MAJOR ASSIGNMENT REPORT

Robotic Arm

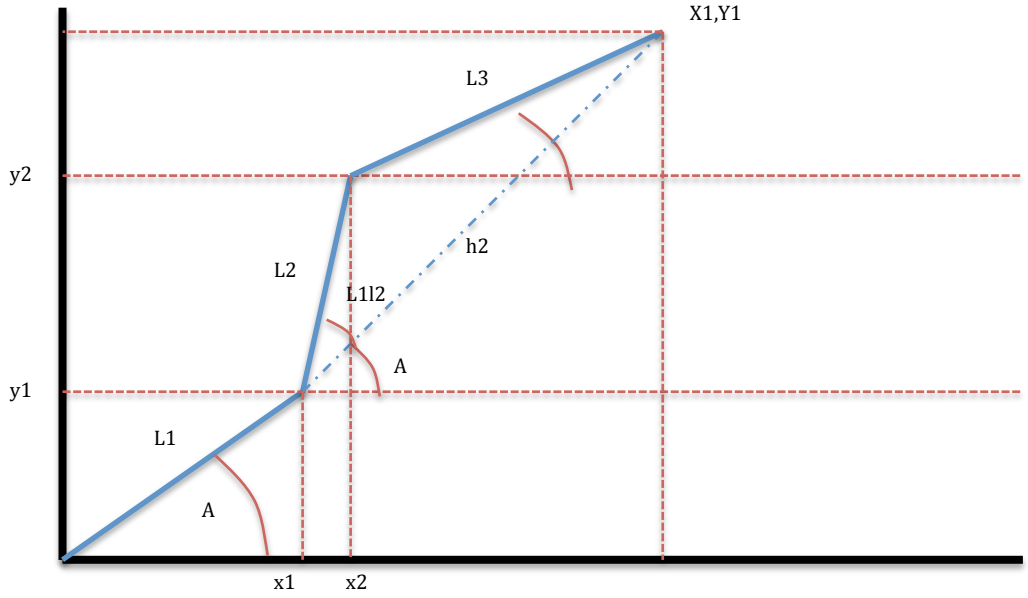
18-May-12

By: Dane Lennon

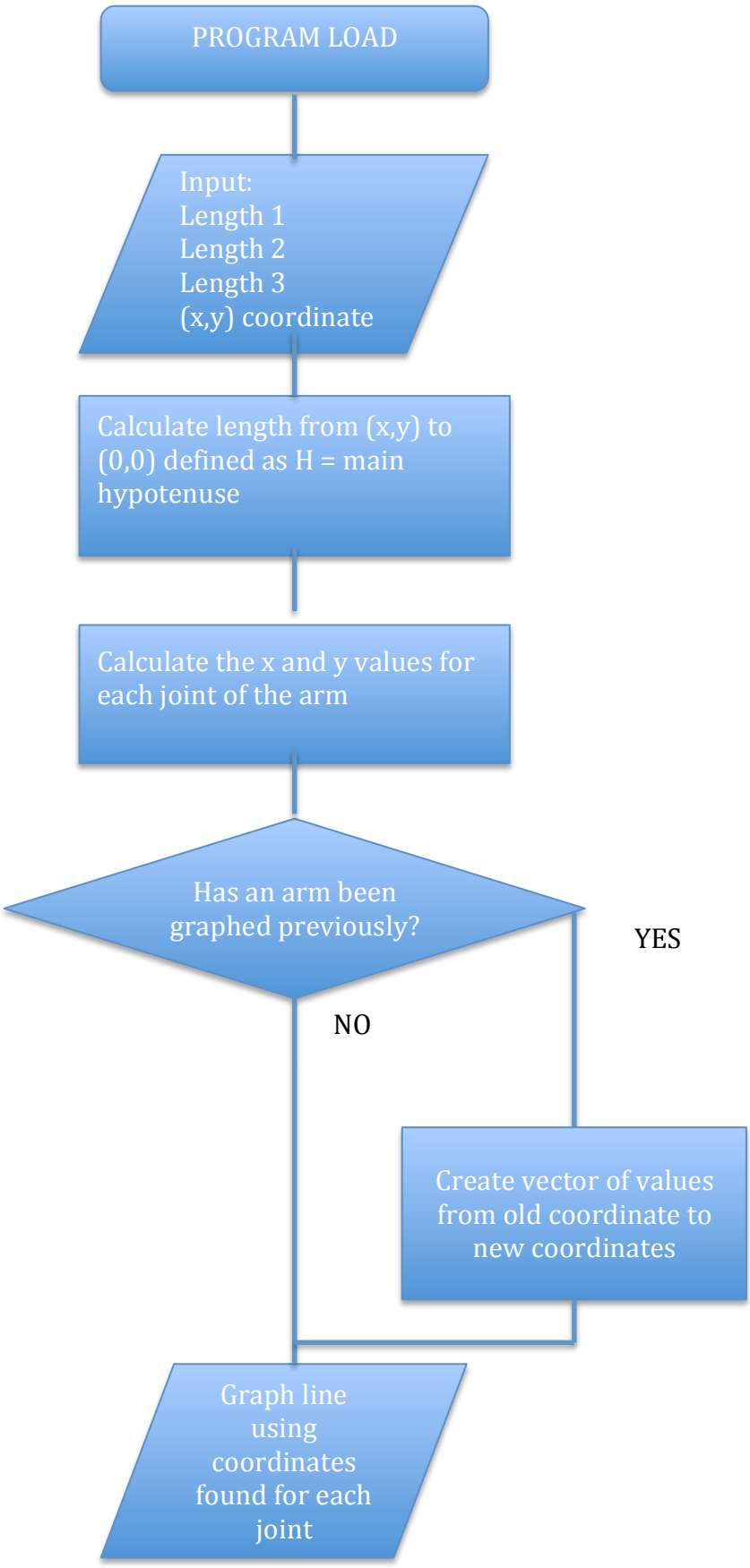
The project specified the requirement for a graphical representation of a robotic arm and its movement across a two-dimensional plane. The robotic arm had three defined lengths and a target within a specified range. The objective was to write and implement a solution that would be capable of taking an (x,y) coordinate along with three lengths and graph a possible position of the arm. Limitations included the ability to graph all possible solutions. This was because in most cases there were an infinite number of possible positions that the robotic arm could take. Further to this, there was no way to complete the project using one mathematical process. Due to the nature of a three segment robotic arm and the method of reverse robotic kinetics, multiple mathematical equations and procedures were required. For the program to be successful an input for the (x,y) coordinates had to be taken and the three segment robotic arm correctly graphed.

The method chosen was to break a three-triangle problem down to a two-triangle solution by forcing the first length of the arm to permanently point towards the final (x,y) coordinate. Following this, the positions of the final two lengths were easily calculated using the cosine rule and sine and cos relationships.

Following is the design method and an insert of the way the solution was analysed and accomplished.



Algorithm Flowchart



There are four important functions that this solution is built around:

```
1. function graph_button_Callback(hObject, eventdata, handles)
```

This function executes when the single bush button in the GUI is pressed. This function is the Parent Function with all other functions being called from this workspace. Firstly, the functions main role is to set up all the variables needed by taking the strings from the GUI edit boxes and storing them in variables. Secondly, the function calls the sub functions in order to graph the final outcome. Finally, once all the x and y values have been calculated using the required sub-functions it plots the values using a for-loop to simulate the movement of the robotic arm.

```
2. function [x1,y1,x2,y2] = calculate(object_handle,handles)
```

This Function plays the biggest role in the code structure as it is responsible for accurately calculating the angles between each join in the robotic arm and subsequently finding the x and y values of each length. It receives the handles structure for the GUI and returns four variables containing the x and y coordinates of each arm.

```
3. function [length_hyp] = calc_hyp(handles)
```

This is a small function that uses Pythagoras's theorem to calculate the length from (0,0) to the final x and y coordinate specified in the GUI.

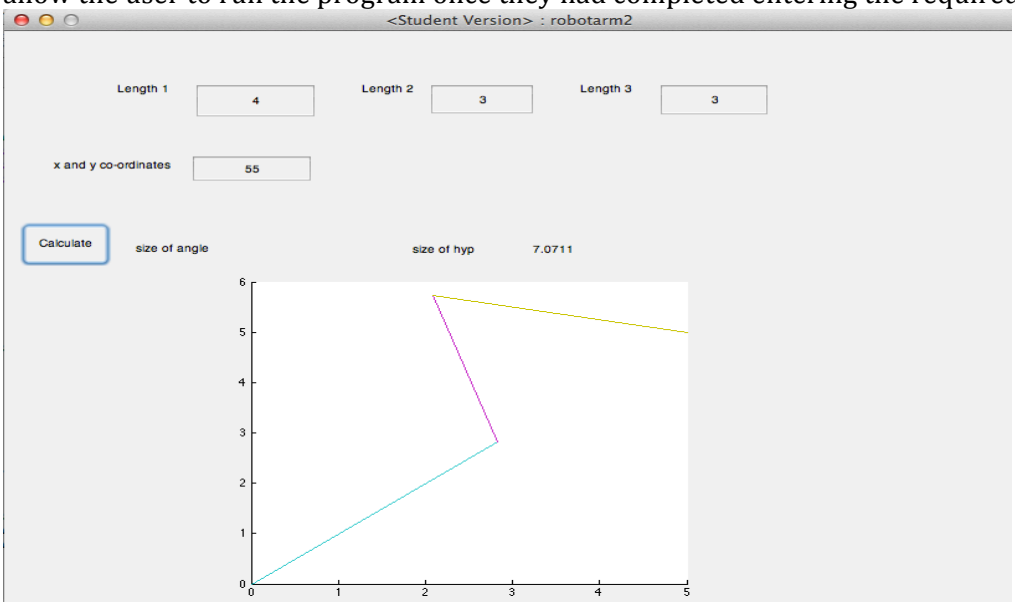
```
4. function set_range(object_handle, handles)
```

The function graphs a circle of the range that the robot arm can achieve with the given lengths. It is called a number of times throughout the code in order to maintain a graph of the range.

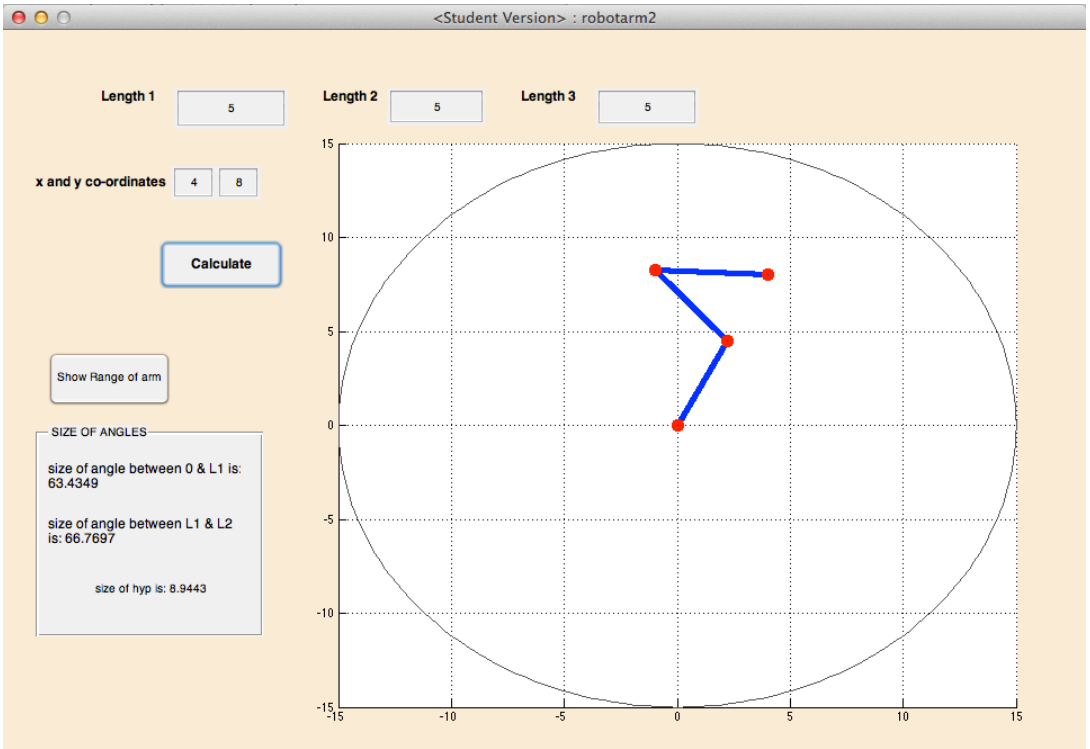
This callback borrows code from an unknown source however I have listed the web address below with links to the users profile on that page:

http://answers.yahoo.com/question/index?qid=20090503213042AA53zx4

The initial Graphical User Interface contained basic features that filled the criteria for allowing the user to input 3 lengths, an x coordinate, and a y coordinate. A pushbutton was supplied to allow the user to run the program once they had completed entering the required values.



The final GUI amended some problems that the draft GUI had. These included: a set x and y limit to ensure the axes did not automatically change during the movement of the arm, two edit boxes instead of one for the values of the x and y coordinates, and a panel box showing the user the angles found for each joint in the arm. Other features of the GUI that were included during the update were: a set colour for the arm, the ability to put the arm into a negative x and y state, a circle graph depicting the range of the robotic arm with the given lengths supplied, and more aesthetically pleasing width and colour for the arm.



The design specifications were successfully met to a level of 80%. The final solution graphs three lengths (L1, L2, L3) to an (x,y) coordinate. The solution also works within all values of the x and y axes including negative values. The movement of the arm was accomplished however it does telescope.

Not all error checks have been successfully confirmed. With regard to the user inputting lengths that are too long or out of range of the mathematical solution being used, some values are still not correctly recognized.

Some improvements for the solution would be the ability to extend the range of the arm beyond the x and y limits provided. With more time, another improvement would be to write more error checks on the lengths and (x,y) coordinates inputted as currently the program does have some minor bugs regarding these. The aesthetics of the arm could also be improved with a thicker, more robotic looking arm being used.