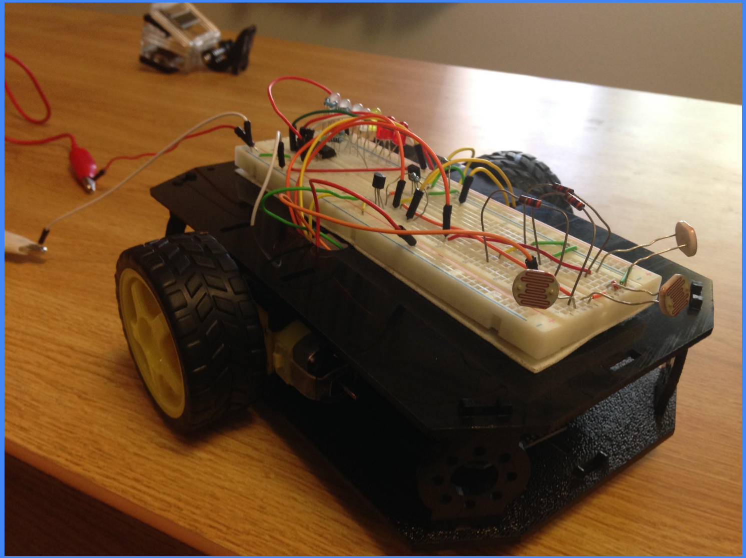
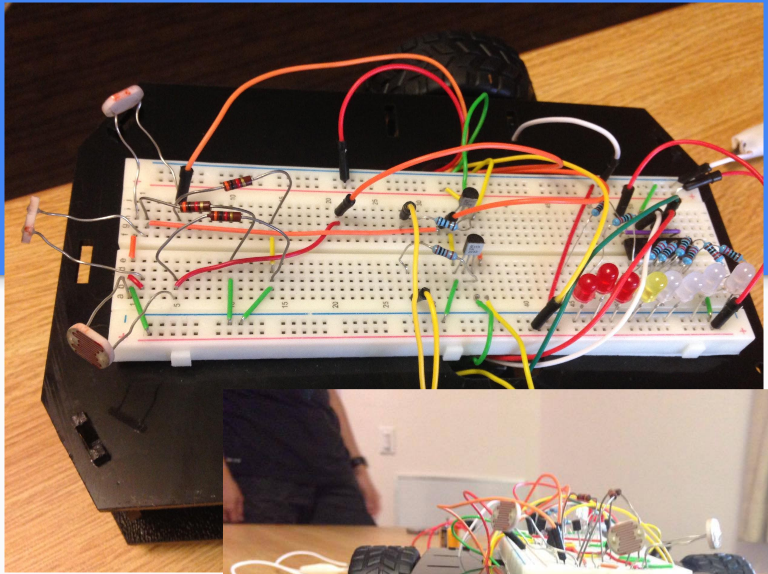
Light-Sensing Car

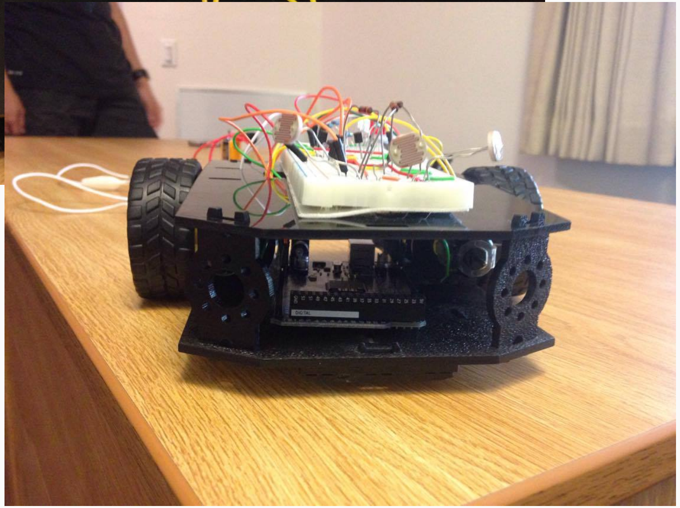
Summer Session C 2016

Members: Ben Bowen, Alexander Chen, Albert Shu



Project Objectives





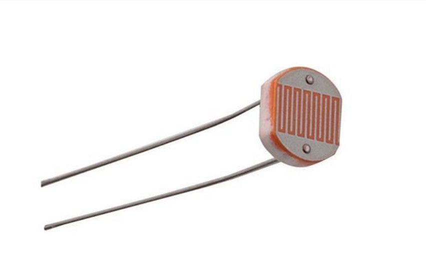
1. Car made of two systems
    a. Light detection system
    b. Drive control system

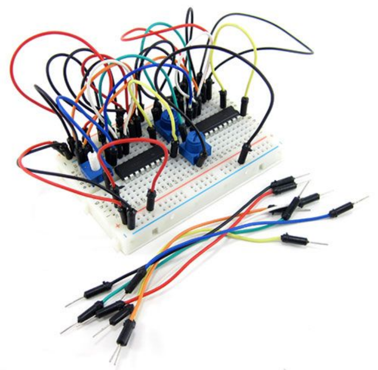
2. Speedometer

    c. Shift-register IC (74HC595)
    d. LEDs represent average motor speed

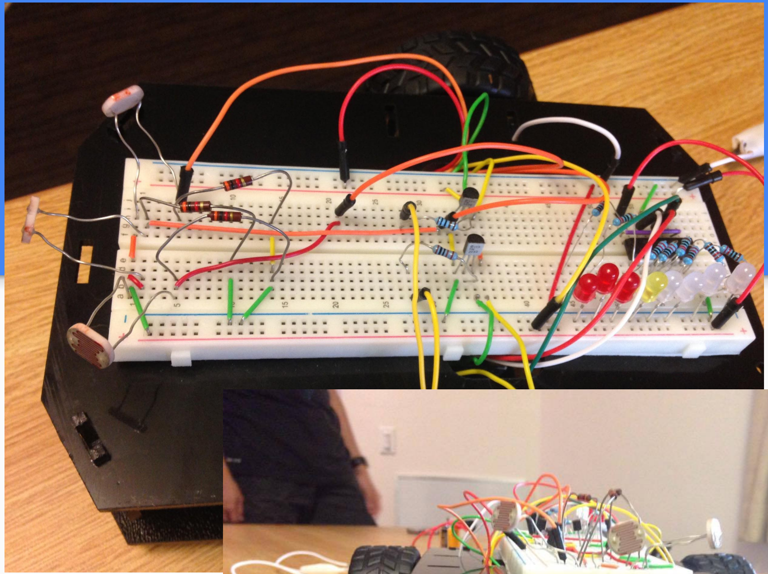
Materials List

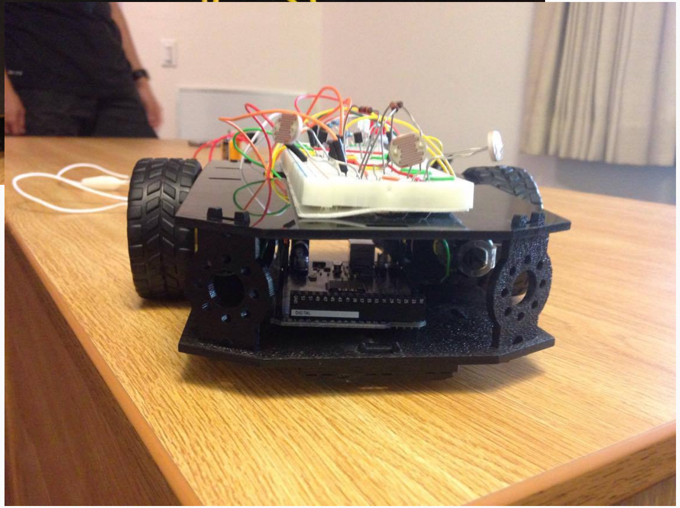
- Car Chassis complete with 2 wheels and 2 motors
- Arduino UNO
- Transistors
- Photoresistors
- 9V Battery
- x8 LEDs
- 74HC595 IC (shift register)
- Breadboard and a whole lot of jumper cables...





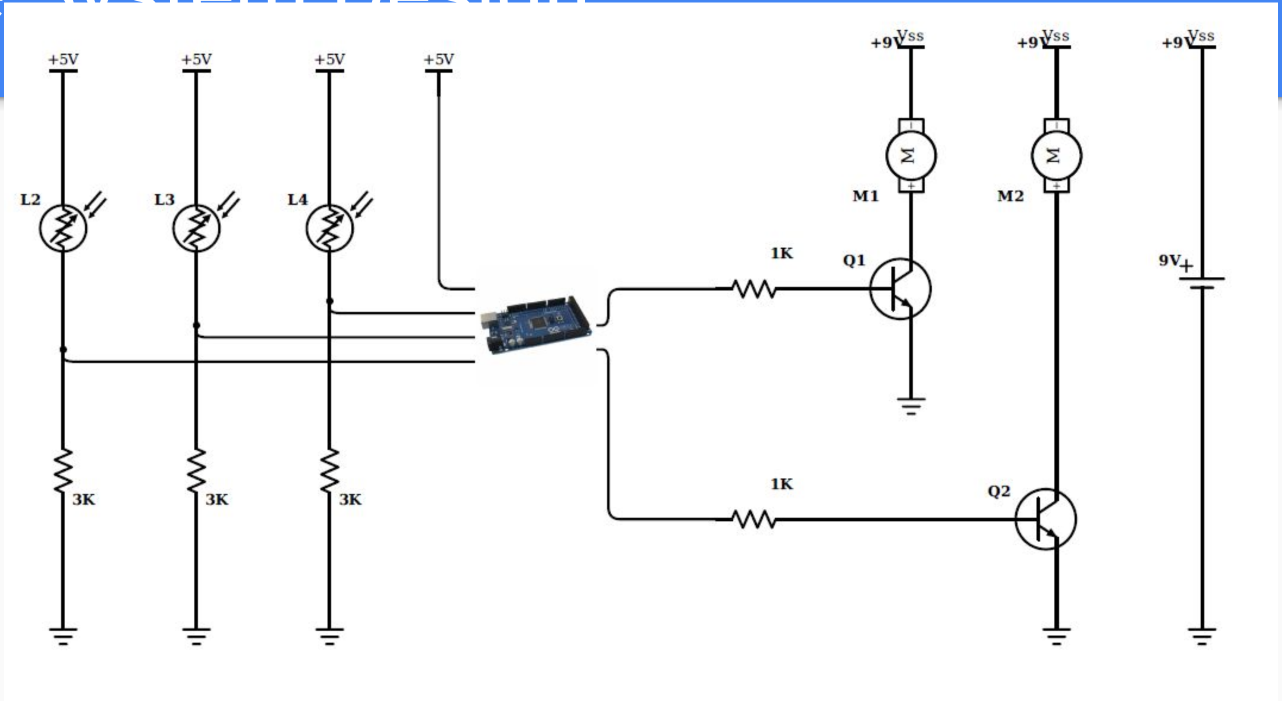
Drive Systems



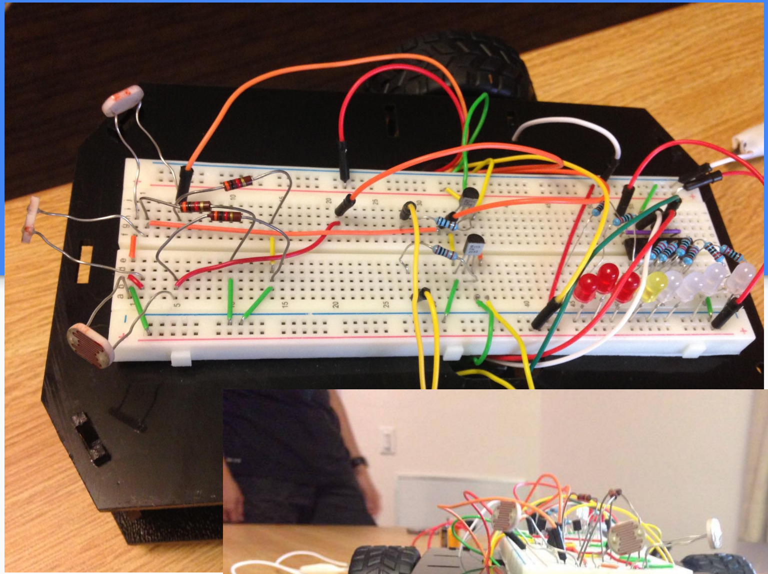


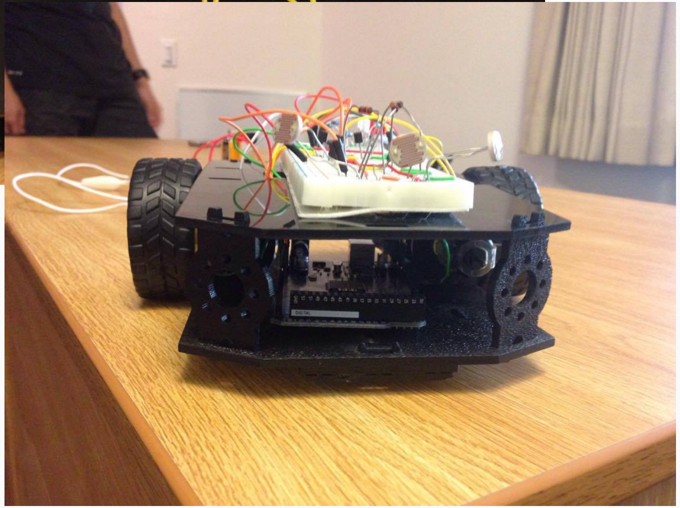
1. Light detection System
2. Drive Control System

Drive System Design



Display System





Speedometer

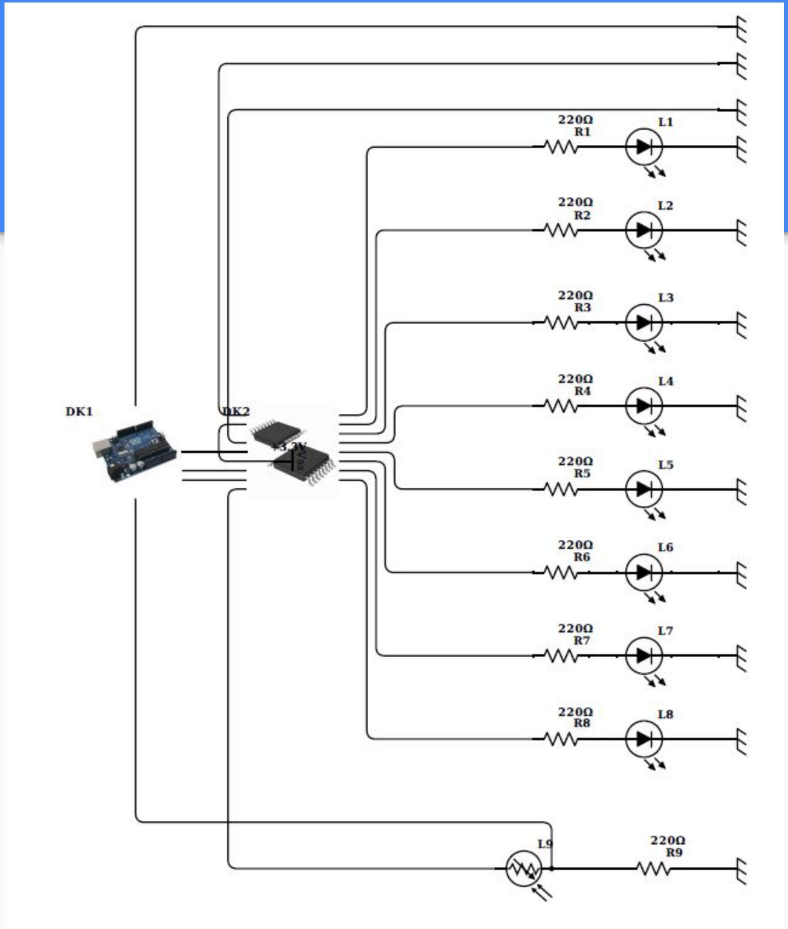
a. Shift-register IC (74HC595)
b. LEDs represent average motor speed

Display System Design

-74HC595

-LEDs

-resistors



Coding - initialization

Initial Values--used for calibration

Dynamic Values--used while driving

EE_3_Final_Codes_With_Comments

```
// pins for the LED
int latch = 8;
int clock_ = 9;
int data = 7;

int leds = 0;

//pins for the sensor and motor system
// creat three variables to store the initial light signal from the three photoresistor.
int initial_sensor_right_Value;
int initial_sensor_left_Value;
int initial_sensor_front_Value;

// These three variables are used for dynamicly recording the light signal from the three photoresistor.
int dynamic_sensor_right_Value;
int dynamic_sensor_left_Value;
int dynamic_sensor_front_Value;

// We will explain the following variables in the place where we will use them.
int input_right = 0;
int output_right = 5;
int right_value = 0;
int right_difference;

int input_left = 2;
int output_left = 12;
int left_value = 0;
int left_difference;

int input_front = 4;
int front_value = 0;
int front_difference;

int max_difference = 0;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(output_right, OUTPUT);
  pinMode(output_left, OUTPUT);
  pinMode(latch, OUTPUT);
  pinMode(data, OUTPUT);
  pinMode(clock_, OUTPUT);
```

Initialize clock, latch, and data pins for IC

Initialize variables for initial and dynamic sensor readings

EE_3_Final_Codes_With_Comments

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(output_right, OUTPUT);
  pinMode(output_left, OUTPUT);
  pinMode(latch, OUTPUT);
  pinMode(data, OUTPUT);
  pinMode(clock_, OUTPUT);

  delay(3000); // delay 3 seconds before we recording the inital light intensity since the initial light
               // intensity might fluctuate (like our hands do not move quickly after setting up the car
               // and the hands, in this case, might block some light.etc)

  initial_sensor_right_Value = analogRead(input_right);   // record the initial enviroment light intensity in order to detect the light change later.
  initial_sensor_left_Value = analogRead(input_left);     // by doing this, we can control the car by light not only in the darkness(zero initial light intensity)
  initial_sensor_front_Value = analogRead(input_front);   // but also in any initial light conditions like the normal classroom (do not need to turn off the light)

}

void updateShiftRegister()
{
   digitalWrite(latch, LOW);
   shiftOut(data, clock_, LSBFIRST, leds);
   digitalWrite(latch, HIGH);
}

void loop() {
  // put your main code here, to run repeatedly:
  dynamic_sensor_right_Value = analogRead(input_right);  // keep reading the light signal detected by the photoresistors
  dynamic_sensor_left_Value = analogRead(input_left);
  dynamic_sensor_front_Value = analogRead(input_front);

  front_difference = dynamic_sensor_front_Value - initial_sensor_front_Value; // calculate the light intensity difference between
  right_difference = dynamic_sensor_right_Value - initial_sensor_right_Value; // the current light intensity and the initial environment light intensity
  left_difference = dynamic_sensor_left_Value - initial_sensor_left_Value;

  if (front_difference > right_difference)     // find the max light intensity difference among the three photoresistor because
    max_difference = front_difference;          // we will use it to show the speed level of the motor by LEDs.
  else
    max_difference = right_difference;
  if (max_difference < left_difference)
    max_difference = left_difference;           // store the max difference in the variable max_difference

  int numLEDSLit = max_difference / 100;       // During the calibrating process, we find that the max_difference will change from
```

Takes reading of ambient light; uses this as a “baseline” for sensing

Maps data from sensors to range of LED display

EE_3_Final_Codes_With_Comments

```
int numLEDSLit = max_difference / 100;      // During the calibrating process, we find that the max_difference will change from
                                            // 0 to 830+ according to how far we put the light source from the photoresistor and
                                            // we have 8 LEDs. Therefore, we simply divided it by 100 to know how many LEDs should
                                            // be turned on to show the current speed level.

if (numLEDSLit > 8) numLEDSLit = 8;         // to constrain the number of turned on LED that since we only have 8 LEDs.
leds = 0;                                   // no LEDs lit to start
for (int i = 0; i < numLEDSLit; i++)
   leds = leds + (1 << i);
updateShiftRegister();
```

-Increases “numLEDSlit” relative to the sensor readings

-“updateShiftRegister()” refreshes the register of the IC with current byte

```
if (dynamic_sensor_front_Value > (initial_sensor_front_Value + 80))        // if the front photoresistor systerm detects there is significant light intensity change
                                                                           // at the front of the car. We use 80 here because we need to set a light noise range to
                                                                           // make sure the car won't move forward if this is subtle light change in the enviroment
                                                                           // (like the running fan on the ceiling, which might block the light in every turns.)
{

  front_difference = dynamic_sensor_front_Value - initial_sensor_front_Value;
  front_difference += 700;                                                 // Since the difference will start from 0 to some particular number like 800 according to the
                                                                           // light intensity detected by the photoresistor. Meanwhile, the motor operating voltage is around 4.5 V
                                                                           // and we use a 9V battery to drive the motor. Thus We need to increase the difference manually in order to
                                                                           // get the proper voltage, which should at least be 4.5 V not 0 V to run the motor. "700" here is the calibrated
                                                                           // value for our motors.

  constrain (front_difference,0,1023);                                     // Since we add the digital signal with the amount 700 in the last step, the difference might be greater than 1023.
                                                                           // We should constrain it in the range (0,1023).

  left_difference = front_difference;                                      // make the left and right wheels have the same "raw" digital signal to make the car move forward.
  right_difference = front_difference;

  left_value = map((left_difference ), 0, 1024, 0, 255);                   // change the digital signal to the voltage range signal to prepare to output the voltage by I/O pin.
  right_value = map((right_difference ), 0, 1024, 0, 255);

  analogWrite(output_right, right_value);                                  // output the voltage by the pin that control the left motor's transistor. The code is tricky here because
                                                                           // we use the varaible name "output_right" to control the left motor. You can think it means turn right and
                                                                           // to turn right, we should make the left wheels run.
  delayMicroseconds(2000);                                                 // This is just out calibration since our left motor react fast and stronger than the right motor. After
                                                                           // calibratiing, we find that delay this amout of time to the left motor can make the car move straight.
  analogWrite(output_left, left_value);                                    // output the voltage by the pin that control the right motor's transistor.

}

else                                                                       // if the front photoresistor does not detect strong light intensity difference
```

EE_3_Final_Codes_With_Comments

```
  analogWrite(output_right, right_value);        // output the voltage by the pin that control the left motor's transistor. The code is tricky here because
                                                 // we use the varaible name "output_right" to control the left motor. You can think it means turn right and
                                                 // to turn right, we should make the left wheels run.
  delayMicroseconds(2000);                       // This is just out calibration since our left motor react fast and stronger than the right motor. After
                                                 // calibratiing, we find that delay this amout of time to the left motor can make the car move straight.
  analogWrite(output_left, left_value);          // output the voltage by the pin that control the right motor's transistor.

}

else                                             // if the front photoresistor does not detect strong light intensity difference
{

if (dynamic_sensor_right_Value > initial_sensor_right_Value + 80)   // then if the right photoresistor detects significant light intensity difference
                                                 // the following codes are pretty the same reason as the above part.
{
  right_difference = dynamic_sensor_right_Value - initial_sensor_right_Value;
  right_difference += 500;
  constrain (right_difference,0,1023);
  right_value = map(right_difference, 0, 1024, 0, 255);
  analogWrite(output_right, right_value);

}
else
{
    analogWrite(output_right, 0);
}

// left photo resister
if (dynamic_sensor_left_Value > initial_sensor_left_Value + 100)
{
  left_difference = dynamic_sensor_left_Value - initial_sensor_left_Value;
  left_difference += 400;
  constrain (left_difference,0,1023);
  left_value = map(left_difference, 0, 1024, 0, 255);
  analogWrite(output_left, left_value);

}
else
{
  analogWrite(output_left, 0);
}

}
}
```

Main turning control-system

Project Results

- Stuff That Didn’t Work
  - Direct power of whole drive train w/ 9V
  - Direct-drive system (without calibration)

- Stuff that DID
  - Using a 9V in a separate circuit operated with a transistor
  - Calibration code and L/R motor adjustment

Conclusion

- What we learned
  - Transistors break easily when you don’t read data sheets...
  - DC Motors have non-negligible internal resistance
  - Programming shift-register IC and its 8 outputs
  - Programming for initial calibration
- Future applications/additions
  - Code to make photoresistors more responsive to light at a distance
  - Remote control system?

Project Demo