Practical Class 1b - ENGM300

The aim of this week's practical exercise is to provide some familiarity with some of the capabilities of the robotics toolbox in Matlab.

Today's class is designed to:

- Let you explore some of the kinematics concepts that we have been learning in the first weeks of the course;
- See practical implementations of both forward and inverse kinematics;
- And identify the range of data, including parameters that are linked to manipulator kinematics.

The robot used in the case study is kinematically similar to the PUMA 762. We saw an engineering drawing/schematic of this robot earlier in the lecture classes:

- https://commons.wikimedia.org/wiki/File:ArmDim700side.png

It is not essential that you work collaboratively on these tasks but it may help you to discuss them with your colleagues.

The function to generate the workspace is available from the Mathworks at:

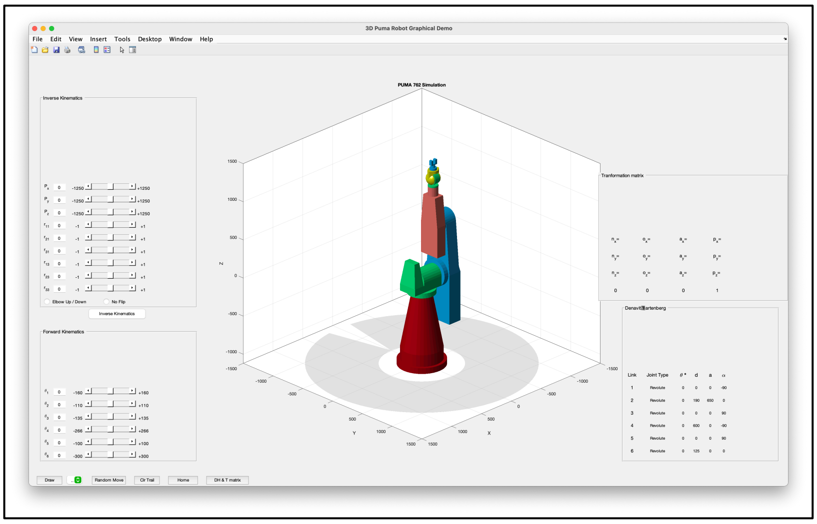
- https://uk.mathworks.com/matlabcentral/fileexchange/59663-puma-robot-simulation

The function downloadable above tends to generate some errors. A more complete set of files is available through the change log on Matlab's file exchange. These are available to you in the Practical Class 1b folder on SurreyLearn. I suggest that you save these to a local folder on your computer and run the function from there by selecting the folder as your Matlab path for this session.

An additional line of code has been added before line 23 in Puma_Simulation.m:

- `fig_1=figure(1);`

Hopefully, when you run the function in the Matlab workspace, you will see the following interface, which is what you will need to answer the questions that follow:



Questions to Explore

- Do the dimensions and link parameters match those in the Wikipedia schematic?

- Identify the ‘home’ position. Hopefully, when you launch the function you should see this. It is also possible to select the ‘homing’ button in the bottom-left of the workspace.

- In much the same way that you did last week with the Trossen robot, identify each of the joints on the PUMA 762 and manipulate them one at a time. The particular version of the Matlab function/interface should allow you to see all kinds of output. Take note of how each bit of output changes depending on a particular input.

- Which bits of the Denavit-Hartenberg table are changing and which bits are staying constant when the robot is instructed to do something? Is this a surprise or expected?

- Choose a relatively simple wrist pose and see whether you are able to predict the joint variables.

- When picking the random configuration option, are you able to reverse engineer the joint variables and transformation matrix by inspection? Can you get a reasonable approximation to them?

- Based on the lecture notes and using the tools in the interface, can you create the eight different possible configurations for the PUMA to achieve a particular pose.

- In the home position, what do the form and specific entries in the transformation matrix show?

- By carefully controlling the joints, can you create specific entries in the overall kinematic transform. For example, can you find configurations that create ones and zeros in the rotation part? Can you create specific translations in the final column?

- Without trying to do it, how would you empirically establish the workspace of this robot? Would your method work best for the reachable or dextrous workspace? Is one more of a challenge than the other?

- What robot parameters are not provided to you in the interface? When may these be useful? Think about not just the final configuration but also the journey between points.

- We have not covered it in the course, but how do you think Matlab’s toolbox is determining the path between points? Are you able to identify this from creating multiple paths?

- As an observational exercise, start the robot in the home position and select a random final manipulator position. Note the joint variables down at the end of the movement. Now, from starting in the home position again, in turn, change joint variables θ_{1-6} one at a time. Is the path followed to the end configuration the same? When Matlab moves the robot, do the joints all begin and end moving at the same time?

- As a last task, have a quick look at the code. How much is related to robot kinematics?!