NYU ROB-UY 2004
Robotics Locomotion and Manipulation
Spring Semester 2026

Due on BrightSpace, Midnight, Monday, February 9th, 2026.

Assignment 1: Coordinate Frames and Rotations

This assignment is to be completed by yourself, without help from the internet, LLMs including ChatGPT, etc. Your class notes should be sufficient.

Question 1 - 10 marks
The answers to this question are to be done with pencil and pen on paper (or stylus on an iPad screen). It should be submitted as a jpeg or pdf.

a) Draw a 2D coordinate frame, including the point $p = [2\ 4]^T$. Be sure to label your axis.
b) On the same diagram as in a), draw a point q that is the point p, after it has been rotated by $-\pi/2$.
c) Draw a 3D coordinate frame, including the point $p = [2\ 4\ 2]^T$.
d) Write out the rotation matrix that could take a point like p from question 1c), and rotate it first by $\pi/2$ about the z-axis, than by $-\pi/4$ about the y-axis.
e) Given the rotation matrices R_{z1}, R_{z2} and R_{x1}, where the first two matrices rotate a 3D point about the z-axis, and the third matrix rotates a point about the x-axis, which are the following are necessarily true?
    a) $R_{z1}R_{z2} = R_{z2}R_{z1}$
    b) $R_xR_{z2} = R_{z2}R_x$

Question 2 - 10 marks
The answers to this question are to be coded in python. Note that all angles are in radians.

a) In a file called `my_asignment_1.py`, Write a python function called `rotate2D` that takes in two input arguments: a scalar angle `theta`, and a 2x1 `numpy` array called `p_point`. The function should return a 2x1 `numpy` array called `q_point`. It is assumed that `q_point`'s elements contain the x, y coordinates of a point with respect to a 2D coordinate frame. The array `q_point` should be calculated as the rotation of `p_point` by angle `theta.` Assume standard right hand rule conventions. Do not use any "rotation" libraries. Create a rotation matrix and use the `matmul` function. Make sure this function passes all unit tests provided in the file `assignment_1_unit_test_2a.py`. You can place your file `my_asignment_1.py` within the same folder as the unit test file, and type python3 `assignment_1_unit_test_2a.py` to make sure your function passes the available tests.

b) In the same file called `my_asignment_1.py,` Write a python function called `rotate3D` that takes in three input arguments: a scalar angle `theta`, a string `axis_of_rotation` (i.e. that should be one of three values `'x'`, `'y'`, or `'z'`), and a 3x1 `numpy` array called `p_point`. The function should return a 3x1 `numpy` array called `q_point.` It is assumed that `q_point`'s elements contain the x, y, z coordinates of a point with respect to a 3D coordinate frame. The array `q_point` should be calculated as the rotation of `p_point` rotated about the `axis_of_rotation` by the amount `theta`. Assume standard right hand rule conventions. Do not use any "rotation" libraries. Create a rotation matrix and use the `matmul` function. Make sure this function passes all unit tests provided in the file `assignment_1_unit_test_2b.py`.

c) In the same file called `my_asignment_1.py`, write a python function called `rotate3D_many_times` that takes in two input arguments: a list of tuples of length greater than 0 called `rotation_list`, and a 3x1 numpy array called `p_point`. Each tuple in `rotation_list` has `[theta, axis_of_rotation]`, and the `axis_of_rotation` should be one of three values `'x'`, `'y'`, or `'z'`. An example input would be [[`math.pi/4, 'z'`], [`-math.pi/8, 'y'`],[`math.pi/4, 'z'`]], [1.1,-2.0, 4.66]. The function should return a 3x1 `numpy` array called `q_point`. It is assumed that `q_point`'s elements contain the x, y, z coordinates of a point with respect to a 3D coordinate frame. The array `q_point` should be calculated as the rotation of `p_point` rotated once for each tuple in the input list. It should be rotated in the order of the tuples, rotated about an amount according to the first argument of the tuple, and about the axis as specified by the second argument of the tuple. Your function `robot3D_many_times` should call your function `rotate3D` for each rotation in the list of tuples. Make sure this function passes all unit tests provided in the file `assignment_1_unit_test_2c.py`.

Question 3 - 10 marks

Without changing any code from question 2, you will be graded upon how well your functions from question 2 pass other, undisclosed test functions.