**Feroze Naina M Dheen Mohamed Ismail**

**Team D - HARP**

**Teammates – Alex Brinkman, Rick Shanor, Abhishek Bhatia, Lekha Mohan**

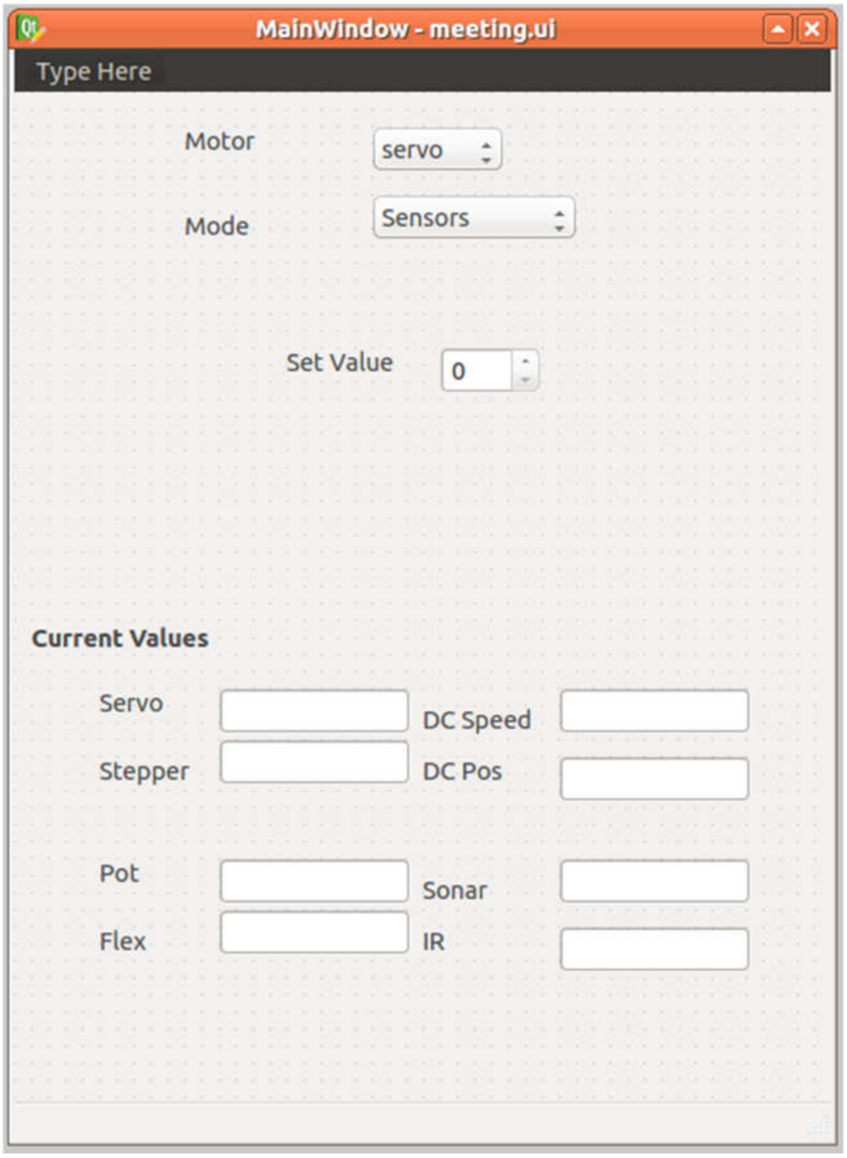
**ILR01**

**Oct. 16, 2015**

## Individual Progress

For the sensors and motors lab, I setup an initial Arduino code template which handles state machine and serial interfacing so that different subtasks could later be integrated into one Arduino program easily. I defined the serial communication structure to transfer data on state, mode and current motor-sensor pair between the Arduino and the GUI. I also decided on using Python and Qt for the GUI as I had prior experience working with it. Pyserial library was used for interfacing the GUI with Arduino. I was responsible for controlling the servo motor using potentiometer input.

### GUI – Qt + Python + Pyserial



Motor: servo
Mode: Sensors
Set Value: 0

Figure 1: Qt GUI

The Qt GUI (Figure 1) was designed using Qt4 Designer. We had initially planned to display all the sensor and motor values simultaneously in the GUI as shown above. However, due to complexity, we simplified the GUI to just display and modify one motor and sensor value for the demo. In the ‘sensors mode’, the motors are controlled by changing the sensor inputs. The motor can be changed by pressing the push-button on the circuit. The user can set the value of the motor from the GUI by switching to ‘manual mode’. We decided to display motor position in relative values from -100% to 100% and map it to the absolute values inside the Arduino code to maintain uniformity.

**Servo Motor – Potentiometer Control**

A potentiometer was used to control the angle of HiTEC HS-225BB servo motor. The servo can turn from 0 to 180 degrees and the position of the servo can be controlled by supplying Pulse Width Modulated signal. Arduino has an inbuilt servo library which was used. Analog input from the potentiometer was mapped from 0-1023 values to 0-180 degrees and written to the servo.

## Challenges

We had difficulty getting the pyserial to work with GUI. The serial port was being accessed inside the GUI loop, causing the Qt GUI to freeze. This was solved by using multi-threading.

Initially, we had planned on using the potentiometer with the servo and DC motor position control with the sonar sensor. However, the DC position controller was very sensitive to noise and output from the sonar range sensor wasn’t very stable. So we switched to using potentiometer for DC motor position control.

## Team Work

We divided the tasks so that we can initially work on our modules independently and later combine them. We had four members who worked on controlling a motor-sensor pair.

Alex worked on DC motor speed control using pressure sensor, encoder library and GUI implementation. Rick worked on DC position control and encoder library. Abhishek worked on stepper motor control and also integrated the Arduino code. Lekha worked on the GUI implementation.

Alex, Abhishek and Rick worked together on debugging the integrated code.

**Plans**

For our first project review, I will get the ROS nodes for the robot state controller, Kinect and suction gripper working. I will be collaborating with Alex to define a software specification document which clearly details what function every ROS node performs and what its inputs and outputs are. This will enable our team to work on software subsystems independently.

We have also been given permission by Prof. Maxim Likhachev to use the PR2 robot at the Search Based Planning Lab. I will setup the simulation environment for testing the kinematics of PR2 on RViz and make the robot arm move from one position to another using a path planner. We will also be getting the Work Breakdown and Milestones schedule ready.

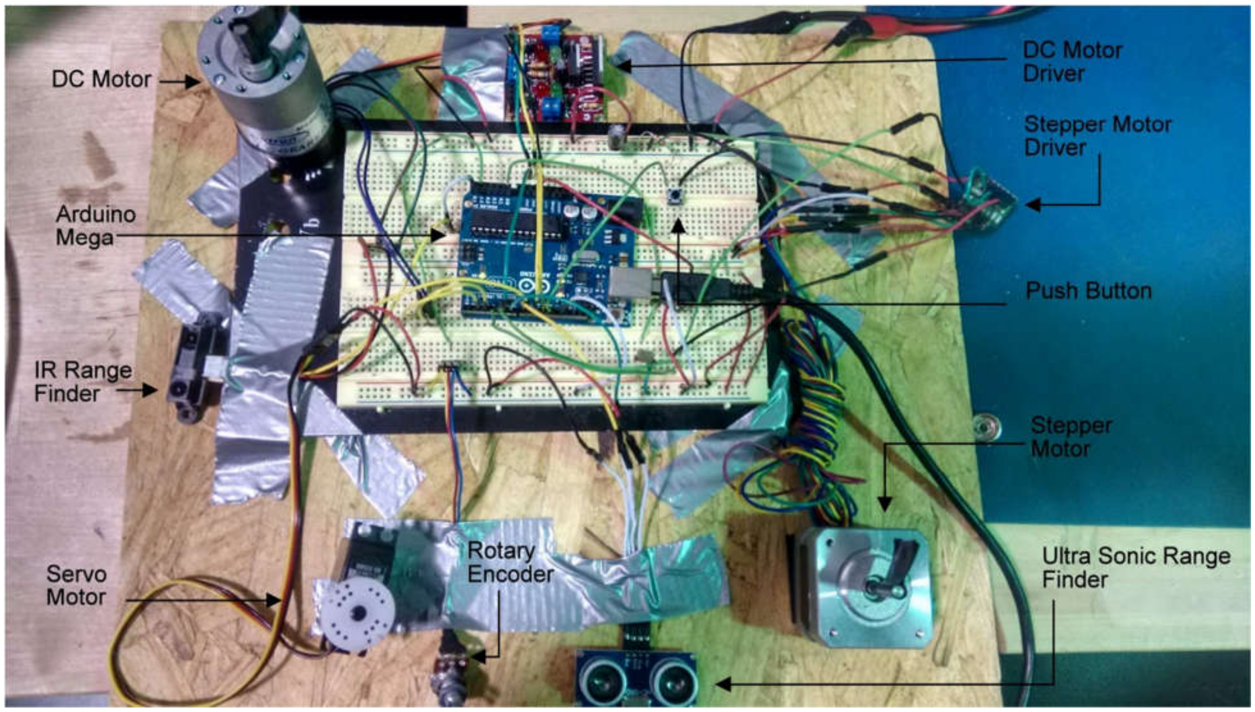


Figure 2: Sensors and Motor Lab Demo Setup