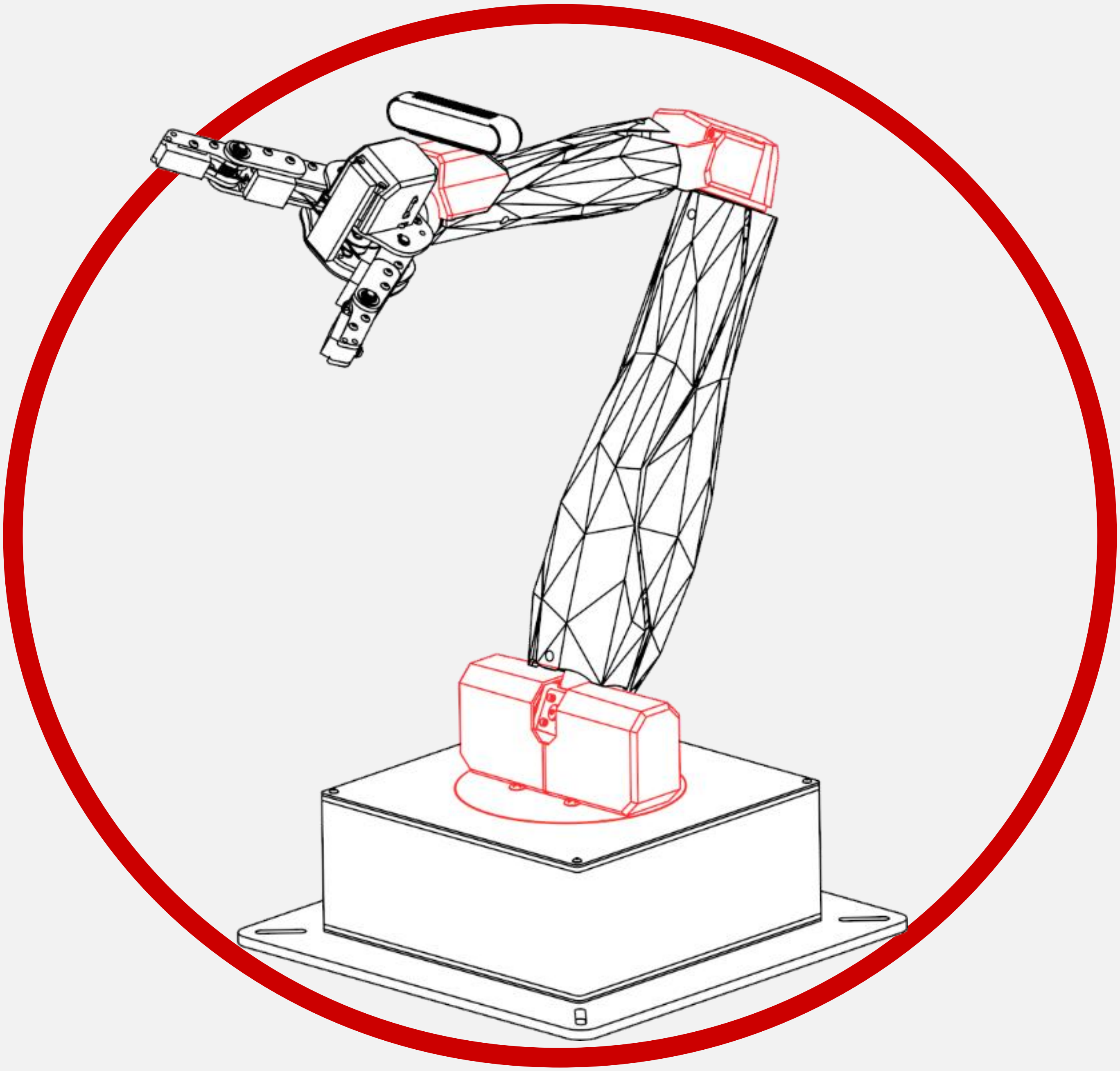

QArm

Trajectory Generation

V1.0 – 13th February 2025

Quanser Consulting Inc.
119 Spy Court
Markham, Ontario
L3R 5H6, Canada

info@quanser.com
Phone : 19059403575
Fax : 19059403576
printed in Markham, Ontario.

Caution

This equipment is designed to be used for educational and research purposes and is not intended for use by the public. The user is responsible for ensuring that the equipment will be used by technically qualified personnel only. Users are responsible for certifying any modifications or additions they make to the default configuration.

QArm – Application Guide
Trajectory Generation

Why explore Trajectory Generation?

A teach pendant is a hardware interface used to first “teach” a set of discrete points to a robotic manipulator, then initiate motion playback where the robot traverses each of the taught points. Teach pendants are commonly used to program robots to perform pick, manipulate and place tasks in an assembly line. Since the desired points that are taught to the robot are Cartesian coordinates of the end-effector, we require a formulation that determines the required joint angles that result in the desired end-effector position. This formulation is referred to as the manipulator’s inverse kinematic model. The purpose of this lab is two-fold. First, you will determine the robot’s inverse kinematics model, then use the model to implement a software-based teach pendant that simulates a robotic assembly process.

Background

The QArm content contains 5 labs that focus on kinematic manipulation. The first one focuses on learning how to do low level control, workspace identification, lead through control, teach pendant and trajectory generation. This lab focuses on Lead Though is performed for a robotic manipulator.

Prior to starting this lab, please review the following concept reviews (should be located in Documents/Quanser/4_concept_reviews/),

- Concept Review – Trajectory Generation

Getting started

The goal of this lab is to study how smooth trajectories are designed for a robotic manipulator to traverse from points A to B.

Before you begin this lab, ensure that the following criteria are met.

- The QArm has been setup and tested. See the QArm Quick Start Guide for details on this step.
- You are familiar with the basics of Simulink. See the [Simulink Onramp]() for more help with getting started with Simulink.

Waypoints

Alongside trajectory generation we need to look at how to handle transitions between waypoints. At the start of the path, the current position/speed corresponds to the initial position/speed for the trajectory generator. The desired setpoint and speed correspond to the final position/speed. When you arrive at the final setpoint, these points become the initial position/speed for the next task, and the final position/speed setpoints change to the next waypoint. A waypoint generator, that correctly handles this transition while checking if the manipulator has arrived at any desired set point, is also required. One simple way to account for this is to use a Finite State Machine (FSM). These are simple versions of a Turing Machine, where the system can jump between a series of finite unique states. Based on the measurements available to the FSM at any given rate, the FSM decides what action to take, and what state to jump to next. A pseudo-code for a simple FSM waypoint navigator would look like,

<u>STATE 1:</u>
IF the user says HOLD, then DESIRED_SETPOINT is the CURRENT location and NEXT_STATE is STATE 1.
ELSE DESIRED_SETPOINT is the NEXT WAYPOINT and NEXT_STATE is STATE 2

<u>STATE 2:</u>
IF you are close to the DESIRED_SETPOINT, NEXT_STATE is STATE 1.
ELSE keep moving towards DESIRED_SETPOINT, and NEXT_STATE is STATE 2.

This can also be represented by a simple state diagram shown in Figure 1.

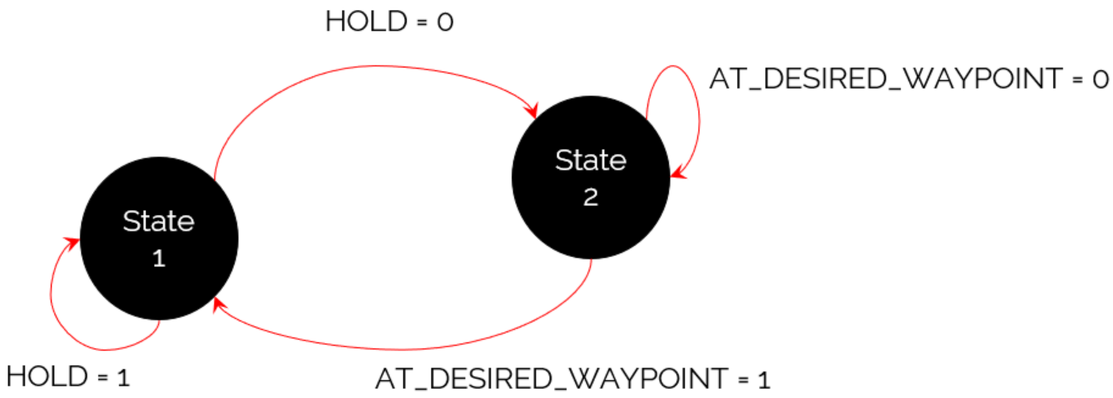

Figure 1: Simple State Machine logic for waypoint navigation

This way, you will be able to provide the set of waypoints for any task including assembly, welding or even a pick/place task, and the waypoint navigator will cycle through them. After the last waypoint, the navigator will simply pick the first waypoint again.