**SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU ELEKTROTEHNIČKI FAKULTET**

**Sveučilišni diplomski studij**

**BESKONTAKTNO UPRAVLJANJE SERVO MOTORIMA POMOĆU KAMERESeminarski rad iz kolegija „Distribuirani Računalni sustavi“  
  
  
  
  
Antonio Lončar  
  
  
  
  
  
  
  
  
  
  
Osijek, 2014/2015.**

sadržaj

[1. Uvod 3](#_Toc412754500)

[1.1. Zadatak i ciljevi rada 3](#_Toc412754501)

[2. Realizacija sustava 4](#_Toc412754502)

[2.1. Sklopovska realizacija 4](#_Toc412754503)

[2.1.1. Mikroupravljački sustav - Arduino „EkitsZone UNO Rev.3“ 4](#_Toc412754504)

[2.1.2. Sustav pozicioniranja kamere 5](#_Toc412754505)

[2.1.3. Električna shema 7](#_Toc412754506)

[2.2. Programsko rješenje 8](#_Toc412754507)

[2.2.1. Kod pokretan na Arduinu 8](#_Toc412754508)

[2.2.2. Kod pokretan na računalu 9](#_Toc412754509)

[2.2.3. Izlged korisničkoga sučelja 14](#_Toc412754510)

[3. Zaključak 15](#_Toc412754511)

[4. Literatura 16](#_Toc412754512)

# Uvod

Napredak tehnologije rezultirao je povećanjem performansi osobnih računala i njihovih mogućnosti. Inovativni i dobro napisani algoritmi dostupni na internetu dopuštaju svakome da bude kreativan i iskoristi mogućnosti svojega računala. Za obradu slike se koristi mnogo matematičkih operacija ali navedeni algoritmi nam olakšavaju posao. Biblioteke koje sadrže te algoritme dopuštaju krajnjemu korisniku da ne mora znati složenu matematiku nego da koristeći ugrađene metode dobije rezultat u trenu oka. Obrada slike je glavni dio ovoga projekta i bez nje ne bi bilo beskontaktnog upravljanja. Ovo je upravljanje servo motorima gdje se ne koriste fizički prekidači već informacije sa kamere. Kamera na laptopu prikuplja informacije u obliku slike, računalo obrađuje sliku i sprema najbitnije podatke. Podaci o položaju predmeta na slici se šalju na Arduino koji zatim pomiče servo motore. Projekt se sastoji od dva dijela, aplikacija na računalu, te koda koji se izvodi na Arduinu. Više o svakome dijelu i njihov izvorni kod u nastavku seminara.

## Zadatak i ciljevi rada

Prema dogovoru sa profesorom napravljen je projekt „Beskontaktno upravljanje servo motorima promoću kamere“. Projekt je zamišljen tako da pomoću kamere detektiramo predmet te njegovim pomicanjem pomičemo servo motore. Detekcija predmeta se zasniva na detekciji boje i određivanjem najveće konture. Koordinate centroida najveće konture su buduće pozicije servo motora.

# Realizacija sustava

## Sklopovska realizacija

### **Mikroupravljački sustav - Arduino „EkitsZone UNO Rev.3“**

„Uno je mikrokontrolerska ploča bazirana na ATmega328p. Ima 14 digitalnih ulaznih/izlaznih pinova (od kojih 6 mogu biti korišteni kao PWM izlaz), 6 analognih ulaza, 1 UART (hardverski serijski portovi), 16 MHz kristalnog oscilatora, USB konekciju, priključak napajanja, ICSP zaglavlje i tipku za reset.Sadržava sve potrebno za podršku mikrokontroleru; jednostavno se spaja s računalom s USB kablom ili s napajanjem s AC na DC pretvornikom ili s baterijom za početak rada.“



**Slika 2.1.** Arduino EkitsZone UNO Rev.3

|  |  |
| --- | --- |
| **Mikrokontroler** | Atmega328p |
| **Operacijski napon** | 5V |
| **Ulazni napon (preporučen)** | 7 – 9V |
| **Ulazni napon (granični)** | 7 – 12 V |
| **Digitalni I/O pinovi** | 14 (od kojih pružaju 6 PWM izlaza) |
| **Analogni ulazni pinovi** | 6 |
| **DC struja po I/O** | 40 mA |
| **DC struja za 3.3V pin** | 150 mA |
| **Flash memorija** | 32 KB (od kojih 0.5 KB se koristi za bootloader) |
| **SRAM** | 2 KB |
| **EEPROM** | 1 KB |
| **Clock brzina** | 16 MHz |

**Tablica 2.1.** Specifikacije Arduina „EkitsZone UNO Rev.3“ [1]

### Sustav pozicioniranja kamere

Servo motori imaju tri žice: napajanje, uzemljenje i signal. Žica za napajanje je crvene boje i spajamo je na izvor od 5V. Žica za uzemljenje je smeđe boje i spajamo je na uzemljenje. Signalna žica je žute boje i treba biti spojena na digitalni Arduino pin. Moramo imati na umu kako servo motori povlače znatnu snagu, pa ako radimo istovremeno sa više od dva servo motora, vjerojatno ćemo trebati osigurati odvojeno napajanje. Korišteni su servo motori su prikazani na Slici 2.2.



**Slika 2.2.**SG90 Mini Gear Micro Servo.

|  |  |
| --- | --- |
| **Modulacija** | Analogna |
| **Moment sile [4.8V]** | 1.80 kg-cm |
| **Brzina [4.8V]** | 0.10 s/60° |
| **Masa** | 9.0g |
| **Dimenzije** | Dužina: 23.1 mm  Visina: 12.2 mm  Širina: 29.0 mm |
| **Tip motora** | 3-pole |
| **Tip zupčanika** | Plastični |
| **Kut zakretanja** | 0° - 180° |
| **Širina pulsa** | 500-2400 µs |

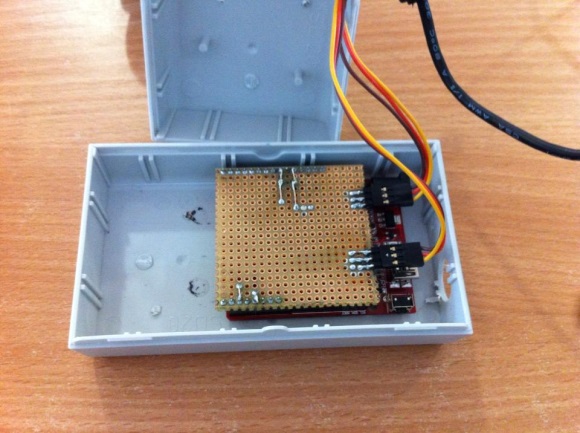
**Tablica 2.2.** Specifikacije SG90 Mini Gear Micro servo motora.[2]

Stalak se sastoji od dijela koji je stacionaran i od dijela koji je pomičan. Stacionarni dio stalka je pričvršćen za kutiju, a pomični dio stalka služi za pomicanje kamere po *x* i *y* osi.



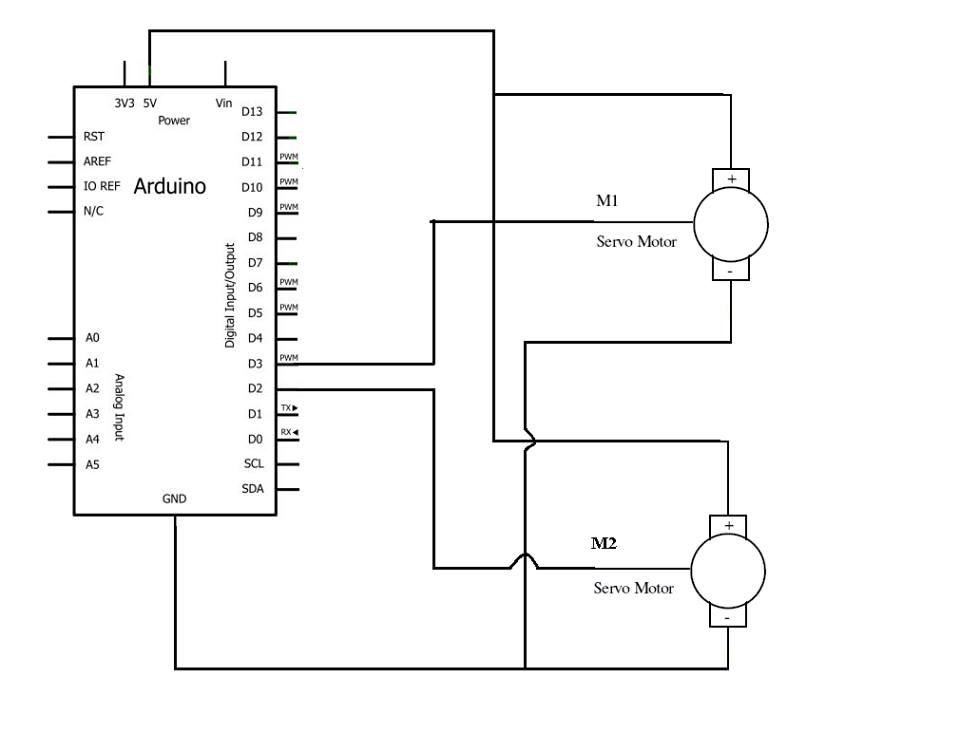
**Slika 2.3.** Stalak za servo motore i kameru.

Na Slici 2.4. se može vidjeti kako izgleda kutija sa sarvomotorima i unutrašnjost kutije.

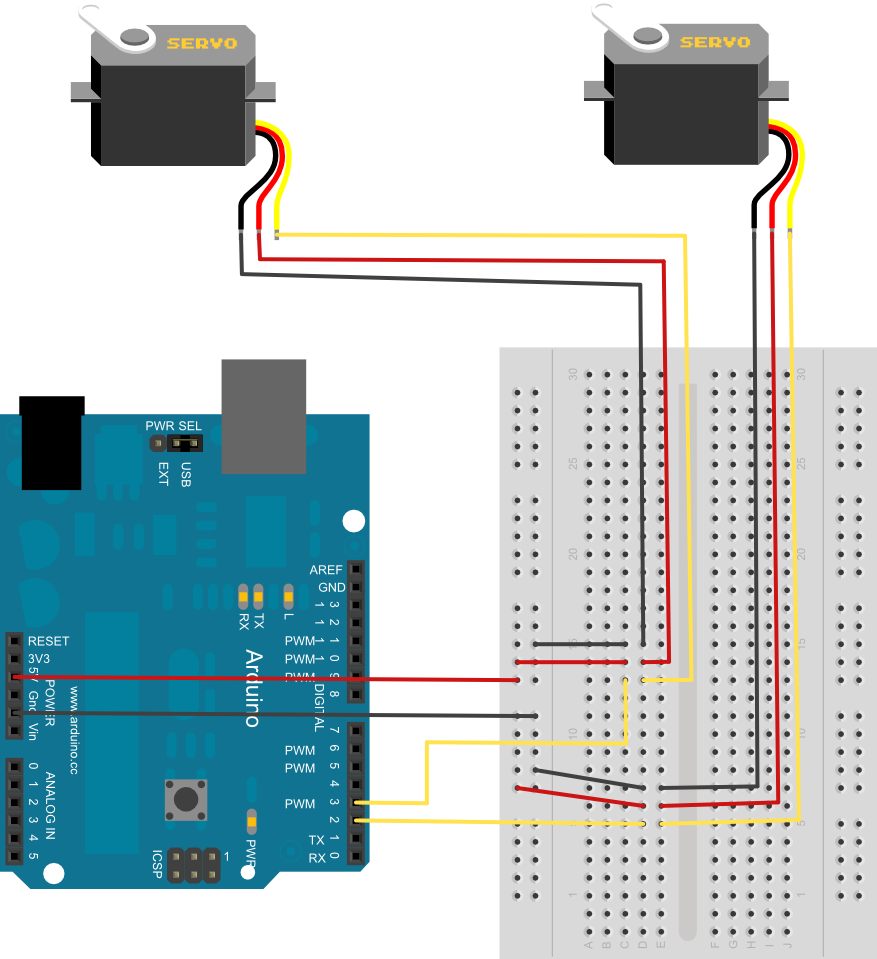
 

**Slika 2.4.**Kutija sa Arduinom i servo motorima.

### Električna shema



**Slika 2.5.** Shema spajanja servo motora na Arduino.



**Slika 2.6.** Grafički prikaz sheme. [3]

## Programsko rješenje

Programsko rješenje je realizirano pisanjem dvaju odvojenih kodova, jedan za Arduino te jedan pokretan na računalu. Kod za Arduino je pisan u Arduino IDE-u [C++], a program pokretan na računalu je pisan u Procesing2 IDE-u [Java]. U sljedećim poglavljima će biti opisani navedeni kodovi.

### Kod pokretan na Arduinu

Arduino u ovome projektu služi kao prijamnik podataka koji se šalju USB serijskom vezom sa računala, obrađivanjem tih podataka te pomicanjem servo motora. Podaci koji se šalju sadrže stupnjeve zakretanja x-os servo motor i y-os servo motor. Servo motori se mogu pomicati od 0 do 180 stupnjeva. Rukovanje servo motorima je olakšano ugrađenom Arduino bibliotekom *Servo.h.*

#include <Servo.h>

const int servoPinX = 2; // pin jednog motora 2

const int servoPinY = 4; // pin drugog motora 4

Servo myServoX; //motor na pinu 2

Servo myServoY; //motor na pinu 4

int x = 90;

int y = 90;

void setup() {

Serial.begin(9600);

myServoX.attach(servoPinX);

myServoY.attach(servoPinY);

}

void loop() {

if(Serial.available() > 0)

{

x = Serial.parseInt();

y = Serial.parseInt();

if(x && y)

{

myServoX.write(x);

myServoY.write(y);

}

}

}

U *setup()* metodi se započinje serijska veza na 9600 bauda, te se postavljaju servo motori na 90 stupnjeva. U *loop()* metodi se svaki trenutak provjerava da li je serijska veza dostupna te ako je parsiraju se cijelobrojni podaci iz zaprimljenoga niza podataka. Ako postoje *x* i *y* vrijednosti veće od 0, servo motori se postavljaju na njih.

### Kod pokretan na računalu

Kod pokretan na računalu je pisan u Processing2 IDE-u Java programskim jezikom. Njegova zadaća je sljedeća:

1. Zabilježiti sliku sa web kamere.
2. Obraditi je te pronaći koordinate centroida koji predstavlja objekt određene boje.
3. Koordinate centroida skalirati i poslati serijskom vezom prema Arduinu.
4. Prikazati grafičko sučelje za korisnika i omogučiti odabir boje objekta.

Realizacija navedenih zahtjeva je pokazana u četiri odvojena koda.

U prvomedijelu koda pozvane su biblioteke i dodjeljena su imena varijablama. Biblioteke su sljedeće: *gab.opencv.\**[4] za obradu slike, biblioteka *processing.serial.\** za serijsku vezu, biblioteka *processing.video.\** za snimanje slika sa web kamere te biblioteka *java.awt.Rectangle* koja omogućuje objekt *Rectangle*. U komentarima je napisano koja se varijabla za šta koristi.

import gab.opencv.\*;

import processing.serial.\*;

import processing.video.\*;

import java.awt.Rectangle;

Serial arduinoPort; //objekt za serijsku vezu

OpenCV opencv; //objekt za obradu slike

Capture cam; //objekt za kameru

int baud = 9600; //brzina serijske veze

int[] send = new int[2]; //podaci koji se šalju na arduino

int servoPositionX; //servo X pozivija

int servoPositionY; //servo Y pozicija

boolean mouseClick = true;

int lowerb = 0, upperb = 20; //gornja i donja granica boje

ArrayList<Contour> contours; //konture iz kojih se računaju koordinate centroida

Drugi dio koda je setup() funkcija, poziva se jedanput i u njemu su inicijalizirani podaci. Neke od bitnijih funkcija u ovome dijelu su: funkcija *write(),* objekta*Serial*, koja šalje niz znakova serijskom vezom na Arudino te funkcija *start(),* objekta*Capture,* koja započinje snimanje slika web kamerom. Uzorak niza koji se šalje serijskom vezom na Arduino je „broj:broj:“, razlog je objašnjen u komentarima.

void setup() {

size(640, 480); //veličina ekrana

background(255, 255, 0); //boja pozadine

opencv = new OpenCV(this, 640, 480); //instanciranje objekta OpenCV

cam = new Capture(this, 640, 480); //instanciranje objekta Capture

contours = new ArrayList<Contour>(); //instanciranje niza sa objektima Contuor

String portName = Serial.list()[0]; //ime prvoga porta sa liste, 'COM X'

arduinoPort = new Serial(this, portName, baud); //instanciranje objekta Serial

//inicijalizacija pozicije servo motora

send[0] = 90;

send[1] = 90;

//kada se pošalje umjesto : razmak ' ', treba mu dugo da obradi poruku

//: je proizvoljno odabrano

//umjesto : moze biti bilo koji znak ili slovo, ne broj!

arduinoPort.write(send[0] + ":" + send[1] + ":"); //slanje podataka na Arduino

cam.start(); //početak snimanja sa kamere

}

Treći dio koda je funkcija *draw()* koja se neprestano poziva sve dok program radi ili dok je neka od funkcija za zaustavljanje ne zaustavi.

Svaku iteraciju funkcija *draw()* radi sljedeće: oboji pozadinu žutom bojom i u sredini prozora nacrta kvadrat veličine 50 x 50 piksela. Nakon toga spremi *frame* sa web kamere u *OpenCV* objekt, pretvori ga u HSV prostor boja, izdvoji samo Hue kanal i na njemu izvede funkciju *inRange()* kojoj predamo parametre gornje i donje granice. Ograđena slika na kojoj se nalazi crno-bijeli obris, dio hue kanala koji je unutar gornje i donje granice, prikazana je u desnome dole kutu prozora. Ta slika nam služi za bolje i preciznije određivanje boje predmeta kojega pratimo. Ručno pomicanje granica se izvodi pritiskanjem slova Q,W i A,S. Na slici koja bijelom bojom opisuje predmet primjenimo funkciju *findContours()*, objekta OpenCV, koja pronalazi konture na slici. Zatim se odrede koordinate najveće konture i spreme se u varijable *servoPositionX* i *servoPositionY*.

Lijevi klik miša predstavlja On/Off gumb za aplikaciju. Ako nismo pritisnuli lijevi klik miša aplikacija neće slati podatke Arduinu i servo motori se neće pomicati. Za početak rada potrebno je pritisnuti lijevi klik miša. Ako se koordinate predmeta sa slike, nalaze unutar kvadrata od 50x50 piksela u sredini ekrana ili nije pritisnut klik miša, servo motori će biti postavljeni na inicijalnu poziciju. Ako je kliknut lijevi klik miša i koordinate najveće konture se nalaze izvan kvadrata, podaci će se slati na Arduino i pomicati će servo motore. Kako mi se vizualno moglo znati koje su koordinate najveće konture, na njihovo mjesto se crta bijela točka.

void draw() {

background(255, 255, 0);

fill(255, 255, 255);

strokeWeight(1);

rect(width/2 - 25, height/2 - 25, 50, 50);

/\*dohvaćanje slike\*/

opencv.loadImage(cam);

/\*obrada slike\*/

opencv.useColor(HSB);

opencv.setGray(opencv.getH().clone());

opencv.inRange(lowerb, upperb);

//opencv.blur(30);

image(opencv.getOutput(), 3\*width/4, 3\*height/4, width/4, height/4);

/\*ptonađi najvecu konturu\*/

contours = opencv.findContours(true, true);

if (contours.size() > 0) {

Contour biggestContour = contours.get(0);

Rectangle r = biggestContour.getBoundingBox();

/\*uodejtaj koordinate tocke objekta\*/

servoPositionX = Math.abs(r.x + r.width/2 - width); //inverzne koordinate x osi

servoPositionY = Math.abs(r.y + r.height/2);

}

//ako se nalazi unutar kvadrata ili nije pritisnuta tipka miša

//ne pomiči servo motore

if(isInside(servoPositionX, servoPositionY) || mouseClick){

send[0] = 90;

send[1] = 90;

arduinoPort.write(send[0] + ":" + send[1] + ":");

delay(5);

}else{

/\*servoPositionX 0-640 -> 0-180

servoPositionY 0-480 -> 0-180\*/

send[0] = (180 - servoPositionX\*180/640);

send[1] = (servoPositionY\*180/480);

arduinoPort.write(send[0] + ":" + send[1] + ":");

delay(5);

}

fill(255, 255, 255);

strokeWeight(1);

ellipse(640 - send[0]\*640/180, send[1]\*480/180, 10, 10);

}

Četvrtio dio koda su pomoćne funckije koje se koriste u *draw()* funkciji. To su funkcija *isInsideI()* koja vraća true ako se *x* i *y* koordinate nalaze unutar kvadrata od 50 x 50 piksela na sredini prozora. Funkcija *mouseClicked()* koja mjenja stanje varijable *mouseClicked,* pomoću te varijable se određuje da li se podaci šalju na Arduino ili se ne šalju. Funkcija *inRange()* mjenja gornju i donju granicu funkciji *inRange()*, objekta OpenCV, koja služi za određivanje boje objekta. Te funkcija *captureEvent()* bez koje *Capture* objekt ne bi mogao snimati sliku sa web kamere.

boolean isInside(int coordx, int coordy)

{

if((coordx > width/2 - 25 && coordx < width/2 + 25) &&

(coordy > height/2 - 25 && coordy < height/2 + 25))

{

return true;

}

else return false;

}

void mouseClicked()

{

mouseClick = !mouseClick;

}

void keyPressed() {

if(char(keyCode) == 'Q')

lowerb += 2;

else if(char(keyCode) == 'W')

lowerb -= 2;

if(char(keyCode) == 'A')

upperb += 2;

else if(char(keyCode) == 'S')

upperb -= 2;

println(lowerb + " " + upperb);

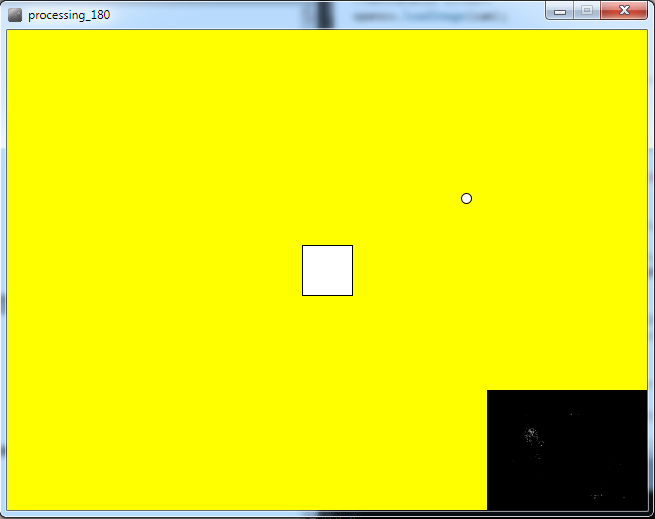
}

void captureEvent(Capture c) {

c.read();

}

### Izlged korisničkoga sučelja



**Slika 2.7**. Izlged korisničkoga sučelja

# Zaključak

U ovome seminaru je napravljena aplikacija za beskontakto upravljanje servo motorima. Korisnik nema potrebu dodirivati tipkovnicu kako bi upravljao servo motorima već samo treba odrediti koji predmet je zadužen za upravljanje. Praćenje predmeta se temelji na najjednostavniom detekcijom boje. Aplikacija koja se pokreće na računalu određuje poziciju servo motora na osnovu predmeta koji se nalazi ispred kamere. Koordinate predmeta se pretvaraju u brojke od 0 do 180 te ih se šalje na Arduino koji pomiče servo motore. Detekcija boje ima puno mana pa bih odma naveo da ovo nije metoda upravljanja koja će zamjeniti stare već provjerene metode. Korištenjem drugih metoda detekcije i praćenja objekata na slici ova metoda bi se itekako poboljšala.

# Literatura

[1] Arduino Uno r3, <http://arduino.cc/en/Main/ArduinoBoardUno>

[2] TowerPro SG90 - Micro Servo, <http://www.servodatabase.com/servo/towerpro/sg90>

[3] PanTilt Servo, <https://sparkfun.com/images/tutorials/PanTiltFaceDetection/PanTilt-7S.png>

[4] opencv-processing, <https://github.com/atduskgreg/opencv-processing>