Department of Electrical and Computer Engineering

University of Victoria

ELEC 360 - Control Systems I

Laboratory Report

Experiment No.:	4
Title:	Introduction to the programming of a robot arm
Date of experiment:	29 November, 2016
Report submitted on:	6 December, 2016
To:	Amirhosein Khazenifard, B04
Lab Group No.:	44
Names:	A.-K. Blanken (V00809798)
	T. Stephen (V00812021)
	S. Todesco (V00821678)

1 Summary

This experiment uses a Lab-Volt 5250 robotic arm (see Figure 1) to demonstrate the differences between angular, linear and GUI control modes.

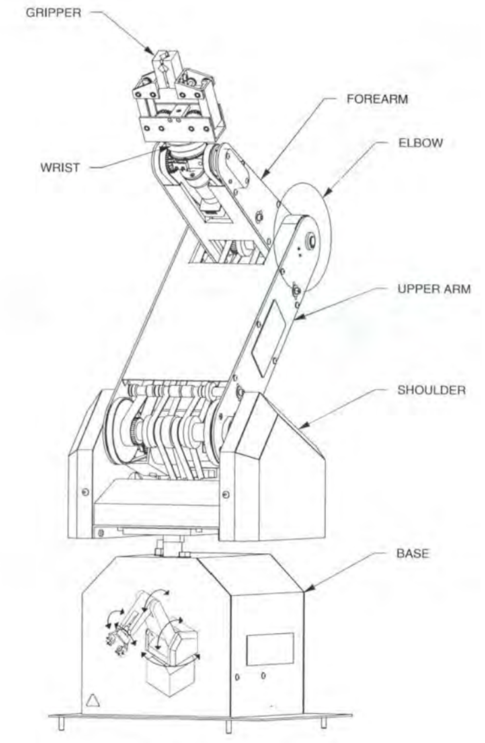

Figure 1: Lab-Volt 5250 robotic arm

Angular and GUI modes allow the user to control the position of each robot joint. Linear mode moves the tip of the robot's gripper through a Cartesian plane and determines the specific joint movements needed to achieve this motion.

2 Introduction

The robot arm moved a stack of three blocks into the pyramid configuration shown in Figure 2.

This task was completed three times, each using a different command input method. Starting at a known, fixed position (i.e. the *hard home* of the arm), the arm is moved through space to move each block. Critical points in the movement are marked as *points* in the program. When the program is played it runs through the entire sequence of points to move the blocks into the pyramid configuration.

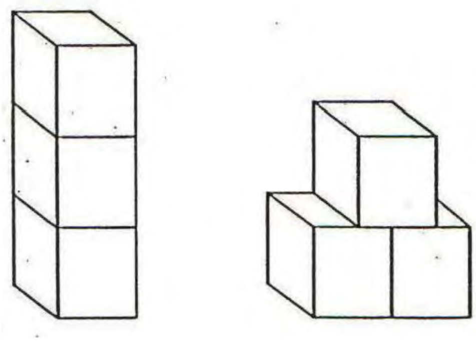

Figure 2: Starting (left) and final (right) block configurations

The *angular* input method allows each joint to be controlled independently. Points can be created between successive joint movements but may also be combined. The robot will execute all joint movements in parallel to transition between points.

The *linear* input method allows the user to specify series of *x, y, z* coordinates for the tip of the arm and the program will translate the $(x_i, y_i, z_i) \rightarrow (x_{i+1}, y_{i+1}, z_{i+1})$ spatial transition into the necessary joint movements.

Both angular and linear modes are controlled through the attached handheld controller (see Figure 3).

GUI mode records a sequence of joint measurements, identical to angular mode. The GUI (see Figure 4) provides the same functionality as the handheld controller but presents a history of the stored points on the right hand side.

3 Prelab questions

Answers to the prelab questions are attached to the end of the report.

4 Results

Pictures of us using the robot!

A listing of the code?

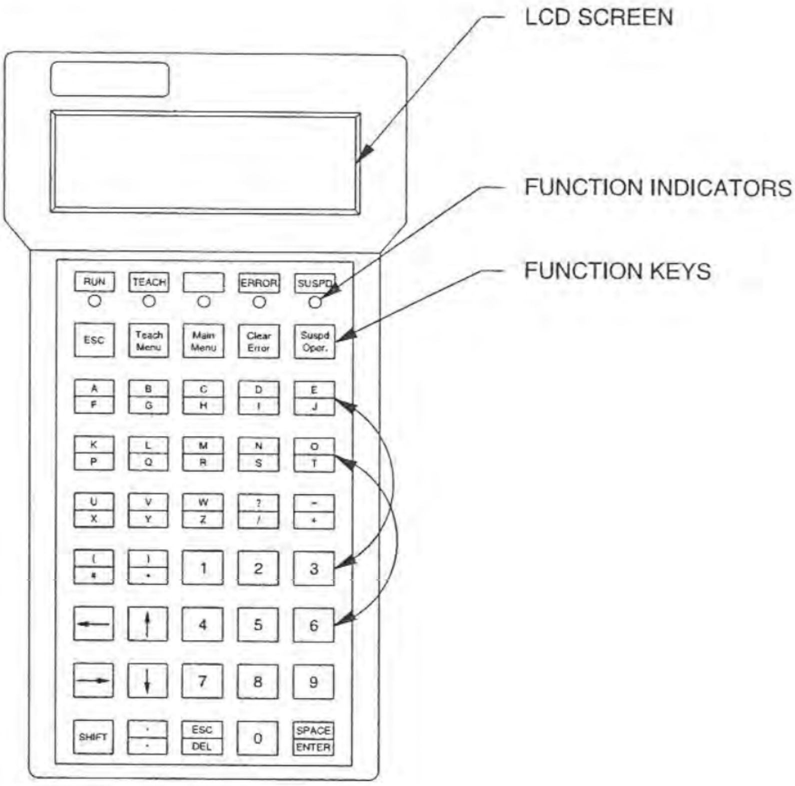

Figure 3: Handheld controller for Lab-Volt 5250

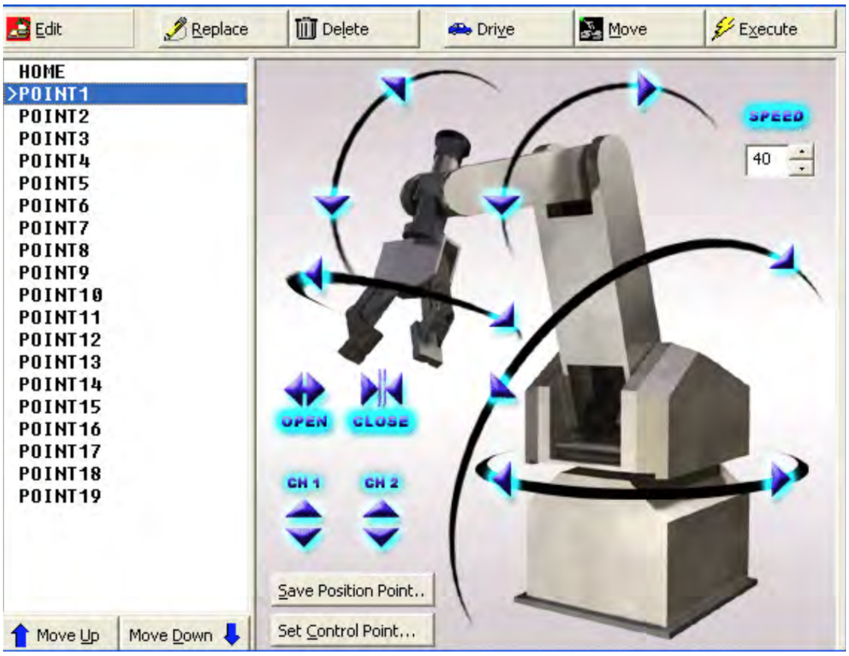

Figure 4: GUI for entering angular movement commands

5 Discussion

We completed pyramid stacking with angular and GUI modes but were unsuccessful with linear command mode.

The controller for angular and linear mode does not allow programs to be altered after creation. This frustrated development because motions were omitted from points during programming. The errors were not apparent until playback and even though it was obvious where to inject another point the series could not be altered and the entire series had to be abandoned.

Linear mode proved to be the most challenging because its method for resolving point transitions masked potential errors. Linear mode execution attempts to transition between points by moving each joint in succession to its final position. If the robot takes a path through space $A \rightarrow B$, the path is decomposed into $A \rightarrow B'_{shoulder} \rightarrow B'_{elbow} \rightarrow B'_{wrist} \rightarrow B'_{gripper} \rightarrow B$. However, the (manual) process to determine the next point involves moving several joints in non-sequential order. We experienced frequent movement out-of-bounds errors when playing back the sequence of linear points.

Though it is not mentioned in the lab manual, our experience in the lab indicates that there is a per-joint reordering occurring in linear mode. First, the arm was not observed executing parallel joint manipulations similar to those it performed in angular mode. Second, if an out-of-bounds error occurred in the elbow then the gripper was never observed to move even though the bounds for the two joints are independent.

Programming with the GUI was very similar to angular mode with the handheld controller. The GUI provided the same set of functions as the controller but the input was simpler because the inputs were mapped to a picture of the robot joints (see Figure 4) instead of what felt like random buttons on the controller. This allowed for faster command input than angular mode and quicker completion of the stacking task.

6 Conclusion

We were able to complete the stacking task with angular and GUI entry modes but were unsuccessful with linear entry mode. The difficulty with linear mode was the result of invalid joint positions demanded by the per-joint decomposition. This naive method for solving this problem is to store more points, since errors arise from the composition of many movements. A more optimal solution is to mirror the decomposition algorithm in point placement – always start movements from the shoulder out to the gripper and create new points whenever you have to correct movement “up” the arm.