Imitating Robot Arm

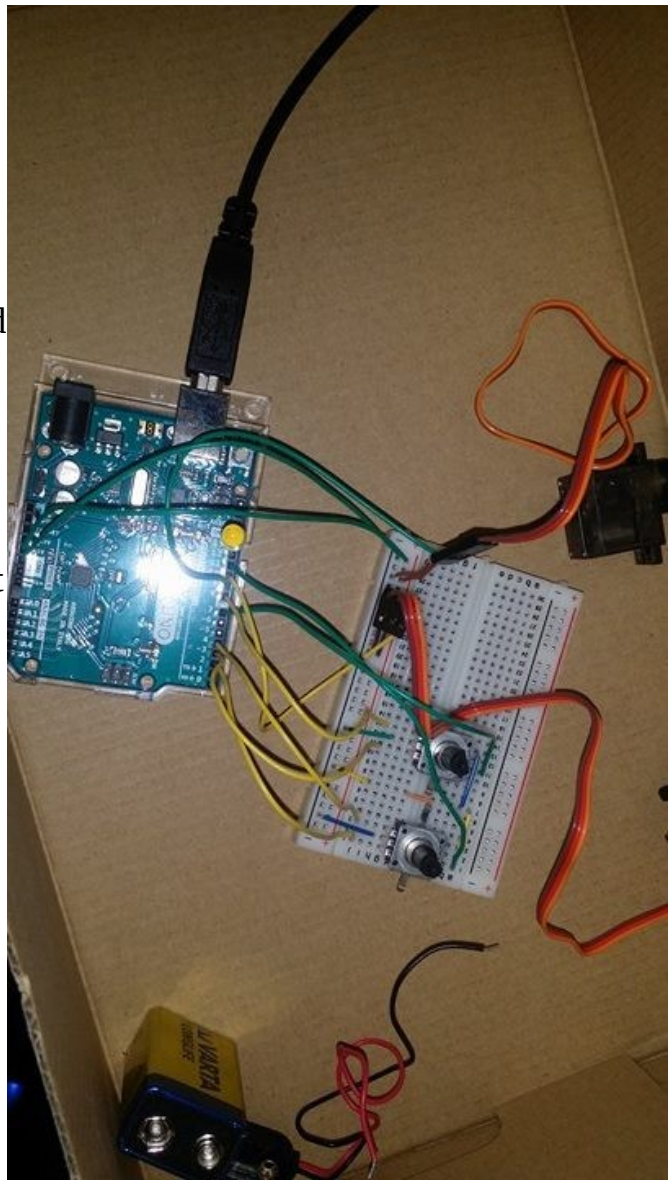
Project report

Embedded Systems Course

CSEN 701

Mohamed Ashraf Ahmed 31-13216
Ahmed Amr Mohamed 31-2273
Omar Mohamed Kamal 31-11151

Circuit Design:



In the image on the side, an overview of the circuit design can be seen. The exact pin out of each component is defined in the Arduino code using the #define directive. The initial design was to include a third rotary encoder and a third servo but due to financial difficulties, this is what we could buy. The design also included a robotic grip which we could not find so it was removed from the design. Also included in the initial design was a metal/wood cylinder to attach the encoders and the servo motors unto it but we could not fine an available/cheap material vendor thus it was removed from the design. Though a lot of differences exist between the initial and final design, the core part of the code will be similar and all the rest functionalities will be mere repetitions of existing code.

Implementation:

The components used were:

- 2 servo motors
- 2 rotary encoders with button switch
- 1 LED and the built in LED
- Arduino built in pull-up resistors
- Breadboard
- 9v battery and cap
- Male-male jumper wires

This project has 3 modes which can be manipulated using the buttons on the rotary encoders. The 3 modes are:Track, Code and Execute. The original idea was to use separate buttons but we were short on jumper wires. The projet starts on Track mode where the servo motor follows the rotation of the rotary encoder in realtime. Both modes are accessible from track mode where pressing the button on rotary encoder 0 will enter code mode and pressing the button on rotary encoder 1 will enter execute mode. In code mode, the servo will not follow the encoder but the arduino will continue to read the encoder values. On pressing button 1, the position of the encoder will be saved in an array of maximum 10 positions. On pressing button 0, code mode will be exited and we are back in track mode. When pressing button 1 in track mode,

the controller will enter execution mode where it will execute the saved positions in order of saving in a continuous loop. Note that if the program needs to be changed , the whole system needs to be restarted. When pressing button 0 in execution mode we will enter code mode and when pressing button 1 we will enter code mode. Simply put:

Mode	Button 0	Button 1
Track	Switch to Code	Switch to Execute
Code	Switch to Track	Save Location
Execute	Switch to Code	Switch to Track

Rotary encoder readings are handled by an ISR while the Servo motor movements and the button readings and done in the Loop. The internal LED in the Arduino is lit when we are in execute/track mode and is not lit when in code phase. The external LED serves 2 functions: In code mode when pressing button 1, it lights up indicating the position is actually saved and it switches off when the rotary encoder is rotated. It's second function is to distinguish between track and execute mode (because the built in LED can not represent 3 states), lighting up when in execute and is off when in track. The only exception to that second case is when the controller is in the code mode and we just saved the location then immediately went into track mode without moving the rotary encoder, in this case the light will remain on till the rotary encoder is moved.

Challenges:

The main challenge faced while making this project was knowing the pin out and the output format of the rotary encoder component as the datasheet of the component was lacking and difficult to comprehend. The youtube video linked below in the references was the main source of understanding how the rotary encoder works. Other than that, all the other wiring and code were just a matter of time.

Future improvements:

- Mount the rotary encoders and the servo motors on actual material to be of use in several applications
- Provide a button to clear the saved positions instead of restarting the whole system
- Provide a robotic grip on top of the slave arm and supply the code and inputs to run the grip

- allow the saved position list to wrap over and replace old values with new values when the saved positions are over 10 (the array limit) instead of simply discarding the new values

References:

http://playground.arduino.cc/
https://www.youtube.com/watch?v=HQuLZHsGZdI