Universiteit Antwerpen

Adjusting program + Command parsing

De Borrekens Gauthier

Intro

For this week I have added and adjusted some things on the Arduino software. I also worked on the final report.

Materials and Methods

Arduino IDE

Results

This is a preparation and nothing is tested yet. This will happen in the meeting when the car is properly set up.

Information & conclusion

Software

As the PWM results were very inconsistent last week, I have added another method to adjust the pwm of the car relative to the HC-SR04 input by checking specific values and in or decreasing the PWM by a fixed amount at specific offsets.

```
if(distance < 75){              //0 - 74
  duty_cycle = 255;
}else if(distance < 85){        //75 - 84
  duty_cycle += 20;
}else if(distance <95){         //85 - 94
  duty_cycle += 10;
}else if(distance < 105){       //95 - 104
  //duty_cycle =
}else if(distance < 115){       //105 - 114
  duty_cycle -= 10;
}else if(distance < 125){       //115 - 124
  duty_cycle -= 20;
}else{                          //125 - ?
  duty_cycle = 0;
}
```

I have also added a test to see if we can parse data from udp packets. What it does for now is it checks if the first character of the packet is ‘b’ and if this is true, set the duty cycle to 0 and stop auto adjusting the distance. This is just a test to see if the UDP parsing works.

```
if(packetBuffer[0] == 'b'){
  duty_cycle_udp = 0;
  autoAdjust = false;
}
```

HC-SR04

Although the test a couple of weeks ago seemed quite accurate, I will still switch to a 5v power supply instead of the 3.3V.

We can take this power supply directly from the H-bridge, which gives a 5V voltage.

The downside is that the echo pin will give a 5V signal back and we cannot connect this to the nodeMCU. We need a voltage divider that divides the 5V echo pin to a usable 3.3v.

Calculations:

3.3V/5V = R2/(R1+R2) : R2 / Rtot = 0.66

Values:

R2 = 200

R1 = 120

V = 3.2V