Wesley Myers
18-578
Team F
Individual Lab Report
Week 4

Individual Progress

I took the lead with designing the initial new wiring for the system. I consulted with the team on which sensors to use for the demonstration that pertained to our project. After determining sensors and wiring, I assisted in entering this information into our wiring chart.

The main thing I've been working on this week is getting the CMUcam4 to work with an Arduino. I set up a basic test bench outside of the Mechatronics lab as illustrated in Figure 1. Was first step was just to test the basic interface with the CMUcam4 via the Parallax Serial Terminal on my Netbook. There were some problems connecting to the CMUcam4 with the FTDI cable, but I quickly overcame them. I tested out different commands that we wanted to use for the project. The most important commands are setting the window, setting the color, and then getting the center of mass of the blob seen. The setting the window function worked perfectly. Since we know the targets will be 10 feet away and that it will be in a 4 foot by 3 foot frame, we can set the "window" seen by the camera such that it only sees the frame and nothing else in the background or foreground. This makes filtering much easier. Next is setting the color models for the various targets. The way this is done is by setting an RGB range. I took a photo with the CMUcam4 and uploaded it to my computer. Figure 3 is the original image I got. I used an image tool to see the RGB values of the pixels on the target. From there I pieced together a range of acceptable RGB values that I wanted for each target. After inputting the ranges into the CMUcam4, it worked as expected. Next I was able to get the center of mass data from the device.

After sending the CMUcam4 basic commands with the Parallax Serial Terminal, I created cables to attach the CMUcam4 and Arduino UNO together as illustrated in Figure 2. With this, the Arduino can power and communicate with the CMUcam4. I was able to write code to communicate back and forth with the CMUcam4. The Code section of this report contains code that I wrote to do this. I was able to transmit commands, but I have not been able to receive data back yet over serial.

What I quickly realized was that finding the images will be very challenging due the changing light. During the demonstration, I saw that the CMUcam4 had difficulty finding the targets since it was a different light source. One thing I discovered was the problem of using the reflective tape. While it is good for the Infrared Cameras, any light that reflects off of the tape shows as something bright, pretty much white. At the moment I'm not completely sure how to fix this.

My other goal was to help Richard Musgrave learn how to program. I assisted as necessary and helped him fix logic and helped him understand how the electronics work. He was a quick study.

I assisted as needed to help get the sensor lab ready for the demonstration. I helped properly wire the limit switch, as shown in Figure 4. There is a 4.7 kOhm resistor hidden underneath the shrink-wrap going to ground. I also helped with initial assembly of the Photomicrosensor, but then passed it off to David and SooHyun. The assembled lab is shown in Figure 5 with important components labeled.

Challenges

A challenge that we ran into was getting the Photomicrosensor working. We had some issues with getting the voltage and current correct for the emitter and collector. The specification sheet was unclear as to what was correct for the current and voltage. We realized that what mattered more was the current flowing through the transistor. The specification sheet wanted 10 volts, but we decided we could instead use 5 volts and use a different resistor value. In any case, that was a bit of a mystery to figure out. Rick and I spent the first night trying to get it to work, but we didn't realize our current was way too high for the transistor. The transistor was in saturation, but I didn't realize it. The next night, David and SooHyun took a look at it and were able to complete it with some assistant from Curtis Layton.

I ran into many issues with trying to communicate with the CMUcam4 serially. It was challenging due to the fact that there isn't much documentation on how to do this. I had to look to the Arduino API to figure out how to do this.

Another bug I ran into was an Auto Reset problem with the Arduino. For some reason when something is plugged into pins 0 (RX) and 1 (TX), and you try to upload new code to the Arduino, the auto reset kicks in and the code cannot upload. This took me a while to figure out. New models of the UNO should not have this problem, so the one I have is an older version. Confirmation occurred when I checked the location of the reset button. The new UNOs have the reset buttons moved the corner of the board. There is much documentation on how to fix this problem with some removal of components, but I felt it was unnecessary given that this is just for testing. Documentation says that the Arduino MEGA, which is what we are really using, does not have this same problem.

Teamwork

This week we split up the work. Richard, SooHyun, and David were tasked with getting the sensors for the demonstration working. My role was to help Richard with learning Arduino C++ and getting the CMUcam4 functioning (or make

progress). We wanted to have it working for the demonstration since this is really just a huge sensor.

One of my goals this week was to help Richard Musgrave with getting familiar with the Arduino interface. He has some experience with coding, but it is a bit rusty. I let him try to figure out everything and assisted when necessary. He was basically able to get the Sharp Range Finder and Potentiometer with minimal assistance from me. He was able to read the Analog value from the Sharp Range Finder, but he did not linearize it.

Once most of the basic work was done, we let Richard work on the CAD for the system. Most of the weight of the project fell on David's shoulders since he was presenting. SooHyun and David spent a lot of time trying to make sure everything worked. I came to help as necessary to steer them on the correct path. Much thanks to everyone for getting everything done!

Future Work

The work with the CMUcam4 is probably the most important thing right now. It will be the most important feature in our system, enabling it to see the targets. The limit switches, both the physical switch and the photomicrosensor, will be very valuable in determining the initial position of the shooter. We will decide which one to use at a later date. We will probably use the potentiometer in the initial testing phases to help point the shooter in an appropriate direction. The Sharp Range Finding sensor will most likely be used to detect a ball being pushed into the shooter. We are also considering using a basic Infrared Sensor to do this.

As for work for the work for next week, we already mapped out what the state machine looks like. Richard can't be here this weekend, so we plan on just getting the basic infrastructure working and then having him help with assembled the demonstration of the state machine for our system when he returns.

Figures

Figure 1. Testing Layout

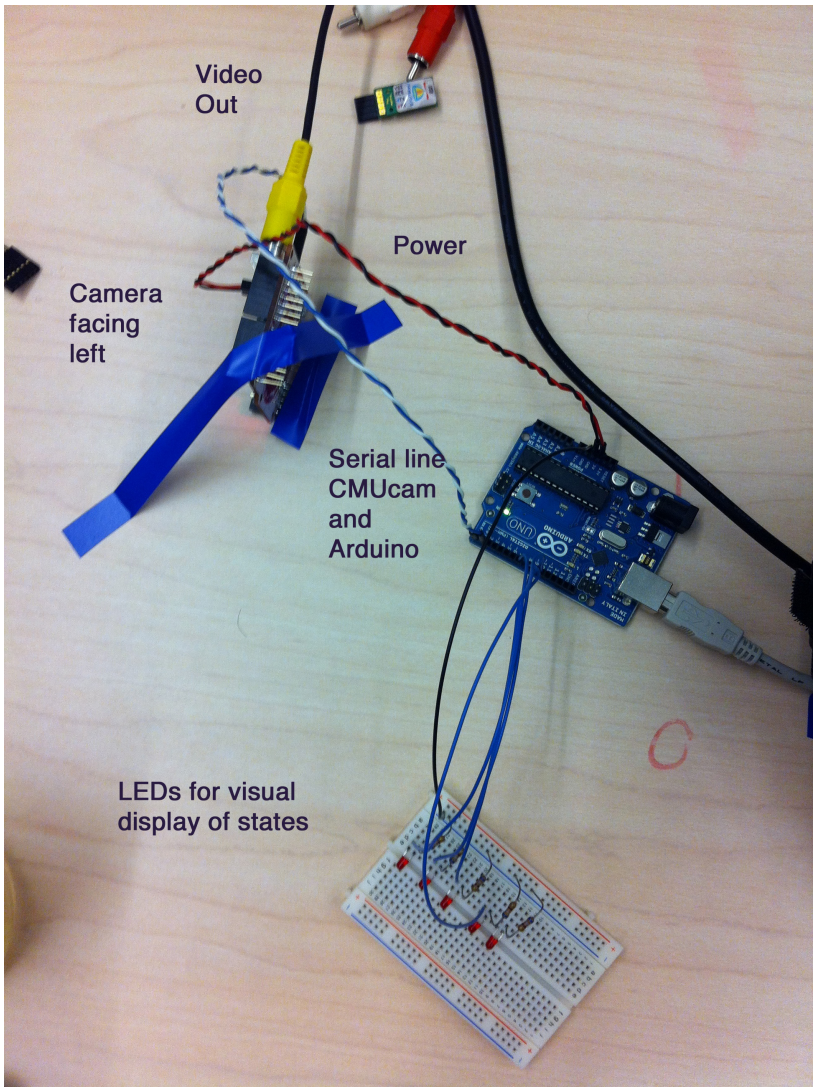

Figure 2. Communication and State Display

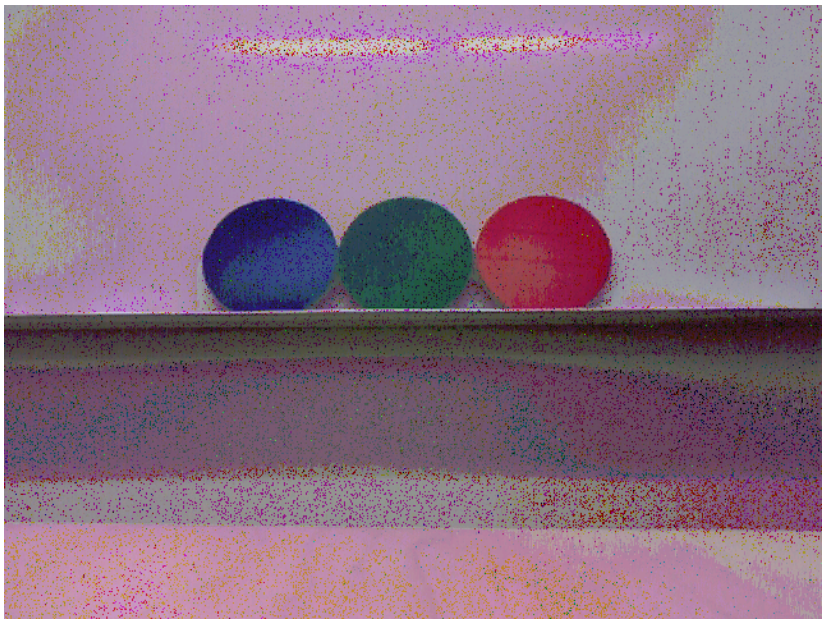

Figure 3. Image as seen from CMUcam4

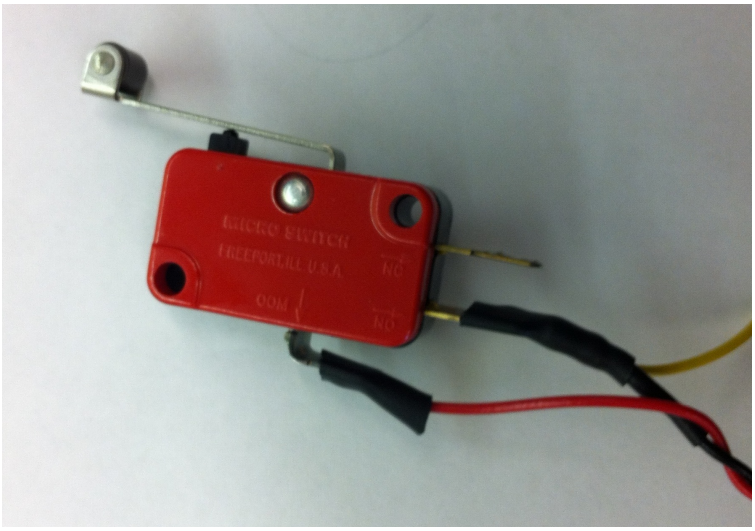

Figure 4. Limit Switch

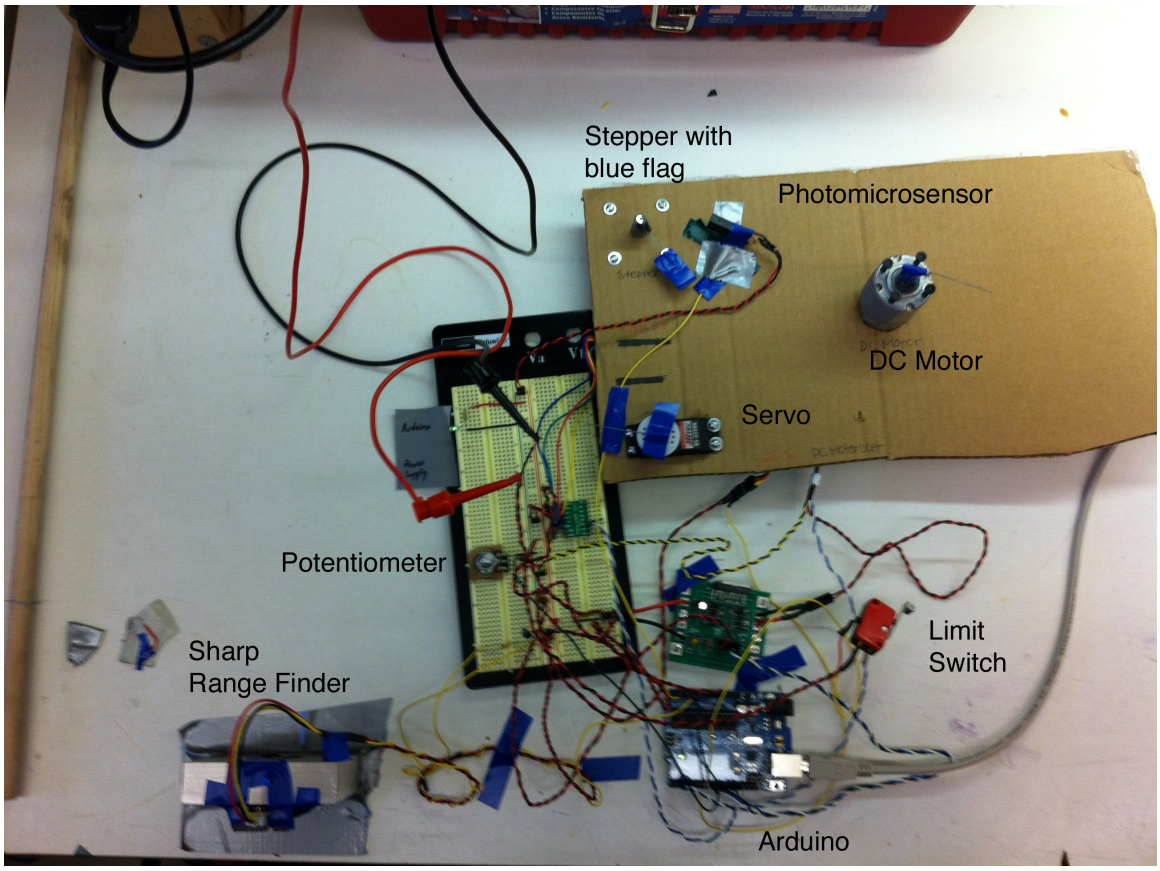

Figure 5. Sensor Lab Component Layout

Code

```
#define SERIAL_BAUD 19200

#define CMUcam Serial

#define out1 7
#define out2 8
#define out3 9
#define out4 10

int mode;

//sends a command, ignoring the result
int cmuSet(char * command)
{
  int byteCount;

  //send command and \r
  CMUcam.print(command);
  CMUcam.print("\r");
  delay(10);

  //rcv "ack\r:"
  while(CMUcam.available() == 0);
  CMUcam.flush();
}

void setup()
{
  delay(5000);  //let cmucam settle down
  CMUcam.begin(SERIAL_BAUD);
  delay(5000);

  // LEDs for STATE
  pinMode(out1, OUTPUT);
  pinMode(out2, OUTPUT);
  pinMode(out3, OUTPUT);
  pinMode(out4, OUTPUT);

  mode = 1;
}

void loop()
{
  if(mode == 1)
  {

    digitalWrite(7, HIGH);
    cmuSet("nf 5");                  //set a filter on length of pixels
    cmuSet("st 30 170 15 50 60 90"); //Look for RED
    delay(10000);
    digitalWrite(7, LOW);


    digitalWrite(8, HIGH);
    cmuSet("st 30 50 65 100 65 85");  //Look for GREEN
    delay(10000);
    digitalWrite(8, LOW);

    digitalWrite(9, HIGH);
    cmuSet("st 30 50 70 100 110 135");    //Look for BLUE
    delay(10000);
    digitalWrite(9, LOW);

    mode = 2;
  }

}
```