# Summative Assessment 2

## Advanced Robotics

**General Instructions:** The zip archive contains the skeleton solutions of each question, and the script *sim_robot.m* that can be used for visualization purposes as shown in the scripts *FK_test.m* and *IK_test.m*. Before running the simulation, you must download and setup Peter Corke's Robotics Toolbox: `http://petercorke.com/wordpress/toolboxes/robotics-toolbox/`
Note that the Toolbox must not be used to implement the functions requested in the assignment.

**Question 1**: Write a MATLAB function *IK.m* to solve for the iterative Inverse Kinematics of a given robot. The inputs of this function should be:

- DH parameters: n × 4 matrix which consists of DH parameters.
- joint type: n-dimensional vector which describes joint types (0 for revolute and 1 for prismatic)
- q: n-dimensional vector of joint variables specifying the robot's configuration.
- desired position: desired position of the robot's end-effector. This is $2\times1$ vector for a planar robot and $3\times1$ vector for a spatial robot.
- desired orientation: desired orientation of the robot's end-effector. This is a scalar for a planar robot and a 3x1 vector for a spatial robot. Use roll-pitch-yaw angles for 3D robots.

IK can be implemented using the Jacobian Transpose to compute the inverse of the Jacobian matrix. The output of the IK function should be:

- a matrix including the robot's joint variables for each iteration.

Note that, this function solve the IK problem iteratively. So, you need to define appropriate terminating conditions for both maximum iterations and accuracy of the final result. When updating joint variables, consider multiplying by a small gain in order to let the algorithm converge. Use the script *IK_test.m* to define your robot, solve the IK problem, and visualize how the robot achieves the desired end-effector pose.