Arduino + senzor de culoare cu LED RGB

Cum se construieste un senzor de culoare ?

In acest tutorial vei descoperi cum se poate realiza un senzor de culoare folosind o celula LDR (Light Dependent Resistor) si un LED RGB (Red, Green, Blue). Pe langa cele doua componente vei mai avea nevoie de o placa Arduino, un breadboard, 2 rezistoare si cateva fire de conexiune.

Rolul senzorului de culoare este sa iti ofere informatii cu privire la culoarea unui obiect. Sketch-ul Arduino de mai jos iti ofera 3 valori corespunzatoare celor 3 culori: rosu, verde si albastru.

Acceseaza link-urile de mai jos pentru a citi mai multe informatii despre celulele LDR:

http://en.wikipedia.org/wiki/Photoresistor

http://www.resistorguide.com/photoresistor/

http://www.societyofrobots.com/schematics_photoresistor.shtml

http://www.instructables.com/id/Using-an-RGB-LED-to-Detect-Colours/

Celula LDR se conecteaza prin intermediul a 3 fire de conexiune la placa Arduino, pinii 5V, GND si A0 iar LED-ul RGB se conecteaza prin intermediul celor 4 fire la pinii GND, D2, D3 si D4. Senzorul de culoare utilizat in acest tutorial se poate construi dupa schema de mai jos.

Unde se poate utiliza senzorul ?

- Masina de sortare pentru Skittles M&M's:

  http://beta.ivc.no/wiki/index.php/Skittles_M%26M%27s_Sorting_Machine

- Detector de culoare:

  https://www.youtube.com/watch?v=RHO3bPcbysM

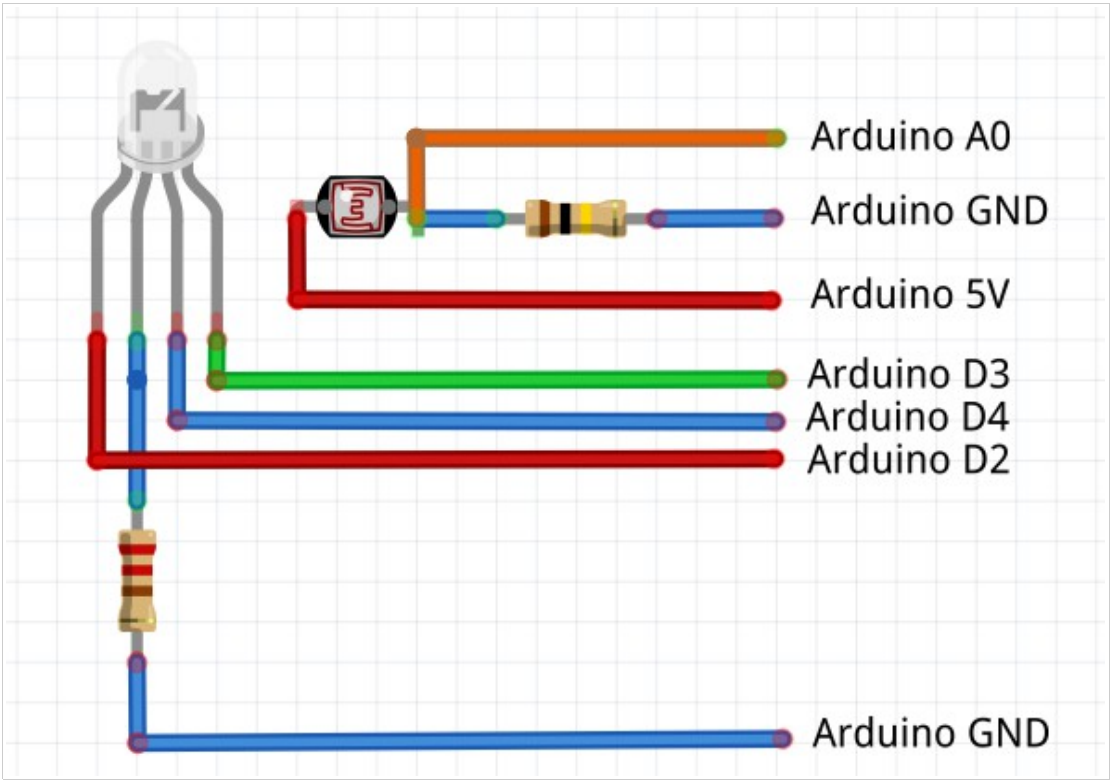
- O alta masina de sortare:

  https://www.youtube.com/watch?v=M_wsz6zmyZo

- Robot detector de culoare:

  https://decibel.ni.com/content/docs/DOC-35927

Schema de conectare



Sketch-ul Arduino

Sketch-ul de mai jos se incarca in placa Arduino iar valorile corespunzatoare celor 3 culori se citesc prin intermediul Monitorului Serial.

Pentru a detecta corect culoarea, senzorul necesita in prealabil o calibrare. Citeste explicatiile din codul sursa pentru a efectua corect calibrarea.

```
// Define colour sensor LED pins
int ledArray[] = {2, 3, 4};

// boolean to know if the balance has been set
boolean balanceSet = false;

//place holders for colour detected
int red = 0;
int green = 0;
int blue = 0;

//floats to hold colour arrays
float colourArray[] = {0, 0, 0};
float whiteArray[] = {0, 0, 0};
float blackArray[] = {0, 0, 0};


//place holder for average
int avgRead;

void setup() {

  //setup the outputs for the colour sensor
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);

  //begin serial communication
  Serial.begin(9600);
```

```
}

void loop() {

  checkBalance();
  checkColour();
  printColour();
}

void checkBalance() {
  //check if the balance has been set, if not, set it
  if (balanceSet == false) {
    setBalance();
  }
}

void setBalance() {
  //set white balance
  delay(5000);                                  //delay for five
seconds, this gives us time to get a white sample in front of our
sensor
  //scan the white sample.
  //go through each light, get a reading, set the base reading for
each colour red, green, and blue to the white array
  for (int i = 0; i <= 2; i++) {
    digitalWrite(ledArray[i], HIGH);
    delay(100);
    getReading(5);          //number is the number of scans to take
for average, this whole function is redundant, one reading works
just as well.
    whiteArray[i] = avgRead;
    digitalWrite(ledArray[i], LOW);
    delay(100);
  }
  //done scanning white, now it will pulse blue to tell you that it
is time for the black (or grey) sample.
  //set black balance
  delay(5000);              //wait for five seconds so we can
position our black sample
  //go ahead and scan, sets the colour values for red, green, and
blue when exposed to black
  for (int i = 0; i <= 2; i++) {
    digitalWrite(ledArray[i], HIGH);
```

```
    delay(100);
    getReading(5);
    blackArray[i] = avgRead;
    //blackArray[i] = analogRead(2);
    digitalWrite(ledArray[i], LOW);
    delay(100);
  }
  //set boolean value so we know that balance is set
  balanceSet = true;
  //delay another 5 seconds to allow the human to catch up to what
is going on
  delay(5000);
}

void checkColour() {
  for (int i = 0; i <= 2; i++) {
    digitalWrite(ledArray[i], HIGH); //turn or the LED, red, green
or blue depending which iteration
    delay(100);                      //delay to allow CdS to
stabalize, they are slow
    getReading(5);                  //take a reading however many
times
    colourArray[i] = avgRead;        //set the current colour in
the array to the average reading
    float greyDiff = whiteArray[i] - blackArray[i];
//the highest possible return minus the lowest returns the area for
values in between
    colourArray[i] = (colourArray[i] - blackArray[i]) / (greyDiff)
* 255; //the reading returned minus the lowest value divided by the
possible range multiplied by 255 will give us a value roughly
between 0-255 representing the value for the current
reflectivity(for the colour it is exposed to) of what is being
scanned
    digitalWrite(ledArray[i], LOW);  //turn off the current LED
    delay(100);
  }
}
void getReading(int times) {
  int reading;
```

```
  int tally = 0;
  //take the reading however many times was requested and add them
up
  for (int i = 0; i < times; i++) {
    reading = analogRead(0);
    tally = reading + tally;
    delay(10);
  }
  //calculate the average and set it
  avgRead = (tally) / times;
}

//prints the colour in the colour array, in the next step, we will
send this to processing to see how good the sensor works.
void printColour() {
  Serial.print("R = ");
  Serial.println(int(colourArray[0]));
  Serial.print("G = ");
  Serial.println(int(colourArray[1]));
  Serial.print("B = ");
  Serial.println(int(colourArray[2]));
  //delay(2000);
}
```