UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET
ODSJEK ZA AUTOMATIKU I ELEKTRONIKU
MEHATRONIKA
AKADEMSKA 2019/2020. GODINA

Razvrstavanje objekata po boji

- Izvještaj projektnog zadatka -

Članovi tima:
Muminović Samira, 17582
Neković Ajla, 17418

Sarajevo, septembar 2019.

Sadržaj

1. Uvod

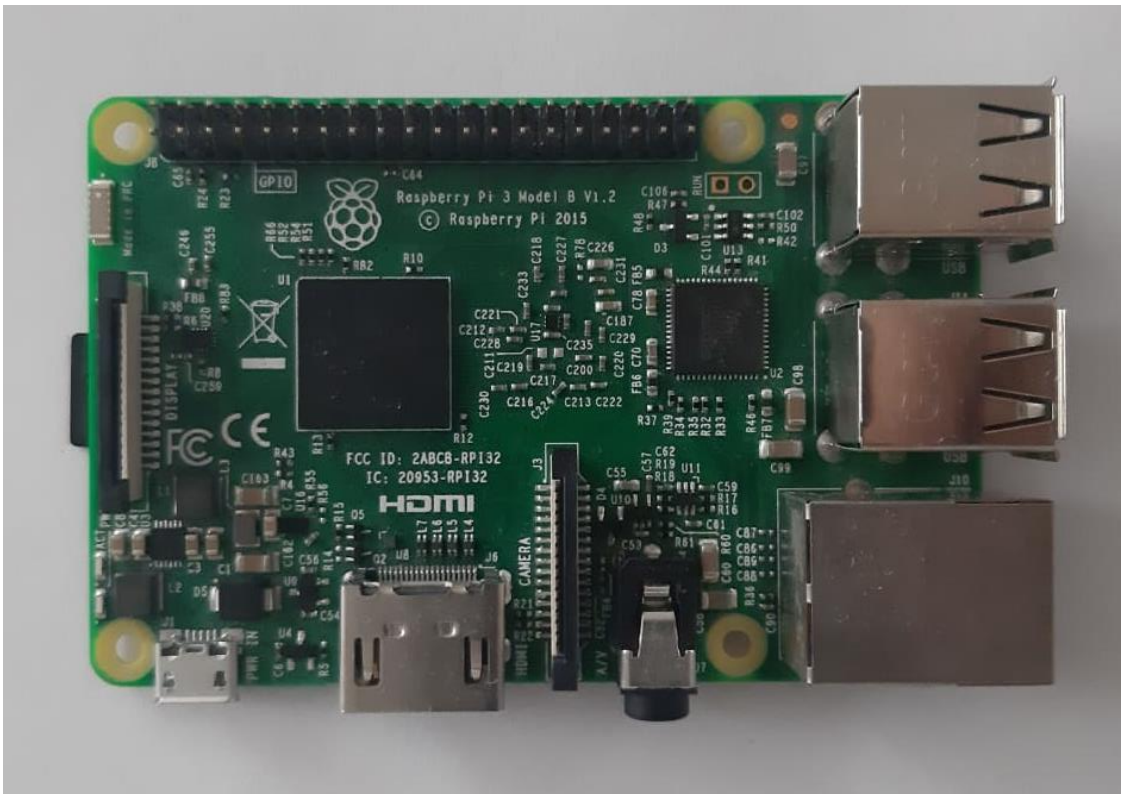
1.1. Opis problema

Tema projekta je bila kreiranje i realiziranje sistema za razvrstavanje objekata uz pomoć mikrokontrolera Raspberry Pi 3B, kamere Raspberry Pi v2, step motora i računara. Kameru ćemo spojiti na mikrokontroler, a koračne motore ćemo pričvrstiti na plastičnu konstrukciju. U zavisnosti od slike dobivene korištenjem kamere, razvrstavanje objekata će se vršiti po boji u tri skupine: crvena, žuta i plava boja. Za upravljanje mikrokontrolerom koristit ćemo razvojno okruženje „Thonny IDE“ (*skr. od eng. Thonny integrated development environment*), koji predstavlja okruženje za jezik „Python“. U našem slučaju razvrstavali smo bombone.

Postavljeni zadatak se mogao dizajnirati i realizirati na mnogo načina te smo se morali odlučiti za funkcionalnosti koje će naš sistem podržavati. Iz ovog razloga smo precizirali na koji će način naš sistem djelovati. U skladu s tim na konstrukciji, koju smo napravili, pričvrstili smo dva step motora. Prvi motor služi za uzimanje samo jedne bombone kako bi se izvršila provjera boje, koju vršimo analiziranjem, kamerom uslikane, fotografije. Kada odredimo boju objekta drugi koračni motor pomjeranjem jednog dijela konstrukcije omogućava razvrstavanje bombone u ispravnu skupinu. Sistem smo programirali u jeziku *Python* razvojnog okruženja *Thonny IDE*.

1.2. Raspberry Pi 3B

Raspberry Pi je mikrokontroler veličine kreditne kartice koji je razvila u Ujedinjenom Kraljevstvu fondacija Raspberry Pi sa namjerom da promoviše i poduči osnove informatike u školama i zemljama u razvoju. Orginalni modeli Raspberry Pi i Raspberry Pi 2 su proizvedeni u više različitih konfiguracija preko licencnih sporazuma za proizvodnju, međutim hardver je isti kod svih proizvođača. Nekoliko generacija Raspberry Pi modela je već izašlo na tržište. Prva generacija (Pi 1) izašla je u februaru 2012. godine kao osnovni model A i specifičniji model B. Modeli A+ i B+ izašli su godinu dana kasnije. Raspberry Pi 2B izbačen je u februaru 2015, a Raspberry Pi 3 model B u februaru 2016. godine. Vrijednost ovih ploča je između 20 i 35$. Pi Zero sa manjom štampanom pločom i ograničenim ulazno - izlaznim mogućnostima (*GPIO, skr. od eng. General Purpose I/O*) izašao je u novembru 2015. godine. Svi modeli imaju Broadcom sistem čip koji podrazumijeva ARM kompatibilni CPU *(skr. od eng. central processing unit, procesor)* i ugrađeni čip GPU (*skr. od eng. graphics processing unit, grafički procesor*).



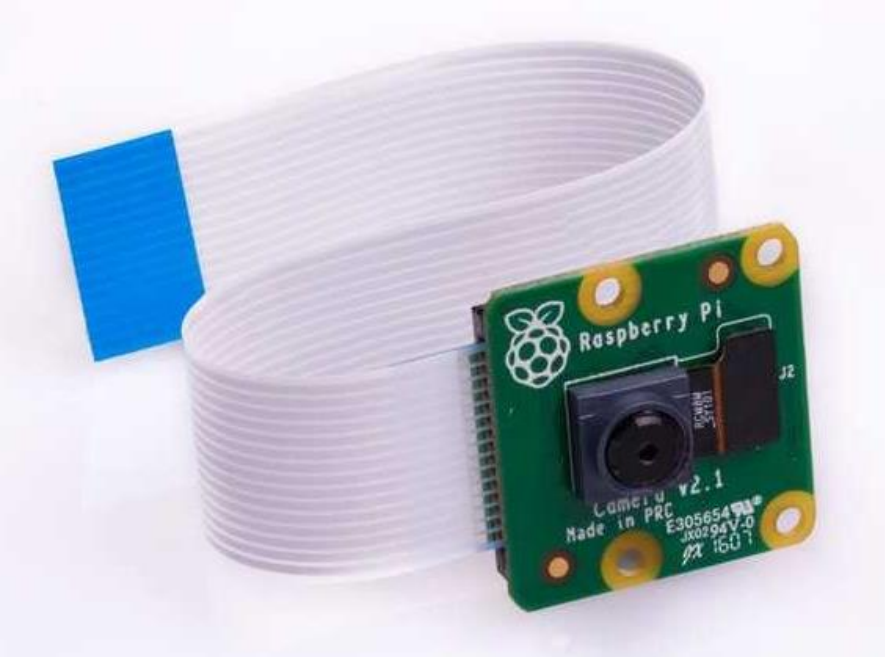
Slika 1.1. Model Raspberry Pi 3B

SD kartice (*skr. od eng. Secure Digital Card*) se koriste za čuvanje operativnog sistema i programske memorije u SDHC (*skr. od eng. SD High Capacity*) ili MicroSDHC veličinama. Većina ploča ima između jednog i četiri slota, zatim HDMI (*skr. od eng. High-Definition Multimedia Interface, multimedijalni sklop visoke definicije*) i video izlaz, kao i 3.5 mm audio džek. Low-level izlaz posjeduje GPIO pinove koji podržavaju zajedničke protokole kao što je I^2C (*skr. od eng. Inter-Integrated Circuit, komunikacija*). Neki modeli imaju i RJ45 Ethernet port.

CPU našeg modela, Pi 3B (prikazanog na slici 1.1), dostiže brzinu od 700 MHz do 1.2 GHz, sa memorijom ploče u opsegu od 256 MB do 1 GB RAM. Za skladištenje koristi microSDHC slot kao i njegovi prethodni. Snaga za napajanje ovog modela iznosi 4.0 W. Grafiku predstavlja Broadcom VideoCore IV ali na većim frekvencijama nego prethodni modeli koji su radili na 250 MHz. Pored ostalih portova model Pi 3B na ploči ima i WiFi 802.11n te Bluetooth ulaz.

1.3. Raspberry Pi camera v2

Raspberry Pi modul kamere v2 (slika 1.2) zamijenio je izvorni modul kamere u aprilu 2016. godine. Model kamere v2 posjeduje Sony IMX219 senzor od 8 megapiksela (u usporedbi s 5-megapikselnim OmniVision OV5647 senzorom originalne kamere) i kao takav nudi veliki napredak u kvaliteti slike, a posebno intenzitetu boja i osvjetljenju. Može se koristiti za snimanje videozapisa visoke rezolucije, kao i fotografija.

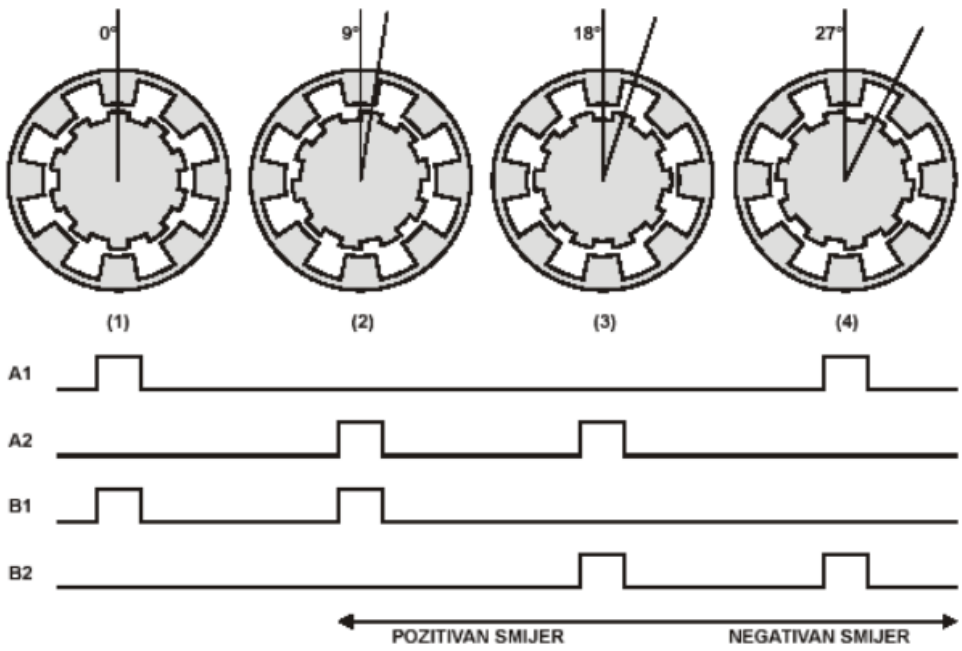


Slika 1.2. Raspberry Pi modul kamere v2.1

Kamera nudi veliki izbor funkcionalnosti, ali je njeno korištenje vrlo jednostavno. Za kreiranje efekata koriste se biblioteke koje se povezuju sa kamerom. Podržava 1080p30, 720p60 i VGA90 video mod, kao i snimanje. Priključuje se preko 15 cm vrpce kabla na CSI port (*skr. od eng. Camera Serial Interface, serijski interfejs kamere*) na Raspberry Pi. Kamera radi sa svim modelima Raspberry Pi 1, 2, 3 i 4. i za nju su izgrađene brojne biblioteke proizvođača, uključujući i biblioteku Picamera Python. Ova kamera je vrlo popularna u kućnim sigurnosnim aplikacijama. Dimenzije su joj $25mm\ x\ 23mm\ x\ 9mm$, a težina nešto više od 3 g.

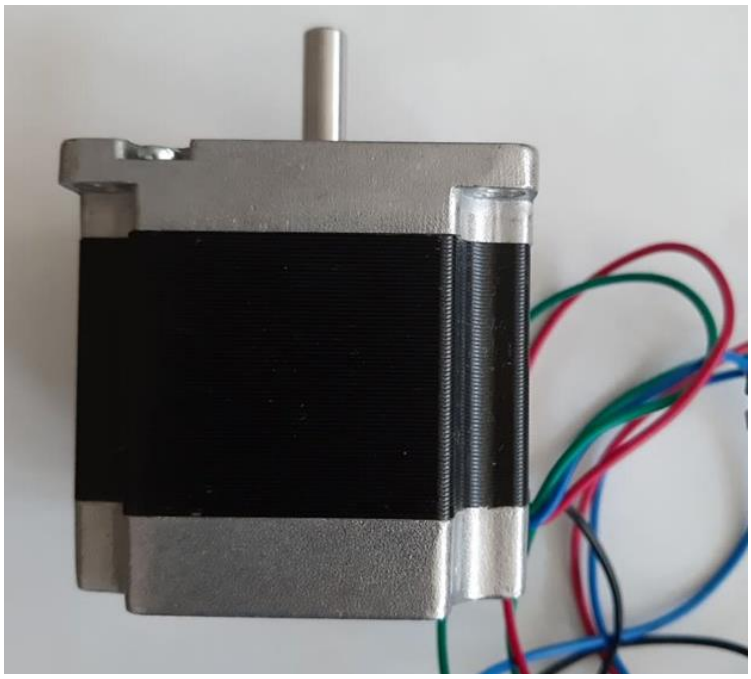
1.4. Step motor

Step motor je vrsta elektromotora bez četkica koji pretvara digitalne impulse struje u fiksne inkremente ugaonog pomjeranja nazvane koraci. Ova vrsta motora obezbjeđuje precizno pozicioniranje tereta, a kontrola motora se vrši direktno računarom, mikrokontrolerom ili programibilnim logičkim kontrolerom. Zbog svoje konstrukcije bez četkica, koračni motori su pouzdani, izdržljivi, i ne zahtijevaju nikakvo održavanje. Na slici 1.3. prikazani su impulsi za upravljanje step motorom.



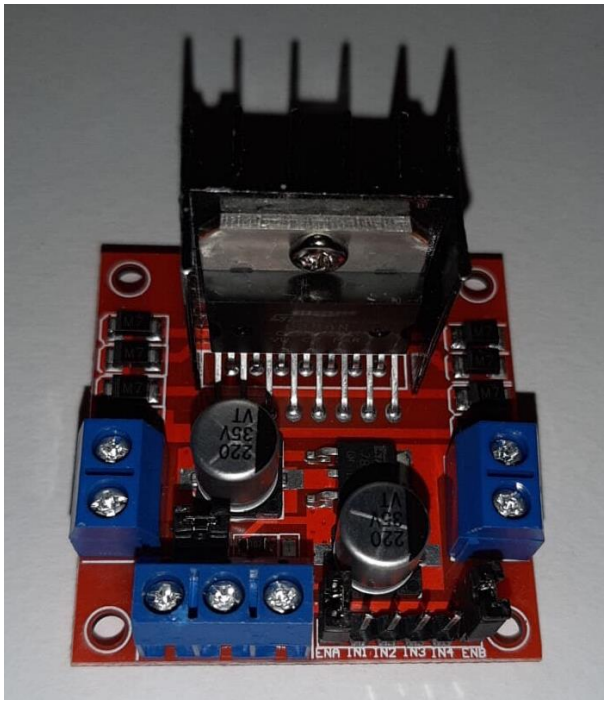
Slika 1.3. Impulsi za upravljanje step motorom

Za motor koji smo mi koristi u ovom projektu potrebna je struja jačine 2.0 *A*, a jednom koraku odgovara 1.8° (stepeni). Model korištenog motora možemo vidjeti na slici 1.4.



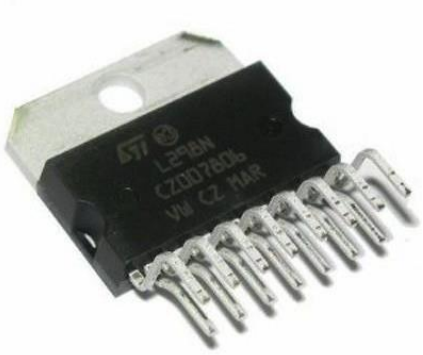
Slika 1.4. Step motor

Osim svega navedenog korišten je i H-most L298N koji se nalazi, kao dio sistema, na pločici prikazanoj na sljedećoj slici (slika 1.5). Koristili smo dva takva sistema. H-most je elektronsko kolo koje omogućuje kontrolu smjera struje kroz jednosmjerni elektromotor ili kroz neki drugi potrošač. Promjenom smjera struje kroz armaturu jednosmjernog motora, mijenja se smjer rotacije njegove osovine. H-most se često primjenjuje u elektronskim kolima, izveden kao integrisano kolo ili sa diskretnim tranzistorima. Može se izvesti i sa relejima ili običnim prekidačima, ako nije potrebna velika brzina prekapčanja.



Slika 1.5. H-most za upravljanje step motorom

Izgled samog, korištenog, modula predstavljen je na slici 1.6.



Slika 1.6. Modul L298N

2. Realizacija projekta

2.1. Svođenje na formu algoritma

Kako bismo izvršili željeni zadatak bilo je potrebno napisati kod za mikrokontroler koji upravlja kretanjem prvog motora, aktiviranjem kamere, obradom fotografije koju dobivamo kao povratnu informaciju sa kamere, aktiviranjem drugog motora za pomjeranje konstrukcije te odvajanje bombone u ispravnu skupinu.

Sa druge strane bilo je neophodno napisati kod koji će obaviti obradu fotografije, tj. odrediti boju objekta. Na osnovu tog podatka poznato je kojoj skupini pripada analizirana bombona. Iz navedenog se vidi da se u kodu mora izvršiti upravljanje drugog motora kako bi konstrukcija propustila bombonu na željeno mjesto.

Prvi korak u realizaciji projekta predstavljala je postavka samog Raspberry-ja i SD kartice. Postavka Raspberry-ja je zahtijevala USB kabl, microSD karticu (najmanje 8 GB), kompjutersku tastaturu, kompjuterski miš i televizor ili monitor sa HDMI portom. Kako bi se mikrokontroler mogao koristiti prethodno smo trebali instalirati operativni sistem Raspbian na microSD karticu. Uputstva za instalaciju našli smo na oficijalnoj Raspberry Pi web stranici (link stranice). Ukratko opisano tačke uputstva su bile: preuzimanje NOOBS-a (pomoćni dokument u obliku .zip foldera), formatiranje SD kartice, ekstraktovanje pomenutog foldera NOOBS te kopiranje fajlova iz foldera na pokazano mjesto na SD kartici. Nakon uspješno instaliranog operativnog sistema microSD karticu ubacimo u Raspberry na koji spojimo miš i tastaturu, a onda uz pomoć HDMI kabla spojimo na TV ili monitor. Na kraju u mikrokontroler ubacimo USB kabl, koji se ponaša kao napajanje i prouzrokuje blinkanje crvene LED. Nakon paljenja Raspberry-ja na povezanom TV-u ili monitoru otvora se Raspbian Desktop, tj. željena aplikacija.

2.2. Korišteni alat

Programski alati koji su korišteni u cilju realizacije projektnog zadatka su sljedeći:

- ✓ Python
- ✓ Thonny IDE
- ✓ Raspbian

2.2.1. Python

Pronalazači Raspberry-ja promovišu Python kao glavni programski jezik. Python je interpretirani programski jezik visokog nivoa opće namjene, napravljen od strane Guido van Rossum-a 1990. godine, a nosi naziv po televizijskoj seriji Monty Python's Flying Circus. Po automatskoj memorijskoj alokaciji, sličan je programskim jezicima kao što su Perl, Ruby, Smalltalk, itd. Python dopušta programerima korištenje sljedećih stilova programiranja: objektno orijentirano, strukturno i aspektno orijentirano programiranje. Ova fleksibilnost čini Python sve popularnijim programskim jezikom. Najviše se koristi na Linuxu, ali postoje verzije i za druge operacijske sisteme. Logo pomenutog programskog jezika prikazan je na slici 2.1.

Slika 2.1. Logo programskog jezika Python

Glavna filozofija jezika sažeta je u dokumentu *The Zen of Python* (*PEP 20*), koji uključuje aforizme poput:

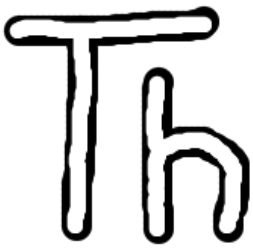
- ✓ *Lijepo je bolje nego ružno.*
- ✓ *Eksplicitno je bolje od implicitnog.*
- ✓ *Jednostavno je bolje od složenog.*
- ✓ *Složeno je bolje od komplikovanog.*
- ✓ *Čitljivost se računa.*

2.2.2. Thonny IDE

Thonny je integrirano razvojno okruženje Pythona. Podržava različite načine prolaska kroz kod, postupno ocjenjivanje izraza, detaljnu vizualizaciju skupa poziva i način objašnjavanja pojmova referenci. Program radi na Windows-u, macOS-u i Linux-u. Dostupan je u obliku binarnog paketa, uključujući nedavni tumač Python ili u paketu instaliran na Pip. Može se instalirati putem paketa operacijskog sistema na Debianu, Raspberry Pi, Ubuntu i Fedora. Logo je prikazan na slici 2.2.

Prednosti:

- ✓ broj linija,
- ✓ prolazak kroz kod bez prekida,
- ✓ praćenje promjene vrijednosti varijabli prilikom uklanjanja grešaka,
- ✓ zamjena izraza njihovim vrijednostima,
- ✓ odvojeni dijelovi za izvršavanje poziva funkcije,
- ✓ varijable i memorija mogu se objasniti ili pojednostavljenim modelom (*ime → vrijednost*) ili realnijim modelom (*ime → adresa / id → vrijednost*).



Slika 2.2. Logo programskog jezika Python

2.2.3. Raspbian

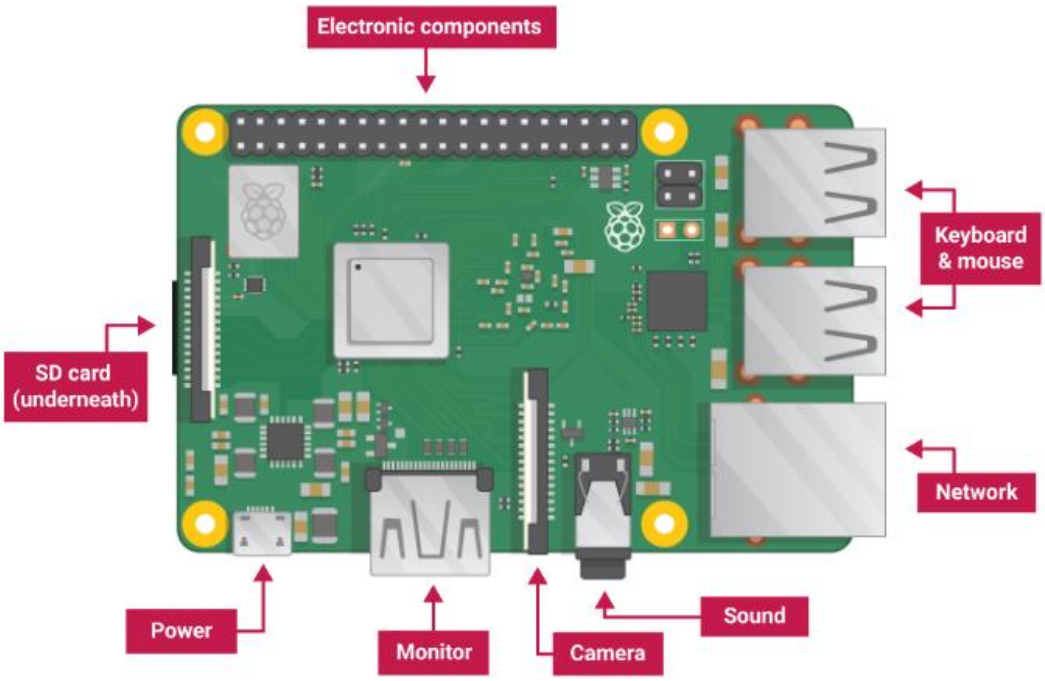
Raspbian je računarski operativni sistem za Raspberry Pi zasnovan na Debianu, operativnom sistemu istoimene organizacije posvećene razvoju i promicanju ideala slobodnog softvera. Slika 2.3. predstavlja Raspian kao spoj Raspberry-ja i Debiana, kao što je objašnjeno u prethodnoj rečenici. Postoji nekoliko verzija Raspbian-a, uključujući Raspbian Buster i Raspbian Stretch. Ovaj operativni sistem je od 2015. godine službeno osiguran od strane Raspberry Pi fondacije kao primarni operativni sistem porodice Raspberry Pi jednopunskih računara. Raspbian su formirali Mike Thompson i Peter Green kao nezavisni projekat, a početna gradnja je završena u junu 2012. godine. Raspbian je visoko optimiziran za ARM CPU-ove niske performanse kompanije Raspberry Pi i idalje je u aktivnom razvoju.

Raspbian koristi PIXEL (*skr. od eng.* ***P****i* ***I****mproved* ***X****-Window* ***E****nvironment,* ***L****ightweight*), kao svoje glavno radno okruženje od posljednjeg ažuriranja. Sastoji se od modificiranog LXDE (*skr. od eng.* ***L****ightweight* ***X****11* ***D****esktop* ***E****nvironment*) radnog okruženja i Openbox upravitelja slaganja prozora. Distribucija se isporučuje s kopijom programa računalne algebre Mathematica i verzije Minecrafta nazvanom Minecraft Pi, kao i laganom verzijom Chromium-a od posljednje verzije.

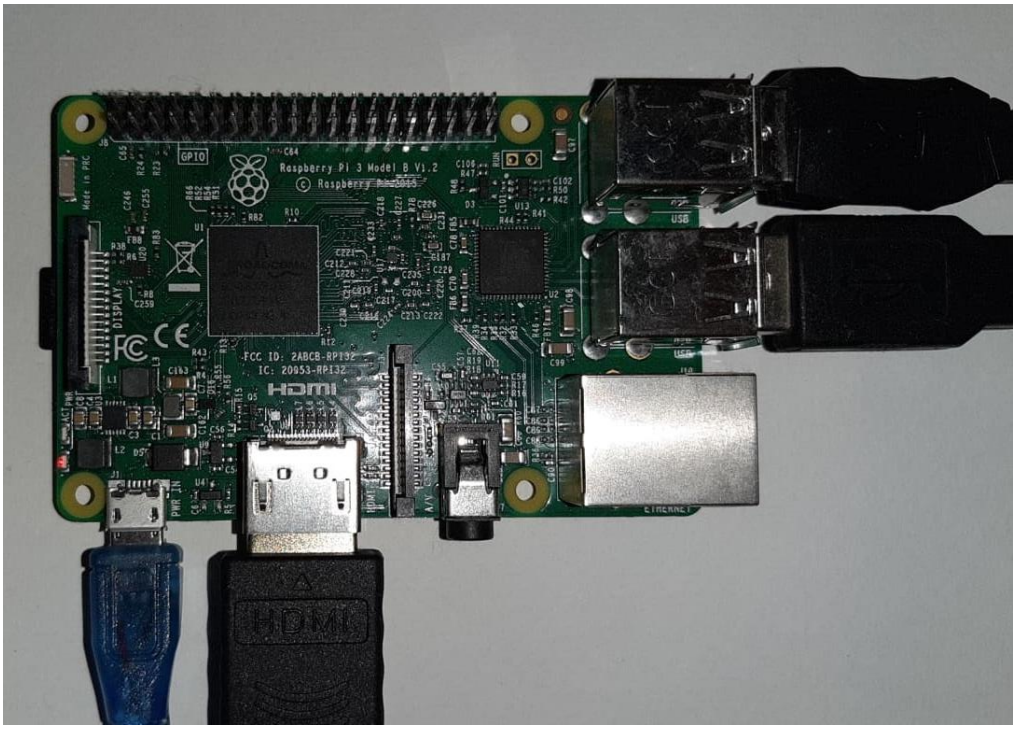
Slika 2.3. Spoj nastanka Raspbian-a

2.3. Shema spajanja

Kao što je prethodno navedeno postavka Raspberry-ja zahtijevala je korištenje microSD kartice, spajanje USB kabla, kompjuterske tastature, kompjuterskog miša i televizora ili monitora. Na slici 2.4. prikazana je shema spajanja svakog od mogućih potrebnih dijelova na mikrokontroleru, dok je na slici 2.5. predstavljeno na koji način smo mi spojili model Raspberry Pi 3B kako bi ga postavili.

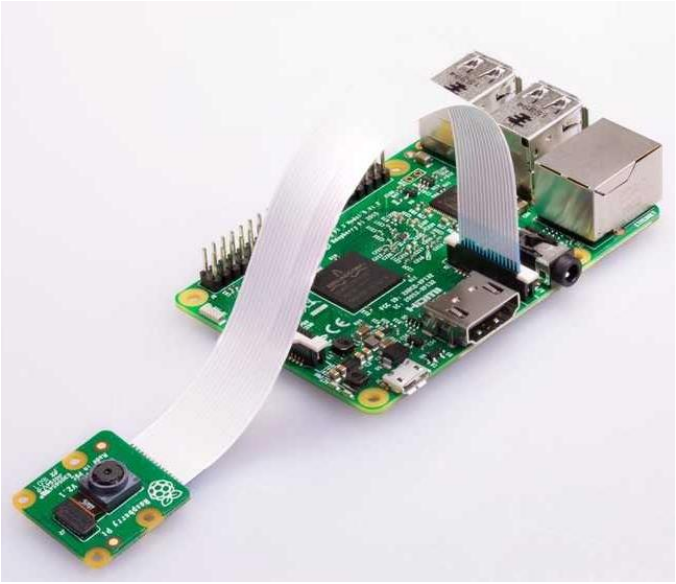


Slika 2.4. Mjesta spajanja ploče i potrebnih komponenti



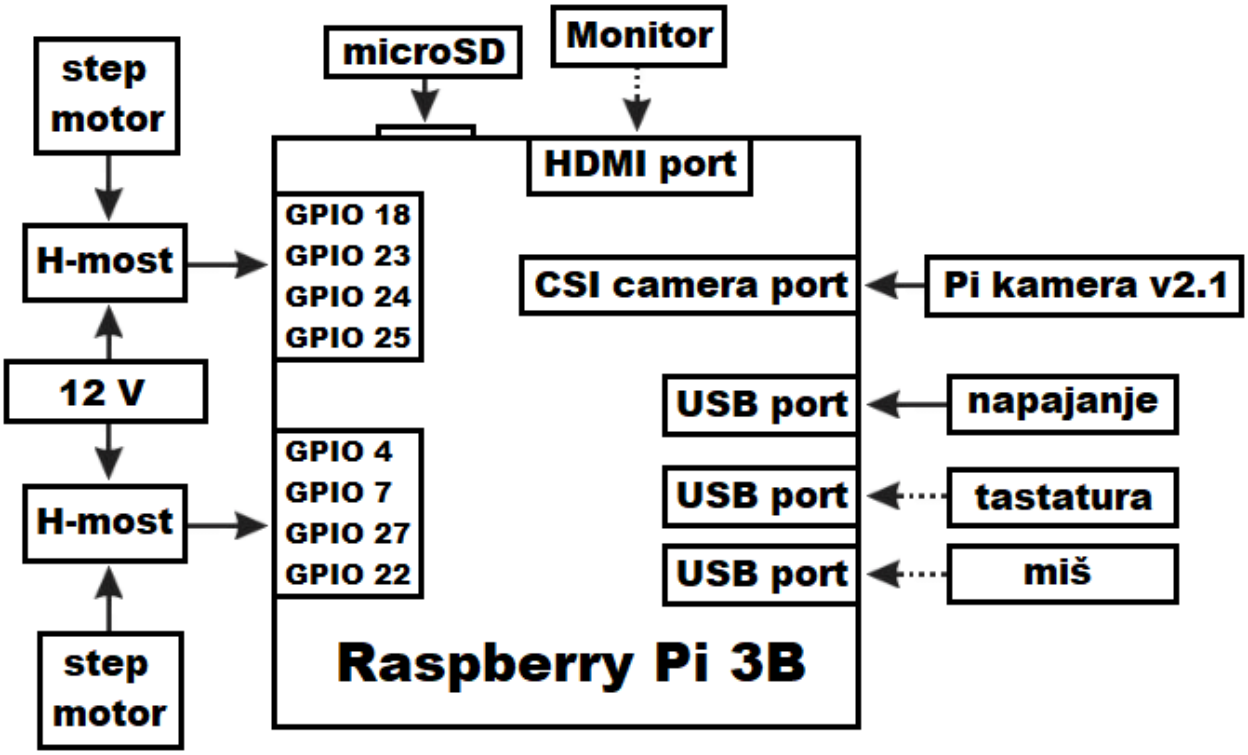
Slika 2.5. Raspberry spojen za postavku

Pored navedenih dijelova potrebnih za postavku, za realizaciju projekta bilo je potrebno i spajanje kamere na mikrokontroler. Spajanje kamere na Raspberry Pi 3B pokazano je na slici 2.6.



Slika 2.6. Način spajanja Raspberry Pi 3B i kamere v2.1

Model Raspberry Pi 3B spojen je na sljedeći način: napajanje je spojeno na USB port, monitor na HDMI port, kamera na CSI port i H-mostovi na neke od ponuđenih pinova. Pinovi koje smo mi odabrali za H-mostove su GPIO 18 i GPIO 23. H-mostovima smo upravljali dva step motora. Shema spajanja prikazana je na slici 2.7.



Slika 2.7. Šematski prikaz spajanja komponenti

Na slici 2.8. prikazana je shema cjelokupnog sistema, spojena u laboratoriji.

Slika 2.8. Sistem spojen u laboratoriji

3. Programska implementacija

Kako smo implementirali zadatak u Pythonu prikazano je u tabeli 3.1.

```
from __future__ import print_function
from time import sleep
from picamera import PiCamera #IMPORTI POTREBNI KAMERI DA USLIKA SLIKU

import RPi.GPIO as GPIO
import time #IMPORTI ZA MOTOR

import binascii
import struct
from PIL import Image
import numpy as np
import scipy
import scipy.misc
import scipy.cluster #IMPORTI POTREBNI DA SE PRONADJE DOMINANTNA BOJA

#POCETNE TRI BOJE (SVE POSTAVIMO NA NULA)
#rgb prve boje
r1 = 0
g1 = 0
b1 = 0
#rgb druge boje
r2 = 0
g2 = 0
b2 = 0
#rgb trece boje
r3 = 0
g3 = 0
b3 = 0

#POZICIJA DRUGOG MOTORA (ON IMA 3 POZICIJE - 3 KUTIJICE ZA BOBE)
pozicija = 1

#VARIJABLE ZA MOTOR
delay = 0.0055

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#PINOVI ZA UPRAVLJANJE MOTORA 1
coil_A_1_pin = 18
coil_A_2_pin = 23
coil_B_1_pin = 24
coil_B_2_pin = 25
```

```
#*******************************************
#PINOVI ZA UPRAVLJANJE MOTORA 2
faza1 = 4
faza2 = 17
faza3 = 27
faza4 = 22

#POSTAVLJANJE PINOVA MOTORA 2 KAO IZLAZNI
GPIO.setup(faza1, GPIO.OUT)
GPIO.setup(faza2, GPIO.OUT)
GPIO.setup(faza3, GPIO.OUT)
GPIO.setup(faza4, GPIO.OUT)

#FUNKCIJA MOTORA 2
def setStep2(w1, w2, w3, w4):
  GPIO.output(faza1, w1)
  GPIO.output(faza2, w2)
  GPIO.output(faza3, w3)
  GPIO.output(faza4, w4)

#*****************************************************************

setStep2(0,0,0,0)

#POSTAVLJANJE PINOVA MOTORA 1 KAO IZLAZNI
GPIO.setup(coil_A_1_pin, GPIO.OUT)
GPIO.setup(coil_A_2_pin, GPIO.OUT)
GPIO.setup(coil_B_1_pin, GPIO.OUT)
GPIO.setup(coil_B_2_pin, GPIO.OUT)


#FUNKCIJA ZA MOTOR 1
def setStep(w1, w2, w3, w4):
  GPIO.output(coil_A_1_pin, w1)
  GPIO.output(coil_A_2_pin, w2)
  GPIO.output(coil_B_1_pin, w3)
  GPIO.output(coil_B_2_pin, w4)

#SVE PINOVE MOTORA 1 VRATI NA LOW
setStep(0,0,0,0)

for k in range(0,10):

    #MOTOR 1 DOVODI BOBU SA POCETNE TACKE DO KAMERE
    for i in range(0, 12): #UNAPRIJED
        setStep(1,0,1,0)
        time.sleep(delay)
        setStep(0,1,1,0)
        time.sleep(delay)
```

```
        setStep(0,1,0,1)
        time.sleep(delay)
        setStep(1,0,0,1)
        time.sleep(delay)

    #SVE PINOVE MOTORA 1 VRATI NA LOW
    setStep(0,0,0,0)

    #KAMERA SLIKA BOBU
    camera = PiCamera()
    camera.resolution = (1024, 768)
    camera.start_preview()
    sleep(2)
    camera.capture('slika.jpg')
    camera.close()

    #ANALIZIRA SE SLIKA, I DOMINANTNA BOJA SE STAVI U VARIJABLU PEAK
    NUM_CLUSTERS = 5
    print('reading image')
    im = Image.open('slika.jpg')
    im = im.resize((150, 150))
    ar = np.asarray(im)
    shape = ar.shape
    ar = ar.reshape(scipy.product(shape[:2]), shape[2]).astype(float)
    print('finding clusters')
    codes, dist = scipy.cluster.vq.kmeans(ar, NUM_CLUSTERS)
    print('cluster centres:\n', codes)
    vecs, dist = scipy.cluster.vq.vq(ar, codes)
    counts, bins = scipy.histogram(vecs, len(codes))
    index_max = scipy.argmax(counts)
    peak = codes[index_max]
    colour = binascii.hexlify(bytearray(int(c) for c in
peak)).decode('ascii')
    print('most frequent is %s (#%s)' % (peak, colour))

    #RGB KOMPONENTE VARIJABLE PEAK
    rp=int(peak[0])
    gp=int(peak[1])
    bp=int(peak[2])

    nasao_boju1=False
    nasao_boju2=False
    nasao_boju3=False
```

```
 #*******UPOREDJIVANJE PRONADJENE BOJE SA 3 BOJE********

    #UPOREDJIVANJE SA PRVOM BOJOM
    if (rp-r1) == rp: #ovo znaci da je peak prva boja koja se pojavila, jer
je r1 == 0
        r1 = rp
        g1 = gp #sada je boja1 jednaka ovoj prvoj pronadjenoj boji
        b1 = bp
        nasao_boju1=True
    #UPOREDJIVANJE SA DRUGOM BOJOM
    elif (rp-r2) == rp: #ovo znaci da je peak prva boja koja se pojavila, jer
je r2 == 0
        r2 = rp
        g2 = gp #sada je boja2 jednaka ovoj pronadjenoj boji
        b2 = bp
        nasao_boju2=True
    #UPOREDJIVANJE SA TRECOM BOJOM
    elif (rp-r3) == rp: #ovo znaci da je peak prva boja koja se pojavila, jer
je r3 == 0
        r3 = rp
        g3 = gp #sada je boja3 jednaka ovoj pronadjenoj boji
        b3 = bp
        nasao_boju3=True

    if ((((abs(rp-r1)) <= 15) and ((abs(gp-g1)) <= 15) and ((abs(bp-b1)) <=
15)) or nasao_boju1==True): #ako su priblizno jednake, odstupanje npr manje
od 15 ILI ako je ovo prva boja
        nasao_boju1=False
        #BOBA IDE U PRVU KUTIJU
        if pozicija == 1:
        #MOTOR 1 dovede bobu do tobogana
            for i in range(0, 13): #UNAPRIJED
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)

        if pozicija == 2:
        #MOTOR 2 SE IZ POZICIJE 2 VRACA JEDNOM UNAZAD DA DODJE U POZICIJU 1
            for i in range(0, 3): #nazad
                setStep2(1,0,1,0)
                time.sleep(delay)
```

```
                setStep2(0,1,1,0)
                time.sleep(delay)
                setStep2(0,1,0,1)
                time.sleep(delay)
                setStep2(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 2 VRATI NA LOW
            setStep2(0,0,0,0)
            #motor 1 dovede bobu do tobogana
            for i in range(0, 13):
                setStep(1,0,1,0) #UNAPRIJED
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)
            pozicija = 1

        if pozicija == 3:
        #MOTOR 2 SE IZ POZICIJE 3 VRACA DVA PUTA UNAZAD DA DODJE U POZICIJU 1
            for i in range(0, 6): #nazad
                setStep2(1,0,1,0)
                time.sleep(delay)
                setStep2(0,1,1,0)
                time.sleep(delay)
                setStep2(0,1,0,1)
                time.sleep(delay)
                setStep2(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 2 VRATI NA LOW
            setStep2(0,0,0,0)
        #motor 1 dovede bobu do tobogana
            for i in range(0, 13):
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)
            pozicija = 1
```

```
    elif ((((abs(rp-r2)) <= 15) and ((abs(gp-g2)) <= 15) and ((abs(bp-b2)) <=
15)) or nasao_boju2==True): #ako su priblizno jednake, odstupanje npr manje
od 15
        nasao_boju2==False
        #BOBA IDE U DRUGU KUTIJU
        if pozicija == 2:
        #MOTOR 1 DOVEDE BOBU DO TOBOGANA
            for i in range(0, 13): #UNAPRIJED
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)

        if pozicija == 1:
        #MOTOR 2 IDE JEDNOM NAPRIJED DA IZ POZICIJE 1 DODJE U POZICIJU 2
            for i in range(0, 3):
                setStep2(1,0,0,1) #naprijed
                time.sleep(delay)
                setStep2(0,1,0,1)
                time.sleep(delay)
                setStep2(0,1,1,0)
                time.sleep(delay)
                setStep2(1,0,1,0)
                time.sleep(delay)
            #SVE PINOVE MOTORA 2 VRATI NA LOW
            setStep2(0,0,0,0)
        #motor 1 dovede bobu do tobogana
            for i in range(0, 13):
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)
            pozicija = 2
```

```
        if pozicija == 3:
        #MOTOR 2 IDE JEDNOM UNAZAD DA IZ POZICIJE 3 DODJE U POZICIJU 2
            for i in range(0, 3): #nazad
                setStep2(1,0,1,0)
                time.sleep(delay)
                setStep2(0,1,1,0)
                time.sleep(delay)
                setStep2(0,1,0,1)
                time.sleep(delay)
                setStep2(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 2 VRATI NA LOW
            setStep2(0,0,0,0)
        #motor 1 dovede bobu do tobogana
            for i in range(0, 13):
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)
            pozicija = 2

    elif ((((abs(rp-r3)) <= 15) and ((abs(gp-g3)) <= 15) and ((abs(bp-b3)) <=
15)) or nasao_boju3==True): #ako su priblizno jednake, odstupanje npr manje
od 15
        nasao_boju3==False
        #BOBA IDE U TRECU KUTIJU
        if pozicija == 3:
        #prvi motor bobu dovede do tobogana
            for i in range(0, 13): #UNAPRIJED
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)

        if pozicija == 1:
        #motor 2 ide dva puta naprijed
            for i in range(0, 6):
```

```
                setStep2(1,0,0,1) #naprijed
                time.sleep(delay)
                setStep2(0,1,0,1)
                time.sleep(delay)
                setStep2(0,1,1,0)
                time.sleep(delay)
                setStep2(1,0,1,0)
                time.sleep(delay)
            #SVE PINOVE MOTORA 2 VRATI NA LOW
            setStep2(0,0,0,0)
        #motor 1 dovede bobu do tobogana
            for i in range(0, 13): #UNAPRIJED
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)
            pozicija = 3

        if pozicija == 2:
        #motor 2 ide jednom naprijed
            for i in range(0, 3):
                setStep2(1,0,0,1) #naprijed
                time.sleep(delay)
                setStep2(0,1,0,1)
                time.sleep(delay)
                setStep2(0,1,1,0)
                time.sleep(delay)
                setStep2(1,0,1,0)
                time.sleep(delay)
            #SVE PINOVE MOTORA 2 VRATI NA LOW
            setStep2(0,0,0,0)
        #motor 1 dovede bobu do tobogana
            for i in range(0, 13):
                setStep(1,0,1,0)
                time.sleep(delay)
                setStep(0,1,1,0)
                time.sleep(delay)
                setStep(0,1,0,1)
                time.sleep(delay)
                setStep(1,0,0,1)
                time.sleep(delay)
```

```
            #SVE PINOVE MOTORA 1 VRATI NA LOW
            setStep(0,0,0,0)
            pozicija = 3

    else:
        print('greska')
    #motor 1 vratiti gdje je bio, da ode po drugu bobu
    for i in range(0, 25):
        setStep(1,0,0,1) #UNAZAD
        time.sleep(delay)
        setStep(0,1,0,1)
        time.sleep(delay)
        setStep(0,1,1,0)
        time.sleep(delay)
        setStep(1,0,1,0)
        time.sleep(delay)
    #SVE PINOVE MOTORA 1 VRATI NA LOW
    setStep(0,0,0,0)
```

Tabela 3.1. Korišteni kod

U kodu vidimo da smo objekte, tj. bombone razvrstavali na tri boje: crvena, žuta i plava. Razvrstavanje smo vršili pomoću intenziteta boje uslikane fotografije, odnosno pomoću RGB (*skr. od eng. Red Green Blue, crvena zelena plava*) modela predstavljanja boja. Model RGB definira boju zadavanjem intenziteta svjetlosti za tri boje: crvenu, zelenu i plavu, koje su primarne boje pomenutog modela. Umjesto zelene mi smo koristili žutu boju, tako da sve što se odnosi na zelenu boju u našem slučaju predstavlja žuta boja.

Definirani raspon intenziteta svjetlosti za svaku boju je 8 bitova, od 0 do 255 (od #00 do #FF heksadekadski), tako da se njihovim miješanjem dobiva 24-bitna paleta s $256 \times 256 \times 256 = 16.777.216$ boja, što većina današnjih hardvera podržava.

Jedinica boje u ovom modelu može se predstaviti na tri načina:

- ***#RRGGBB*** - heksadekadskim brojem, u kojem svaki par brojeva redom predstavlja intenzitet crvene, zelene (tj. žute) i plave boje, npr. #808080 (može se pisati malim ili velikim slovima, ali obavezno s vodećom nulom za brojeve $\#0 - \#F$ i bez razmaka); ovaj način dopušta i kraću notaciju s 3 znamenke u obliku ***#RGB***, koja se konvertira u punu notaciju udvajanjem broja te tako na primjer #$A9C$ postaje #$AA99CC$
- ***rgb(R, G, B)*** - dekadskim vrijednostima za svaku boju, npr. $rgb(128, 128, 128)$
- ***rgb(R%, G%, B%)*** - postotkom intenziteta određene boje od $0\% = 0$ do $100\% = 256$, npr. $rgb(50\%, 50\%, 50\%)$ (znak za postotak obavezan je i kad je vrijednost 0, za razliku od jedinica dužine gdje se može izostaviti).

4. Zaključak

Projekat je predstavljao kreiranje aplikacije za razvrstavanje objekata prema boji koristeći Raspberry Pi, Raspberry Pi kameru i step motore. Čitavim sistemom (tj. step motorima i kamerom) upravljali smo kodom upisanim na razvojni sistem (opisan u poglavlju 3). Za postavku microSD kartice Raspberry-ja koristili smo upute opisane u poglavlju 2. Radi lakše izrade projektni zadatak smo podijelili u više dijelova: postavka Raspberry-ja, rad kamere, upravljanje motorima, upravljanje kamerom te pravljenje cijele konstrukcije od ploča plastike. Kada smo sve te dijelove uspješno obavili, posvetili smo se sastavljanju jedne cjeline. Prilikom toga nailazili smo na određene probleme, koje smo na kraju uspješno riješili.

Prvi od problema bio je vezan za postavku Raspberry-ja i microSD kartice, koje do sada nismo koristili te smo uputstva za instalaciju našli na oficijalnoj Raspberry Pi web stranici (link stranice). Još jedan od problema bio je rad kamere sa kojom se, također, do sada nismo susretali, ali smo istraživanjem uspjeli to riješiti.

Ovaj projekat može se iskoristiti za razvrstavanje objekata prema boji. U našem slučaju kao objekte koristili smo bombone kako bi dimenzije cijelog sistema bile manje, međutim sistem se može iskoristiti i za veće objekte uz ploče plastike većih dimenzija. Razvrstavanje smo vršili na tri boje: crvena, žuta i plava, ali naravno razvrstavanje se može vršiti i na druge boje ili više boja promjenom dijela koda namijenjenog za to (kod dat u poglavlju 3). U svemu navedenom ogleda se funkcionalnost našeg projekta koji je i pored svih problema na koje smo naišli uspješno završen.

Literatura

[1] https://www.raspberrypi.org/, septembar 2019.

[2] Datasheet PIC16F1939 http://ww1.microchip.com/downloads/en/DeviceDoc/40001574C.pdf, septembar 2019.

[3] https://en.wikipedia.org/wiki/Stepper_motor, septembar 2019.

[4] https://en.wikipedia.org/wiki/Raspbian, septembar 2019.

[5] https://en.wikipedia.org/wiki/Thonny, septembar 2019.

[6] Laboratorijska vježba 5, Aktuatori, Elektrotehnički fakultet u Sarajevu, Jasmin Velagić, Sarajevo maj 2018.