SCARA GUI

Selective Compliant Articulated Robotic Arm Graphical User Interface

282778 Mechatronics
Assessment 4: “SCARA robot GUI project”
Jamie Churchouse, 20007137

0 Contents

1 Introduction

The School of Food and Advanced Technology (SFAT) possesses three Selective Compliant Articulated Robotic Arms (SCARAs). These SCARAs are controlled with an Arduino Due microcontroller board which actuates stepper motors to move the links of the arm, as well as solenoids to actuate the pneumatic piston and End Effector (EE). The objective of this project is to create a Windows application to convey commands to the SCARA via serial communication (a form of PC to microcontroller communication) which can be operated by the user with an easy to interpret and use Graphical User Interface (GUI).

The SCARA has three revolute Degrees Of Freedom (DOF) which are the joints at the base of each arm, and the joint at the end of the second arm that holds the EE assembly. The pneumatic piston at the end of the second arm has only the up and down state, and the EE has only open and closed states. These are the main functions of the SCARA; move to an X,Y coordinate, rotate the EE, extend/retract the piston, and open/close the EE. The SCARA comes with firmware pre-installed on the microcontroller - including pre-defined commands -, and accepts commands as a slave from the computer who acts as a master.

2 Methodology:

Describes the design process and the decisions made.
Outlines the methods used to implement the GUI.
Provides clear diagrams, schematics, and/or flowcharts to support the description.

GUI Requirements

This application is required to enable the user to both send valid commands, and manipulate the robot's settings all while being intuitive to use and pleasing to look at. This breaks down to four parts:

- Interface - UI is aesthetically appealing, intuitive to use, not cluttered
- Commands - can send every type of command
- Settings - can change every setting easily
- Validation - only valid settings / commands can be saved / sent

The flow of this project is fairly straight forward, and more or less comes in the order outlined above. First the UI needs to be designed and then implemented in XAML code. Next, communication needs to be established with the microcontroller. To aid in this, I took the code from the actual SCARA, removed all the functionality that pertains to movement, and refactored the code so that the serial commands act identically but have no action. This code was then uploaded to an Arduino Uno which then acted as my emulator for testing purposes.

Visual Studio

The project brief recommended the QT Creator (QTC) IDE with Python to develop this programme, however, I have gained much experience over the last six months using Microsoft's Visual Studio (VS) IDE. VS is also far more popular than QTC, and is used for many, if not most, development worldwide. VS's default language is the popular C# which has a great variety of documentation sources, and there is a large variety of NuGet packages available from within the IDE itself.

For this project, I opted to use WPF for the GUI design rather than WinForms. This is because WPF is more widely used nowadays and GUIs made with WinForms don't tend to scale well as easily. It also means that assigning C# functions as event handlers is merely a single parameter in the XAML code for buttons and text boxes.

Interface

Commands

Settings

Validation & Error Handling

Quality-Of-Life Features

Rescalable, auto connection, output files

3 Results:

Describes the outcome of the design and implementation process.
Includes data and/or measurements to support the results.
Provides clear and well-labelled photographs and/or videos of the GUI in action.

Images of the ui
Testing procedures
Output files

4 Discussion:

Analyses the results and the methods used.
Discusses the strengths and limitations of the design and implementation.
Provides suggestions for future work.

Did i meet the goal
What extra features would i add
Strengths and weakness of my design

5 Conclusion:

Summarizes the main findings and conclusions.
Restates the objectives and problem statement.

Overview of system and how it answers the question

6 Appendix

SCARA Functions

Type	Name	Command	Parameters	Returns	Description
Commands	Stop Robot	STOP	-	STOPPED	Stop the robot immediately. This command is processed differently to all other commands in that it is processed on receive rather than added to a buffer.
	Ping	PING	-	PONG	Ping the device to test communication.
	Home	HOME	-	HOME	Home the robot to 0,0,0
	Move to	MOVE	,*x*,*y*,*w*	MOVE: *x y w*	Move to coordinates *x*,*y* and rotate the EE *w*.
	Air control	AIR	,*c*[,*w*]	AIR: *c*[*d*]	Activate air function *c* with optional delay *d*.
	Wait for	WAIT	,*w*	WAIT: *d*	Wait a time in milliseconds, *w*, before proceeding.
	Ding	DING	,*msg*	DONG: *msg*	Send a message, *msg*, when the robot reaches that command.
Settings	Clear buffer	CLEAR	-	CLEAR	Clear the buffer.
	Echo commands	ECHO	,*b*	ECHO *b*	Turn on/off (*b*) confirmation of command receive messages.
	Set speeds	SPEEDSET	,*v*,*a*	SPEEDSET: *v a*	Set the velocity, *v*, and acceleration, *a*, values of the robot.
	Read offset	ROFFSET	-	OFFSET: *v*	Read the offset of the stepper motors, *v*.
	Set offset	SOFFSET	,*v*		Set the offset of the stepper motors, *v*.
	Get ID	ID	-	ID: *v*	Read the ID of the robot, *v*.
	Get proximity	PROX	-	PROX: *v1*,*v2*,*v3*	Read the values of the proximity sensors, *v1*,*v2*,*v3*.
ERROR	Negative acknowledge	-	-	NACK: *cmd*	Command was received, *cmd*, but is invalid.

asdf