·LESSON 11·

Robotic Systems Toolbox Using MATLAB

Peter Corke's Robotics toolbox was a great introduction to robotic arm simulations in MATLAB, but it is somewhat limited. Mathworks (MATLAB's parent company) offers the Robotic Systems Toolbox (RST), which can be applied to any robot; we will use it to advance the capabilities of our robotic arm simulations.

This lesson will mostly refer you to the toolbox's excellent documentation on manipulator algorithm design. Go through them and test the code yourself on MATLAB.

Installing The Toolbox

1. Open MATLAB. In the 'Home' Panel, click the arrow next to 'Get Add-Ons', then click 'Manage Add-Ons'.
2. The Add-On Manager appears. If the Robotics System Toolbox is present, that means it is already installed. If not, go ahead to the next step.
3. Go to Add-Ons > Get Add-Ons.
4. In the Add-On explorer, search for the Robotics System Toolbox.
5. Click on the correct search result and install it.

Robot Models

We first need to understand how RST stores models. This is explained by the following link:
https://in.mathworks.com/help/robotics/ug/rigid-body-tree-robot-model.html

The next link shows how to construct your own rigid body tree in MATLAB using RST.
https://in.mathworks.com/help/robotics/ug/build-a-robot-step-by-step.html

A note on coordinate systems: Read this page to understand the different coordinate systems and transformations you will likely encounter in RST's documentation. Here is a video explaining homogeneous transformation matrices (which is ubiquitous in robotic manipulator design).

Inverse Kinematics

Now we shall delve into inverse kinematics. The next link gives a short description of the algorithms used, you may check it out.
https://in.mathworks.com/help/robotics/ug/inverse-kinematics-algorithms.html
Next, try to implement a simple 2-D robotic arm and make it sketch a circle. Follow this link as a guide.
https://in.mathworks.com/help/robotics/ug/2d-inverse-kinematics-example.html

Optional:

You can try a more complex 2D manipulator by following this link:
https://in.mathworks.com/help/robotics/ug/solve-inverse-kinematics-for-a-four-bar-linkage.html

The next link shows you how to make a proper 3 dimensional robot follow a trajectory:
https://in.mathworks.com/help/robotics/examples/plan-a-reaching-trajectory-with-kinematic-constraints.html