EXPERIMENT 5: AUTOMATION CHALLENGE

The purpose of this challenge is to implement an example of manipulator control in a real-world factory automation context. The scenario that is presented involves many of the topics covered in the 4DOF MICO curriculum including kinematics and trajectory generation. In addition, the concept of state machines is introduced to bring the components of the overall system together.

Topics Covered

- Automation tasks
- State machines

Prerequisites

- The robot has been setup and tested. See the Quick Start Guide for details.
- You have access to the User Manual.
- You are familiar with the basics of Matlab® and Simulink®.
- You should have an understanding of the previous topics covered including kinematics and trajectory generation

1 Challenge

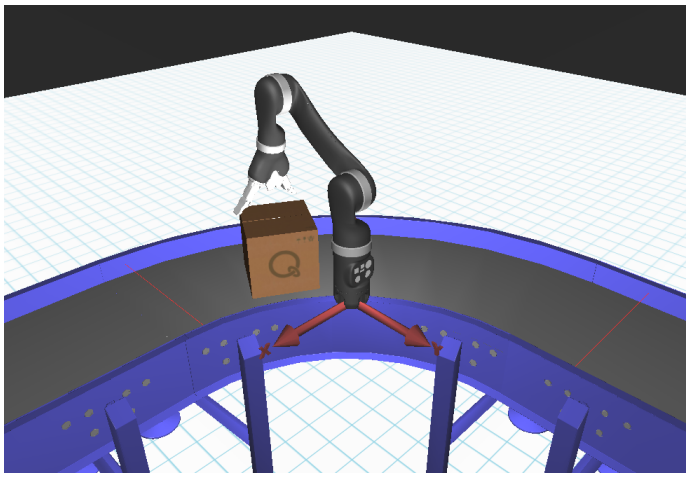

Figure 1.1: Automation Challenge.

The scenario that is presented in this challenge is that of a typical automation task. The 4DOF MICO has been placed in a factory, alongside a conveyor system. The workspace for the manipulator is defined as the reachable area between the two break beam photosensors on either side of the curved portion of the belt. Your task is to define a trajectory for the manipulator to follow in order to "inspect" the box when it is inside the workspace. The sequence of commands that bring the manipulator to the initial position of the box when it enters the workspace, then follow the box until it leaves the workspace, and finally return the manipulator to the home position, should be defined using a state machine. The easiest method for creating state machines in Matlab® is to use the Stateflow toolbox. The toolbox allows you to define states and transitions based on conditional statements and workspace variables. Each state can then call command arguments, or even custom functions. For more information, refer to the Stateflow documentation.

In the case of the automation challenge, the states that are defined will be determined using the break beam sensors. When the box breaks the first beam and enters the workspace, the `inspect` variable will become *true*. The box then moves along the curve shown in Figure 1.2, with velocity defined by the `vel` variable. As the box leaves the workspace, the `inspect` variable becomes *false* and the manipulator should return to the home position defined by the `home` variable.

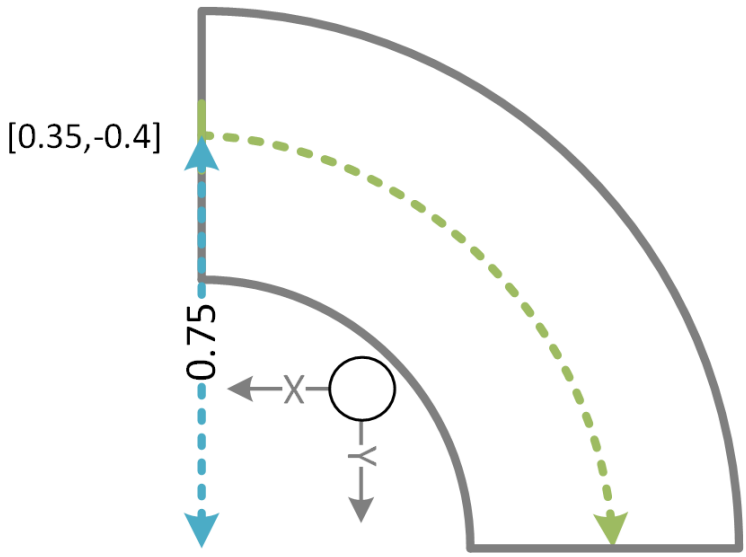

Figure 1.2: Workspace for the automation challenge.

Quanser Inc.
119 Spy Court
Markham, Ontario
L3R 5H6
Canada
info@quanser.com
Phone: 1-905-940-3575
Fax: 1-905-940-3576

Printed in Markham, Ontario.

For more information on the solutions Quanser Inc. offers, please visit the web site at:
http://www.quanser.com