr / julien.seinturier@univ-tln.fr

Rotation can be represented using function composition. Rotation of a point P around the axis X, Y and Z can also be represented by:

$$R_z(\kappa) \circ R_y(\varphi) \circ R_x(\omega)(P)$$

Exercise 8

Write a function `rot_point_comp(point, omega, phi, kappa)` that takes in parameter a point represented by a tuple `(x, y, z)` and returns the tuple `(xr, yr, zr)` that represents the rotated point around X, Y and Z axis by the angles `omega`, `phi`, `kappa` respectively.

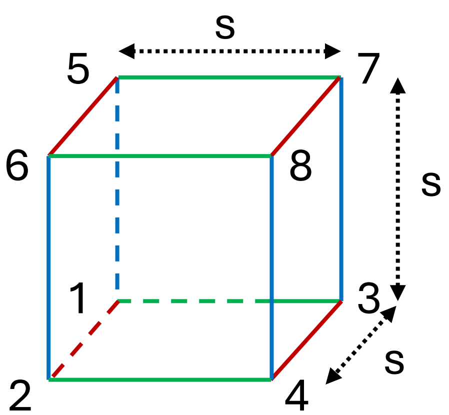
The function `rot_point_comp` has to use functions `rot_x_point`, `rot_y_point` and rot_z-point to perform the rotation.

Test the function by displaying the point `(4.0, 4.0, 4.0)` as a black circle and displaying the result of its rotation by three angles of $\frac{\pi}{4}$ as a violet cross.

Compare the use of both functions `rot_point` and `rot_point_comp` by rotating the same point using the same angles.

Working with shape

For the rest of the work, a cube is represented as Python array of 8 points corresponding to its vertices.



according to the figure below, a cube of size s is defined by the following array:

[(-s, -s, -s), (s, -s, -s), (-s, s, -s), (s, s, -s), (-s, -s, s), (s, -s, s), (-s, s, s), (s, s, s)]

1 2 3 4 5 6 7 8

Exercise 9

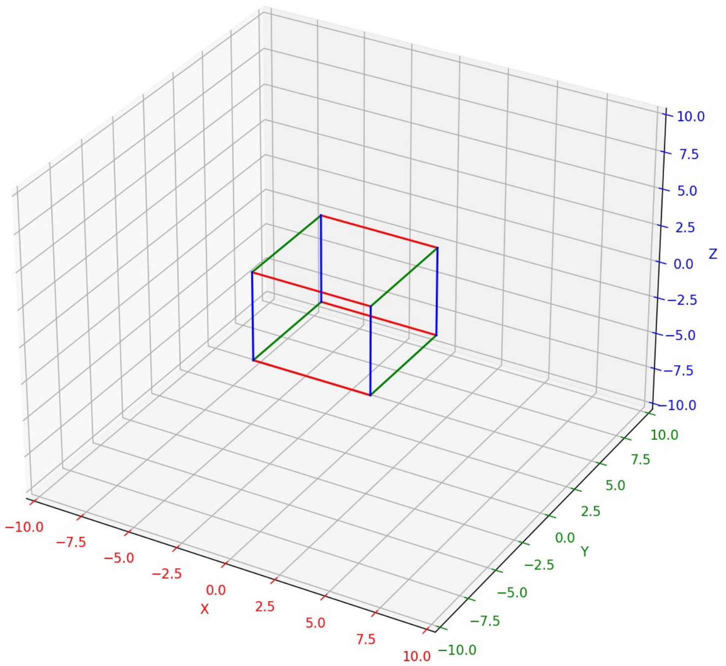
Write a function `cube(size)` that takes in parameter a `size` and create a cube represented by an array of 8 tuples corresponding to its vertices. The vertices order has to respect the figure above.

Julien SEINTURIER
http://www.seinturier.fr / julien.seinturier@univ-tln.fr

Exercise 10

Write a function `display_cube(vertices)` that take in parameter an array of tuples that represent the `vertices` of a cube and that display the cube within the 3D environment.

Each edge of the cube has to be colorized with the same color as its parallel axis (see image below).



Translation

Translating a shape can be done by translating all its vertices.

Exercise 11

Write a function `translate_cube(vertices, alpha, beta, gamma)` that takes in parameter an array of tuples that represent the `vertices` of a cube and that return an array of tuples that represent the vertices of the translated cube along vector (α, β, γ).

Test the function `translate_cube` by displaying the result of the translation of a cube of size 3.0 along vector `(1.0; 2.0, 3.0)`.

Rotation

Rotating a shape can be done by rotating all its vertices.

Exercise 12

Write a function `rotate_cube(vertices, omega, phi, kappa)` that takes in parameter an array of tuples that represent the `vertices` of a cube and three rotation angles `omega, phi, kappa` and that rotate the cube according to the given angles.

Test the function `rotate _cube` by displaying the result of the rotation of a cube of size 3.0 for the angles $\omega = \frac{\pi}{4}$, $\varphi = \frac{\pi}{3}$ and $\kappa = \frac{\pi}{2}$.

Merging transformations

Translation and rotation can be combined in order to locate shapes.

Exercice 13

Using previous functions, create a cube with a size of 2.0 and display simultaneously

- The cube transformed by a translation $T(\alpha, \beta, \gamma)$ where $\alpha = 0.25$, $\beta = 0.50$ and $\gamma = 0.75$ then a rotation $R_z(\kappa) \circ R_y(\varphi) \circ R_x(\omega)$ where $\omega = \frac{\pi}{6}$, $\varphi = \frac{\pi}{4}$ and $\kappa = \frac{\pi}{3}$
- The cube transformed by a rotation $R_z(\kappa) \circ R_y(\varphi) \circ R_x(\omega)$ where $\omega = \frac{\pi}{6}$, $\varphi = \frac{\pi}{4}$ and $\kappa = \frac{\pi}{3}$ then a translation $T(\alpha, \beta, \gamma)$ where $\alpha = 0.25$, $\beta = 0.50$ and $\gamma = 0.75$

Do the two transformed cube share the same location? Why?