1 Lab Scope and Aim

The aim of this lab is to introduce the students the basic notations and functionalities in time and motion for robots, especially while using MATLAB and the corresponding Robotics Toolbox. The lab will conclude with assignments that summarize the discussed topics, where the students will have to solve various tasks connected to real life problems.

Prerequisites: The students should have a good background in the following areas:

- *Chapter 2 - Robotics Vision and Control*
- *Chapter 3 - Robotics Vision and Control*

Lab Supplementary material: This lab is accompanied with the supplementary material to support the assignments

1. Download lab1.zip file from CANVAS and unzip it. The folder names correspond to the number of each Task in Section 2.2.

1.1 Robotics Toolbox

The lab implementation is structured around the Robotics Toolbox which is accompanied with the book Robotics, Vision and Control: Fundamental algorithms in MATLAB. The toolbox contains functions and classes to represent 2D or 3D position and orientation, while includes conversions between these data types to various representations. More information about the toolbox specifics and information on how to install, can be found in the following link:

`http://petercorke.com/wordpress/toolboxes/robotics-toolbox`

2 Assignments

2.1 Exercises from Chapter 3 of the Book

Exercise 7 For a *tpoly* trajectory from 0 to 1 in 50 steps explore the effects of different initial and final velocities, both positive and negative. Under what circumstances does the quintic polynomial overshoot and why?

Exercise 8 For a *lspb* trajectory from 0 to 1 in 50 steps explore the effects of specifying the velocity for the constant velocity segment. What are the minimum and maximum bounds possible?

Exercise 9 For a trajectory from 0 to 1 and given a maximum possible velocity of 0.025 compare how many time steps are required for each of the *tpoly* and *lspb* trajectories?

Exercise 13 For the *mstraj* example a) Repeat with different initial and final velocity. b) Investigate the effect of increasing the acceleration time. Plot total time as a function of acceleration time

2.2 Lab tasks

Task 1 (Object Cartesian Motion) In this task you should perform and visualize the Cartesian motion of a 3D object (shown as a 3-axis frame in Figure 1) from the starting pose **A** to the final pose **D** while visiting two intermediate poses **B** and **C** respectively.

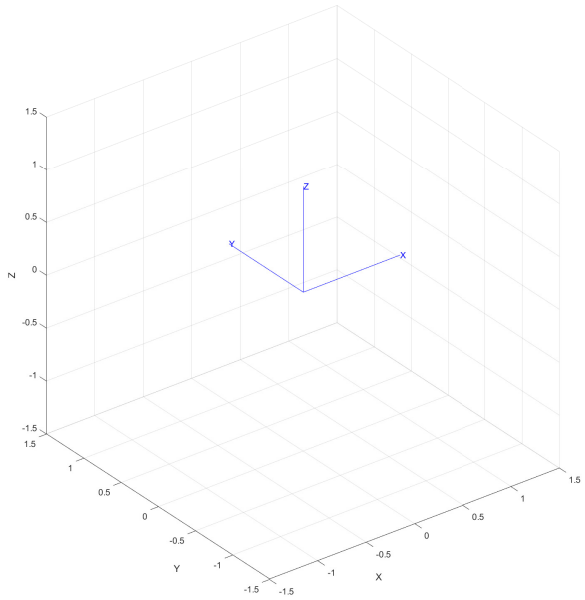

Figure 1: Initial pose A of the 3D object

Given

- Starting Pose **A** at (x, y, z, ϕ, θ, ψ)=(0, 0, 0, 0, 0, 0)
- Final Pose **D** at (x, y, z, ϕ, θ, ψ)=(1, 0, -0.5, 0, $\frac{\pi}{2}$, 0)
- Intermediate Pose **B** at (x, y, z, ϕ, θ, ψ)=(0, 0, 1, 0, $\frac{\pi}{2}$, 0)
- Intermediate Pose **C** at (x, y, z, ϕ, θ, ψ)=(1, 0, 1, 0, -π, 0)

ToDo

- Design the trajectory that smoothly moves the object from pose A to pose D.

Task 2 (Ground Robot) In this task you will be finding trajectory for a mobile robot.

Given

- Initial position at [50,30].
- Goal position at [20,5].

- The map is depicted in Figure 2 and it will be plotted in "Task2/run.m" file.
- The path should be added to "Task2/run.m" file.

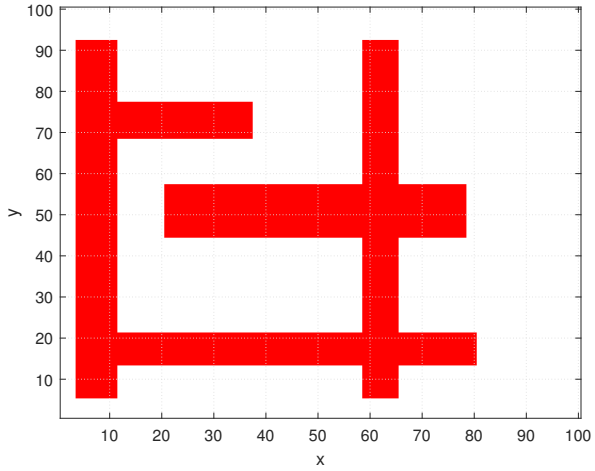

Figure 2: Map of the area.

ToDo

- Design the multi-segment trajectory that moves the robot from the initial point to the goal point with maximum velocity of [1,1]. You can access the map by running "run.m" in Task2 folder and the value of obstacle" should be 0.
- Increase the maximum bound on velocity and observe the results.
- Assume that the door will block the path for 5 sec, generate new path which pass the door with correct timing. The door location is shown in Figure 3. You can access the map by running "Task2/run.m" and changing the value of obstacle" to 1.

Task 3 (Manipulator) In this task you will be using the PUMA560 robot to draw letters on a wall.

Given

- The wall can be standing either vertically or inclined and is defined as $ax + by + cz + d = 0$.
- In the vertical case the wall is characterized by equation $x = 1$ (y-z plane)
- In the inclined case the wall is characterized by equation $cos(10^o)x + cos(10^o)z = 1 * cos(10^o)$ (xy plane pitch 80^o)
- The end-effector is equipped with a pen to perform the writing task. The tip of the pen has a body frame coordinate (0, 0, 0.4).
- If the pen tip does not interact with the wall then nothing will be drawn
- MATLAB files associated with the task are provided in folder Task3.

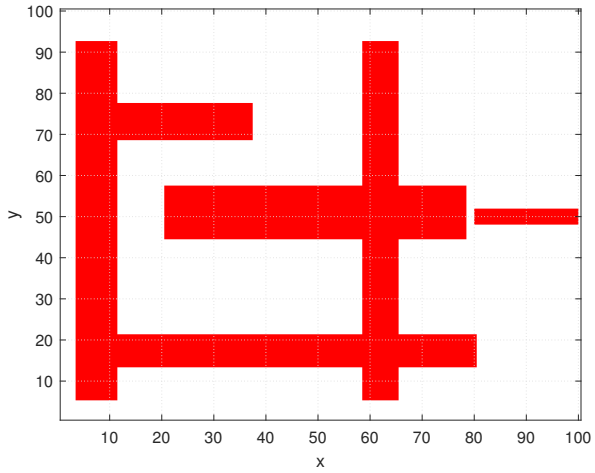

Figure 3: Map of the area with door.

ToDo

- Design the trajectory that moves the pen along the wall and draws the first letter of your last name on the vertical wall. You will use the code inside the Task 3 folder and only need to execute the file "run_vertical.m". Example trajectory for the manipulatpr is shown in Figure 4.
- Design the trajectory that moves the pen along the wall and draws the first letter of your last name and the first letter of your first name with different speeds on the vertical wall. You will use the code inside the Task 3 folder and only need to execute the file "run_vertical.m".
- Design the trajectory that moves the pen along the wall and draws the first letter of your last name for the wall parallel on the inclined wall. You will use the code inside the Task 3 folder and only need to execute the file "run_inclined.m". Example trajectory for the manipulator is shown in Figure 5.

Task 4 (MAV) In this task, you will find the path for the UAV to navigate in complex environments and inspect different structures.

Given

- The MAV can move in any direction.
- It is assumed that the heading of the MAV is looking to the object.
- The camera covers an area of 1.5×1.5 m^2 when MAV is 1 m away from the structure (applies to Figures 8 and 9).
- The MAV should have 1 m safety distance to the object (applies to Figures 8 and 9).

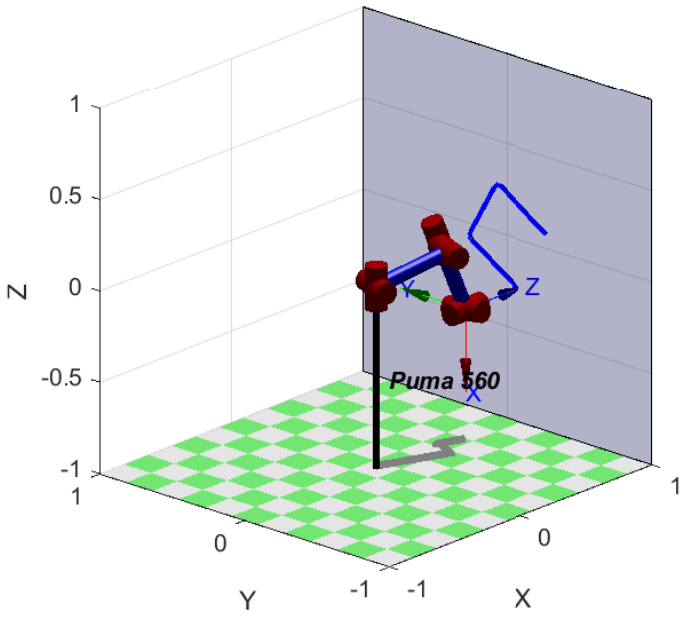

Figure 4: Example trajectory for the manipulator on the vertical wall

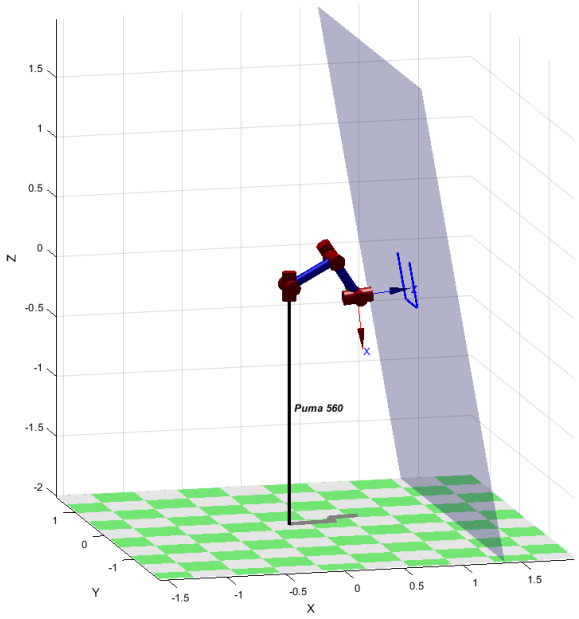

Figure 5: Example trajectory for the manipulator on the inclined wall

ToDo

- Design a trajectory for the MAV to start from position (-5,2,0) and arrive to position (22,2,0), while avoiding obstacles in Figure 6. You can access the map by running "run.m" in Task4 folder and adding map1.txt" to loadmap line.

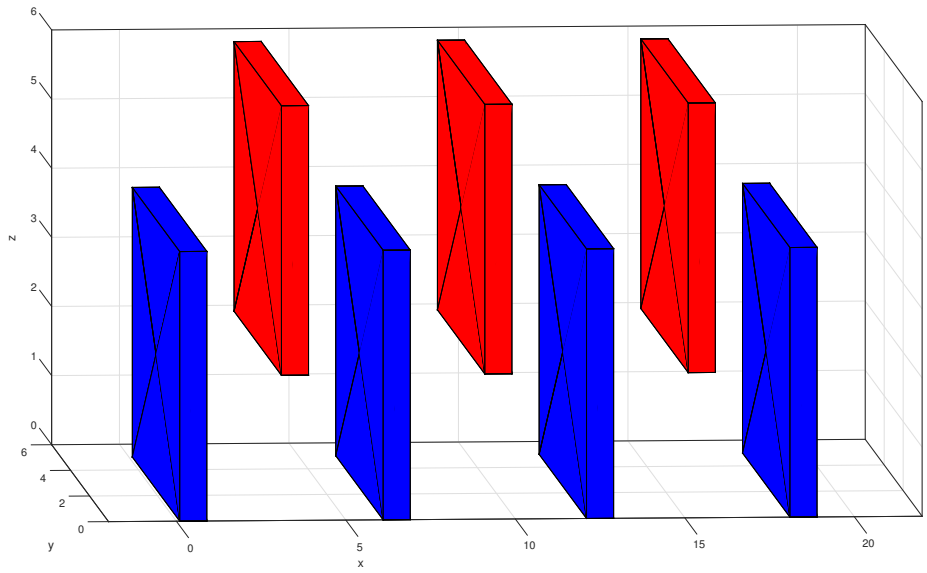

Figure 6: Complex structure that the MAV should navigate while avoiding collision.

- Design a trajectory for the MAV to start from position (5,1,0) and arrive to position (5,20,2), while avoiding obstacles in Figure 7. You can access the map by running "run.m" in Task4 folder and adding map2.txt" to loadmap line.

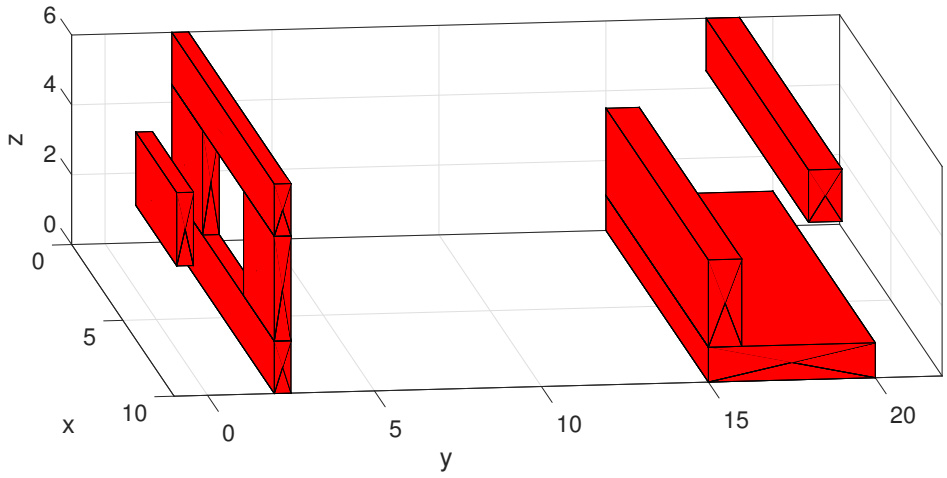

Figure 7: Complex structure that the MAV should navigate while avoiding collision.

- The MAV can start with arbitrary initial condition. The path should be generated in a way that the camera footprint inspect the structure depicted in Figure 8. You must run the "fountain.m" file to access the LTU fountain 3D model.

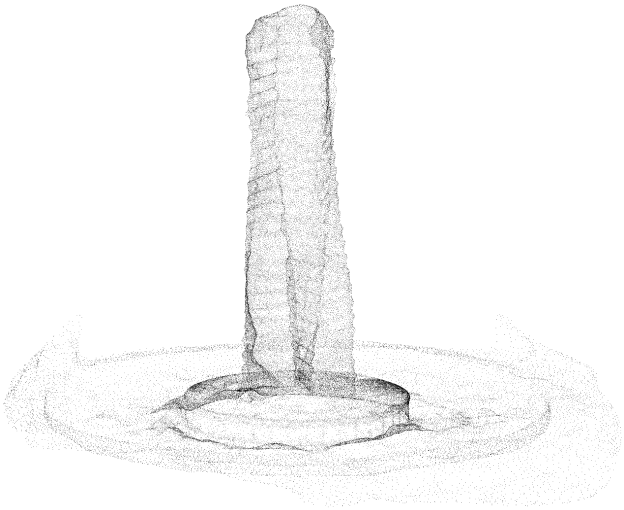

Figure 8: 3D model of the LTU fountain that the MAV should navigate around

- The MAV can start with arbitrary initial condition. The path should be generated in a way that the camera footprint inspect the structure depicted in Figure 9. You must run "windturbine.m" file to access to the wind turbine structure.

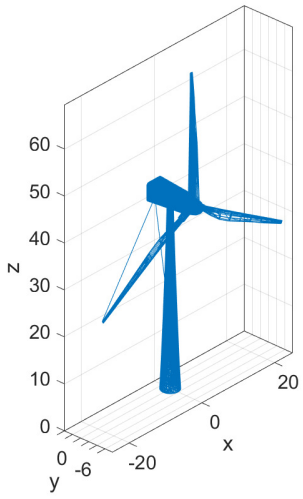

Figure 9: Inspection of wind turbine.

- (Optional) In case of multiple MAVs, generate paths for each MAV for Figures 8 and 9. The paths should be collision free and results to shorter flight times compare to single MAV.