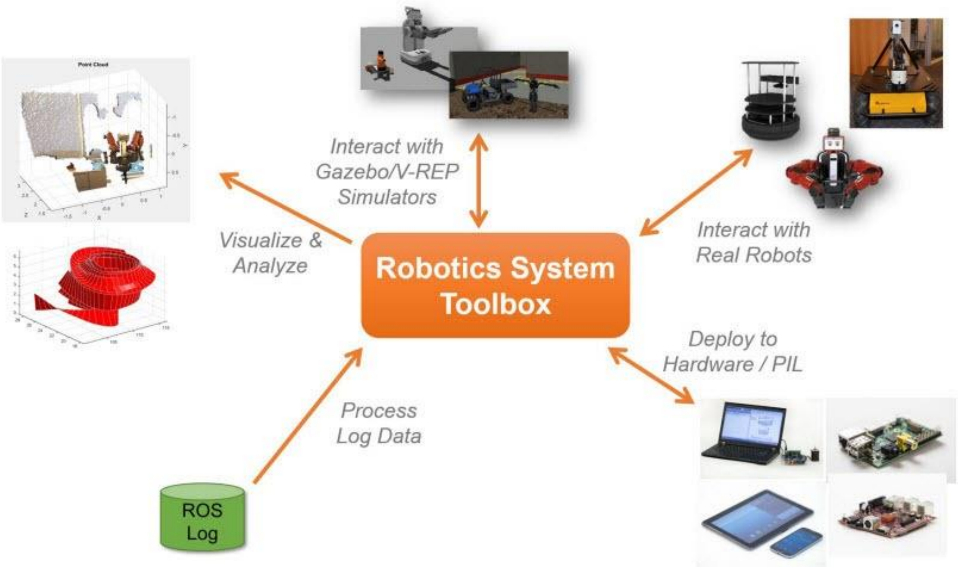
WEEK 3

We'll be starting with Week 3 of this QSTP. This week deals with the Robotics System Toolbox and Stateflow.

ROBOTICS SYSTEM TOOLBOX

Robotics System Toolbox provides tools and algorithms for designing, simulating, and testing manipulators, mobile robots, and humanoid robots. For manipulators and humanoid robots, the toolbox includes algorithms for collision checking, trajectory generation, forward and inverse kinematics, and dynamics using a rigid body tree representation. For mobile robots, it includes algorithms for mapping, localization, path planning, path following, and motion control. The toolbox provides reference examples of common industrial robot applications. It also includes a library of commercially available industrial robot models that you can import, visualize, and simulate.



GETTING STARTED WITH RST

[Rigid Body Tree Robot Model]
Explore the structure and specific components of a rigid body tree robot model.

Robot Dynamics
Robot dynamics is the relationship between the forces acting on a robot and the resulting motion of the robot

Build a Robot Step by Step
This example goes through the process of building a robot step by step, showing you the different robot components and how functions are called to build it.

2-D Path Tracing with Inverse Kinematics
Calculate inverse kinematics for a simple 2-D manipulator.

You can also go through the below videos to understand basic functionality and tools provided by robotic system toolbox and the trajectory planning and manipulation algorithms that can be implemented with it.

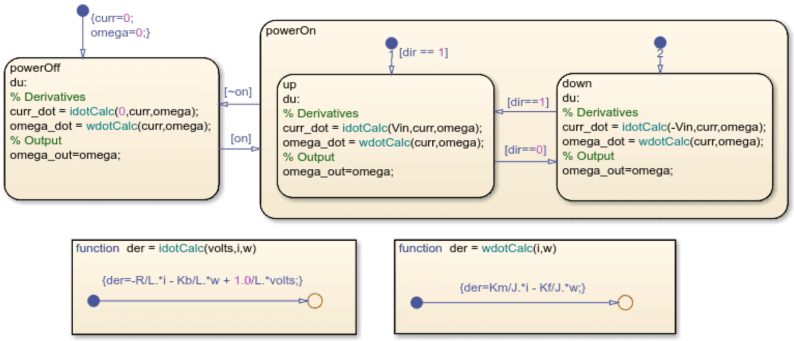
1. Designing Robot manipulator algorithms
2. Manipulation
3. Developing Robotics Applications with MATLAB, Simulink, and Robotics System Toolbox
4. Trajectory Planning for Robots

It's highly suggested to refer to Robotic System Toolbox Documentation in case you face any doubts.

STATEFLOW

Stateflow provides a graphical language that includes state transition diagrams, flow charts, state transition tables, and truth tables. You can use Stateflow to describe how MATLAB algorithms and Simulink models react to input signals, events, and time-based conditions.
Stateflow enables you to design and develop supervisory control, task scheduling, fault management, communication protocols, user interfaces, and hybrid systems.

You have to complete Stateflow Onramp this week. Apart from this it's also highly recommended to go through the documentation of Stateflow.



```
powerOff
du:
% Derivatives
curr_dot = idotCalc(0,curr,omega);
omega_dot = wdotCalc(curr,omega);
% Output
omega_out=omega;
```

```
up
du:
% Derivatives
curr_dot = idotCalc(Vin,curr,omega);
omega_dot = wdotCalc(curr,omega);
% Output
omega_out=omega;
```

```
down
du:
% Derivatives
curr_dot = idotCalc(-Vin,curr,omega);
omega_dot = wdotCalc(curr,omega);
% Output
omega_out=omega;
```

```
function  der = idotCalc(volts,i,w)
{der=-R/L.*i - Kb/L.*w + 1.0/L.*volts;}
```

```
function  der = wdotCalc(i,w)
{der=Km/J.*i - Kf/J.*w;}
```

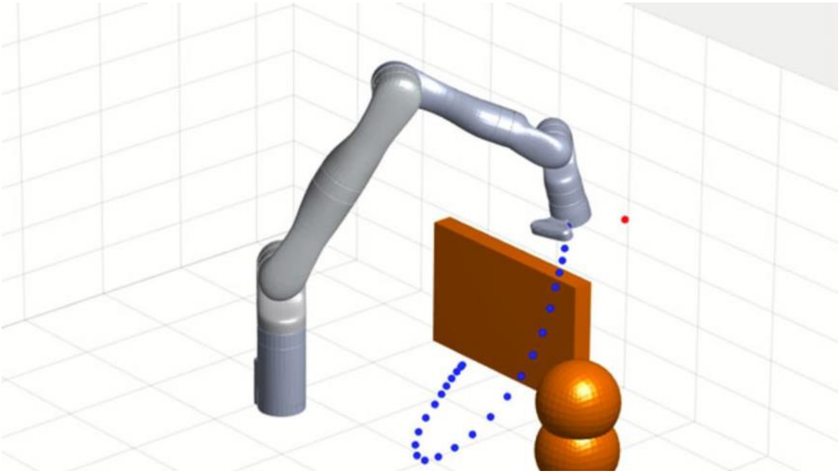
Model of a DC motor in Stateflow

ASSIGNMENT 3

1. Submit the Certificate for Stateflow Onramp in the similar manner you did for Assignment 1
2. Make your own Robotic arm (2d manipulator) in the Robotic system toolbox using a rigid body tree model. The arm should have 3 joints (can be of any type) and the length of each link should be **0.6cm, 0.4cm and 0.2cm** respectively. Using The RST functions, find the **Inverse Kinematics matrix** and the **DH parameters** of the arm.

To have a better understanding of DH parameters and Inverse Kinematics you can refer to these videos:

1. DH parameters
2. Inverse Kinematics



Submit a .zip file containing the certificate and the script.