SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

ELEKTROTEHNIČKI FAKULTET

Projekt iz kolegija „Elementi automatike“

**Face Tracking System**

Antonio Lončar

Matko Čeme

Osijek, 2013/2014.

Sadržaj

[1. Uvod 3](#_Toc391929771)

[1.1. Zadatak i ciljevi rada 3](#_Toc391929772)

[2. Sustav prepoznavanja i praćenja lica 4](#_Toc391929773)

[2.1. Idejno rješenje prijedloga strukture sustava 4](#_Toc391929774)

[2.2. Detekcija lica – Viola-Jones algoritam 4](#_Toc391929775)

[2.3. Praćenje pomaka lica u prostor-vremenu 6](#_Toc391929776)

[3. Realizacija sustava 8](#_Toc391929777)

[3.1. Sklopovska realizacija 8](#_Toc391929778)

[3.1.1. Mikroupravljački sustav - Arduino „EkitsZone UNO Rev.3“ 8](#_Toc391929779)

[3.1.2. Sustav pozicioniranja kamere 9](#_Toc391929780)

[3.1.3. Električna shema 11](#_Toc391929781)

[3.2. Programsko rješenje 12](#_Toc391929782)

[3.2.1. Algoritam detekcije i praćena ljudskog lica – blok dijagram toka 12](#_Toc391929783)

[3.2.2. Realizacija algoritama detekcije i praćena ljudskog lica 13](#_Toc391929784)

[3.2.3. Upravljanje sustavom pozicioniranja kamere 15](#_Toc391929785)

[3.3. Ekonomska bilanca izgradnje rješenja 16](#_Toc391929786)

[4. Ispitivanje i evaluacija sustava 17](#_Toc391929787)

[4.1. Postupci, metode i kriteriji ispitivanja i evaluacije 17](#_Toc391929788)

[4.2. Usporedba praćenja sa jednokanalnim i dvokanalnim histogramom 18](#_Toc391929789)

[4.3. Usporedba praćenja sa dinamičkim i statičkog histogramom 21](#_Toc391929790)

[5. Zaključak 24](#_Toc391929791)

[6. Literatura 25](#_Toc391929792)

[7. Prilozi 26](#_Toc391929793)

[8. Elektronička verzija 27](#_Toc391929794)

# Uvod

Napredak tehnologije omogućio je obradu velikog broja informacija. Slika kao skup informacija predstavljena slikovnim elementima (*pixel*) istom tehnologijom je postala obradiva tako da se krajnje rješenje prikazuje u stvarnome vremenu. Do prije dvanaest godina nemoguće, 2001. godine inžinjeri Paul Viola i Michael Jones predstavili su prvo rješenje za praćenje objekata u stvarnome vremenu. Nakon toga se dogodio veliki napredak u toj znanstvenoj grani. Definiranjem objekta kao ljudsko lice, u mogućnosti smo detektirati, prepoznati i pratiti određene ljude. Primjena je višestruka, od sigurnosti i zaštite pa sve do vojske i uhođenja ljudi.

## Zadatak i ciljevi rada

Prema dogovoru s mentorom napravili smo projekt „Face Tracking System“. „Face Tracking System“ je sustav za praćenje ljudskog lica. Sustav je zamišljen tako da program pomoću kamere detektira ljudsko lice, izračuna u kojem se smjeru kreće lice i prema tome šalje arduinu informacije u kojem smjeru treba pomicati servo motore koji pomiču kameru. Detaljniji opis kako sustav izgleda i kako radi opisan je u nastavku seminara.

# **Sustav prepoznavanja i praćenja lica**

## Idejno rješenje prijedloga strukture sustava

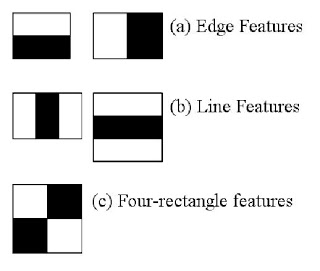
Prvobitna ideja praćenja zasnivala se strogo na detekciji lica. Računalo bi spremilo okvira koji trenutno prikazuje (*frame*) sliku sa kamere, *frame* bi se obradio i ako se detektira lice na njemu ono bi se pratilo, postupak bi se ponavljao sve dok se ne ugasi kamera. Odmah na početku, zbog svojih nedostataka, taj algoritam je bio odbačen i morali smo se dosjetiti nekih drugih potencijalnih rješenja. Jedno od njih je bila metoda kombiniranja algoritma detekcije lica Viole-Jonesa i Camshift metode praćenja.

Umjesto da svaki *frame* detektiramo lice, odlučili smo to obaviti samo na početku algoritma, a zatim nakon toga pratiti karakterističnu boju lica. Karakterističnu boju lica pronalazimo računanjem histograma boja na području gdje se nalazi lice. Primjenjivanjem odgovarajuče maske, izostavljamo boju pozadine ili drugih predmeta. Nakon pronalaska odgovarajučega histograma, primjenjuje se metoda koja određuje vjerojatnosti pronalaska toga histograma na cijeloj slici koju dobivamo od kamere. Nakon što smo odredili vjerojatnost gdje se može pojaviti naše lice, koristimo metodu praćenja i pratimo lice. Prednosti koje nam donosi pračenje boje lica, su brzina obrađivanja slike i otpornost algoritma na zakretanje glave.

## Detekcija lica – Viola-Jones algoritam

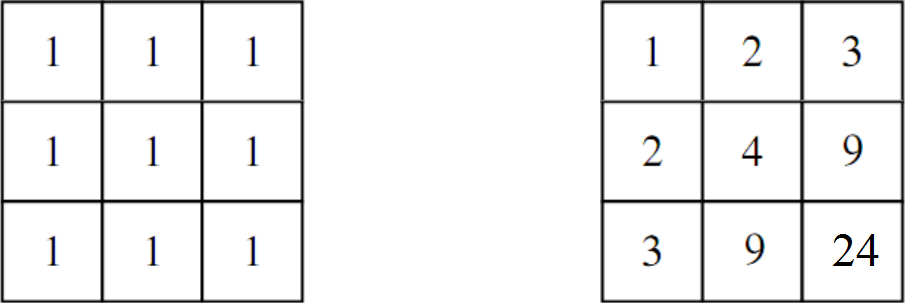
Detekcija lica koja se spominje u predhodnome poglavlju u ovome poglavlju je ukratko objašnjeno. 2001. godine Paul Viola i Michael Jones predstavili su algoritam za detekciju objekta. Algoritam temeljen na pristupu strojnog učenja gdje se klasifikator uči kako bi mogao prepoznavati traženi objekt.

Na početku algoritma klasifikator se trenira sa nekoliko tisuća slika objekta koji želimo detektirati, te slike se zovu pozitivne slike i isto toliko slika koje ne sadrže željeni objekt, njih zovemo negativne slike. Autori algoritma su trenirali svoje klasifikatore na slikama veličine 24x24 piksela. Iz slika se izdvajaju značajke pomoću haarovih likova. Slika 2.1. prikazuje haarove likove.



**Slika 2.1.** Haarovi likovi korišteni u izdvajanju značajki.

Svaki haarov lik je pojedinačna vrijednost dobivena oduzimanjem zbroja piksela ispod bijelog kvadrata sa zbrojem piskela ispod crnoga kvadrata. Autori algoritma su za prozor 24x24 piksela dobili oko 160000 izračunatih značajki. Računanje suma haarovih likova može biti izuzetno procesno zahtjevno, zbog toga se uvodi pojam integralne slike. Integralna slika ubrzava račuanje suma piksela. Primjer integralne slike prikazana je na Slici 2.2. Možemo vidjeti da je vrijednost integralne slike na x,y koordinati jednak zbroju vrijednosti svih piksela lijevo i iznad.



**Slika 2.2.** Primjer vrijednosti integralne slike.

Kako bi smanjili broj pronađenih bezznačajnih značajki i pronašli one najbolje koje raspoznaju lice od ne-lica modificiran je AdaBoost algoritam. Izabiranjem značajki koje najbolje raspoznaju lice od ne-lica i odbacivanjem onih loših gradimo slabe klasifikator. Slabi klasifikatori sami ne mogu klasificirati sliku. No njih spajamo u jaki klasifikator koji može raspoznati željeni objekt na slici. Ova metoda smanjuje 160000 značajki za prozor 24x24 piksela na samo 6000 značajki. Primjenjivanjem 6000 značajki za svaki prozor slike veličine 24x24 piksela može biti neefikasno i dugotrajno. Znamo i sami da na cijeloj slici ima prostora gdje se ne nalazi lice, kako bi izbjegli bespotrebno provjeravanje takvih dijelova slike Paul Viola i Michael Jones su napravili kaskadnu klasifikaciju. 6000 značajki je raspodjeljeno na razine koje predstavljaju klasifikatore. Ako prozor slike ne prođe prvu razinu klasifikatora, on se odbacuje, ako prođe, nastavlja dalje na drugu razinu. Kada prozor slike prođe sve razine klasifikatora, za njega možemo reći da sadrži lice. Ova metoda detekcije objekta uvodi i klasifikatore koji se mogu „rastezati“, za razliku od prijašnjih metoda gdje su klasifikatori bili nepromjenjive veličine, a slika se rastezala.

Ukoliko želite opširnije objašnjenje predlažemo Vam da pročita znanstveni članak Paul Viola i Michael Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”[1] u kojemu je predstavljena metoda detekcije objekata.

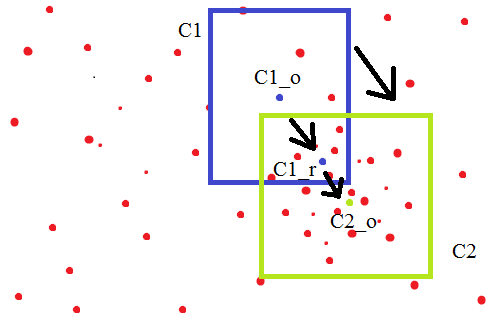
## Praćenje pomaka lica u prostor-vremenu

Nakon što smo detektirali lice algoritmom Viole-Jonesa i izdvojili ga sa slike, računamo histogram boje lica. Izračunati histogram je početna stanica praćenja lica. Pomoću *histogram backprojection*[2]metode pronalazimo mjesta na *frameu* kamere koja najbolje opisuju izračunati histogram.

*Histogram backprojection* predstavlja vjerojatnost koliko dobro vrijednosti piksela *framea* kamere pripadaju vrijednostima piksela boje lica, distribucija piksela boje lica nalazi se u izračunatome histogramu. Možemo predpostaviti da mjesta koja imamo najveću vjerojatnost opisuju prostor gdje se nalazi lice. Algoritam *histogram backprojection* metode je sljedeći:

* Za svaku vrijednost piksela framea kamere pronađi mjesto (*bin*) u histogramu.
* Očitaj vrijednost koja se nalazi na tome mjestu.
* Napravi novu sliku i spremi tu vrijednost na poziciju početnoga piksela.

Dobivena slika predstavlja *backprojection* sliku. No slika sama kao takva ne pratiti lice, alat koji koristimo za praćenje lica je *camshift*[3] metoda praćenja. *Camshift* je skraćeni naziv za „*Continuously Adaptive Meanshift*“. Za razliku od camshift metode praćenja, meanshift metoda koja se spominje u punome nazivu ne mjenja veličinu kvadrata koji opisuje objekt koji se prati. Na samome početku praćenja postavimo kvadrat veličine lica na koordinate gdje je algoritam detekcije lica detektirao lice. Nakon toga camshift metoda pomiće taj kvadrat ovisno o gustoći piksela koja je pokazana backprojection slikom. Kako se mjenja backprojection slika iz framea u frame, veličina i pozicija kvadrata se isto mjenjaju. Slika 2.3. predstavlja naćin rada *camshift* metode praćenja.



**Slika 2.3.** Način rada camshift metode praćenja.

Na slici 2.3. možemo vidjeti dva kvadrata, inicijalni kvadrat „C1“ i konačni kvadrat „C2“. Inicijalni centar kvadrata „C1“ je „C1\_o“, ali centar centroida piksela unutar kvadrata „C1“ se nalazi na mjestu „C1\_r“. Camshift metoda pomiče centar „C1\_o“ prema centru „C1\_r“. Taj postupak pomicanja se nastavlja sve dok centar kvadrata ne bude jednak centru centroida piksela unutar kvadrata. Konačno, centar centroida piksela i centar kvadrata je isti „C2\_o“, što bi značilo da kvadrat „C2“ opisuje prostor sa najvećom gustoćom piksela na slici.

# **Realizacija sustava**

## Sklopovska realizacija

Sklop je realizirana pomoću Arduino sustava, dva servo motora i web kamere. Servo motori spojeni na Arduino pomiću web kameru i pozicijoniraju je prema detektiranome licu. U slijedećim podnaslovima su opisani dijelovi sklopovskoga riješenja.

### **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.“ [4]



**Slika 3.1.** 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 3.1.** Specifikacije Arduina „EkitsZone UNO Rev.3“

### Sustav pozicioniranja kamere

Servo motori imaju tri žice: napajanje, uzemljenje i signal. Žica za napajnaje 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 3.2.



**Slika 3.2.** 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 |

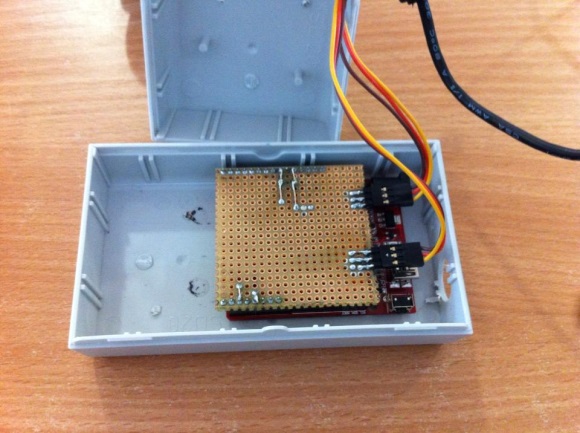
**Tablica 3.2.** Specifikacije SG90 Mini Gear Micro servo motora.[5]

Stalak za kameru i servo motore služi kako bi servo motori pravilno pomicali kameru. Stalak se sastoji od dijela koji je stacionaran i od dijela koji je pomičan. Stacionarni dio stalka je pričvršćen za neki objekt, a pomični dio stalka služi za pomicanje kamere po x i y osi.



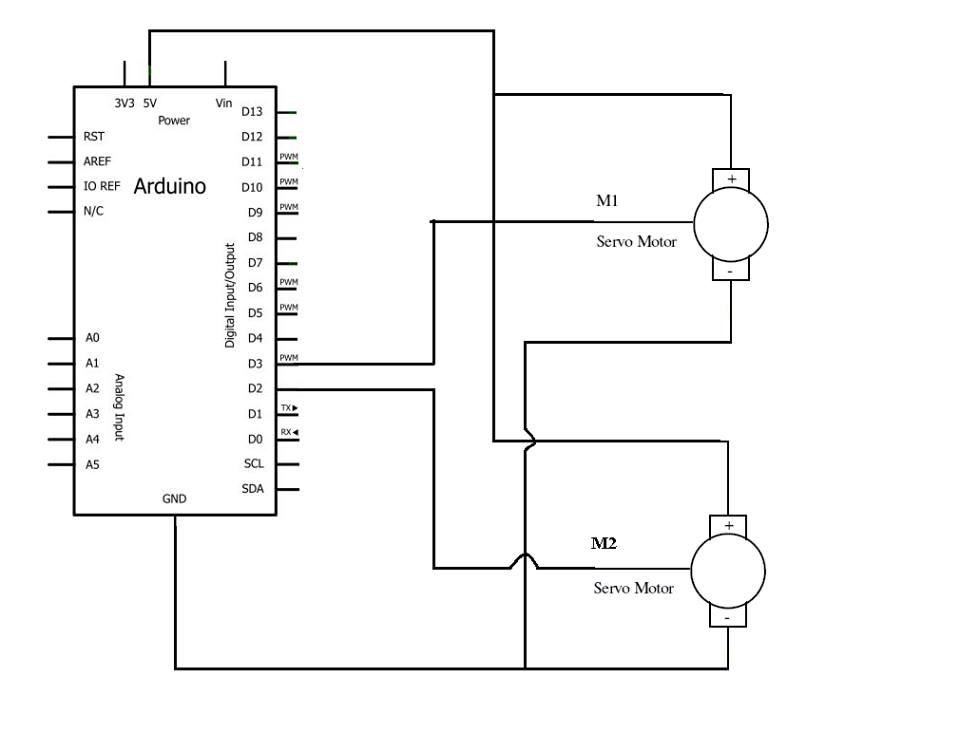
**Slika 3.3.** Stalak za servo motore i kameru.

Na Slici 3.4. se može vidjeti kako izgleda „Face Tracking“ sustav.

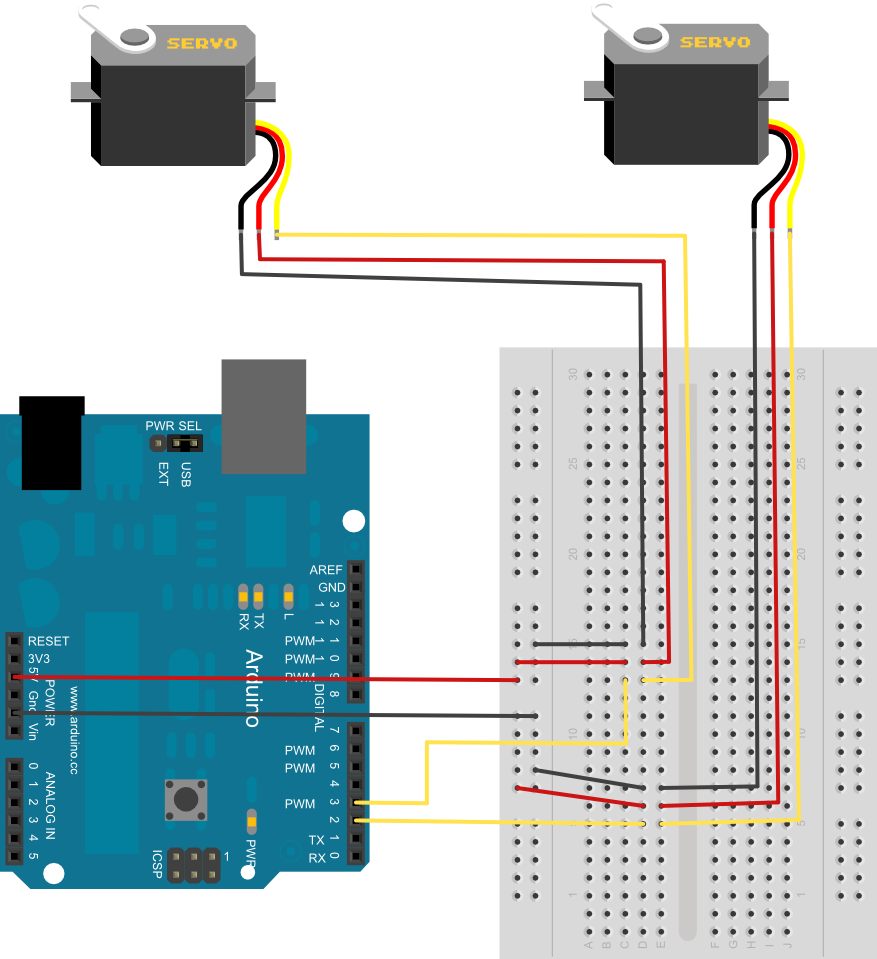
 

**Slika 3.4.** Izgled „Face Tracking“ sustava.

### Električna shema



**Slika 3.5.** Shema spajanja servo motora na Arduino.



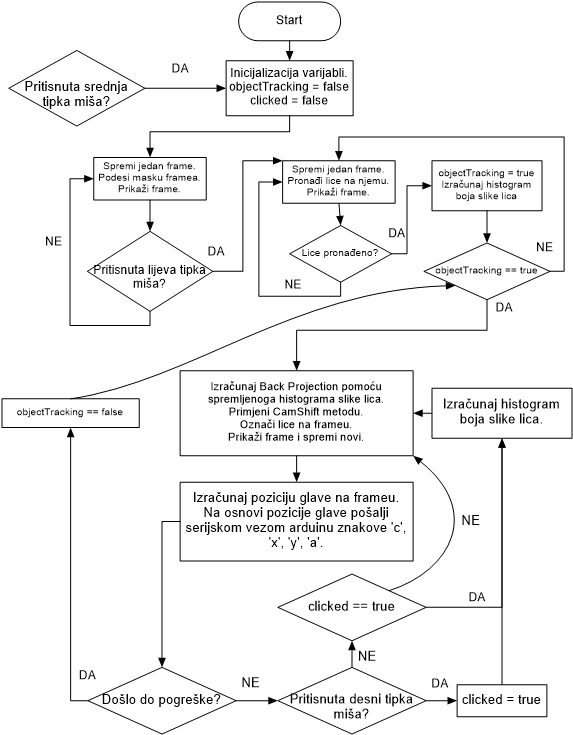
**Slika 3.6.** Grafički prikaz sheme

## **Programsko rješenje**

Programsko rješenje je realizirano sa C++ programskim jezikom i bibliotekom OpenCv[6]. Iako postoji veliki broj programskih jezika sa kojima se može realizirati ovo rješenje, C++ je izabran zbog iskustvu u programiranju s njime, a OpenCV biblioteka zbog otvorenoga koda i jednostavnosti. U sljedećim poglavljima će biti opisan algoritam detekcije i praćenja lica te sama realizacija algoritma.

### Algoritam detekcije i praćena ljudskog lica – blok dijagram toka

Slika 3.6. predstavlja dijagram toka koji opisuje krajnji algoritam detekcije i praćenja lica.

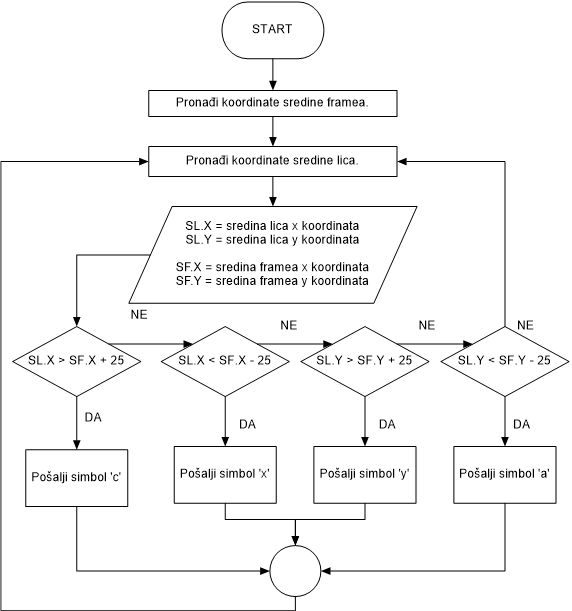


**Slika 3.7.** Algoritam detekcije i praćenja lica

Algoritam detekcije i praćenja lica se sastoji od 4 *while()* petlje, jedna glavna i 3 sporedne koje se nalaze unutar glavne petlje. Prije ulaska u glavnu petlju, inicializiraju se varijable potrebne za obradu slike i rukovanje događajima, klika srednje, lijeve i desne tipke miša. Prva unutarnja *while* petlja sprema *frame* kamere, prikazuje ga i omogućava korisniku da naredi što bolju masku slike. Petlja se izvodi sve dok se ne pritisne lijeva tipka miša. Nakon pritiska lijeve tipke miša, počinje se izvoditi druga po redu unutarnja petlja. Druga *while* petlja sprema *frame* kamere, obrađuje ga i primjenjuje algoritam detekcije lica koji je objašnjen u poglavlju 2.2. Prepoznavanje lica – Metode. Petlja se izvodi sve dok lice nije pronađeno, nakon što se lice pronađe, izračuna se histogram slike lica i uđe se u treću petlju. Treća *while* petlja računa *back projection* za histogram slike lica i *frame* kamere sve dok ne dođe do pogreške ili dok se ne stisne srednja tipka miša. Ukoliko se stisne srednja tipka miša, algoritam se vraća na početak, tj. ponavlja se prva *while* petlja i podešavanje maske slike. Ako se stisne desna tipka miša, algoritam svakih 5 sekundi računa novi histogram slike lica i na osnovi novoga histograma i *framea* kamere računa *back projection*. Nakon što se izračuna *back projection* slike, primjenjuje se metoda *Camshift* koja kvadratom označi regiju slike na kojoj se treba nalaziti lice.

### Realizacija algoritama detekcije i praćena ljudskog lica

Algoritam detekcije i praćenja ljudskoga lica je realiziran povezivanjem osobnog računala i Arduino sustava serijskom vezom. Osobno računalo izvodi, u predhodnom poglavlju opisani algoritam i ovisno o trenutnoj poziciji ljudskoga lica šalje informacije Arduinu. Arduino na osnovu primljenih informacija pali/gasi servo motore. Povezivanje osobno računala odnosno C++ koda i Arduino sustava omogućila je biblioteka *SerialClass.h* [7]. Slika 3.7. prikazuje dijagram toka slanja podataka na Arduino.



**Slika 3.8.** Slanje informacija o položaju lica na Arduino.

Dio C++ koda koji predstavlja algoritam slanja podataka na Arduino nalazi u trećoj *while* petlji. Prvo se pronalaze x i y koordinate sredine *framea*, nakon toga se pronalaze x i y koordinate lica. Ovisno o postavljenim uvjetima šalju se simboli na Arduino. Kako bi se izbjeglo nekontrolirano pomicanje kamere u sredinu *framea* kamere postavljen je kvadrat od 25 *pixela*. Ako se lice nalazi unutar toga kvadrata kamera se ne pomiće tj. servo motori miruju. C++ kod opisanoga algoritma je sljedeći:

if (centroid.x > (centarOkvira.x + 25)) SerialPort->WriteData("c", 4);

else if (centroid.x < (centarOkvira.x - 25)) SerialPort->WriteData("x",4);

if (centroid.y > (centarOkvira.y + 25)) SerialPort->WriteData("y", 4);

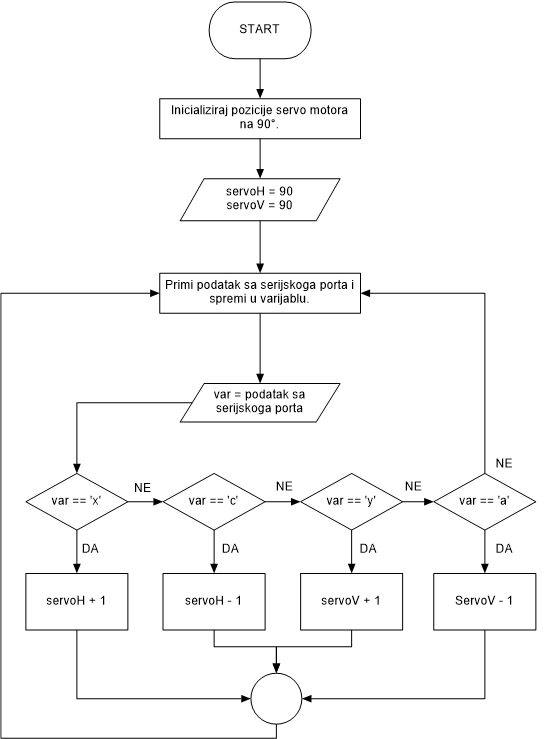
else if (centroid.y < (centarOkvira.y - 25)) SerialPort->WriteData("a", 4);

### Upravljanje sustavom pozicioniranja kamere

Sustav pozicijoniranja kamere realiziran je pomoću Arduino sustava i njegovih biblioteka. Za upravljanje servo motorima korištena je biblioteka „Servo.h“ i njene funkcije:

* **attach() -** Pridružuje *Servo varijablu* pinu,
* **write()** - Piše vrijednosti za servo motor, kotrolira osovinu u skladu s tim,

Pozicioniranje servo motora ovisi o trenutnoj poziciji detektiranoga lica u odnosu na centar *framea.* Ako se sredina lica nalazi u sredini *framea*, servo motori miruju, u suprotnome se servo motori pokreću i pomiču kameru tako da detektirano lice bude u sredini *framea.* Ideja algoritma upravljanja servo motorima je sljedeća, arduino prima simbole sa računala preko serijske veze. Svaki simbol određuje koji će se servo motor pomaknuti za plus ili minus 1 stupanj. Slika 3.8. prikazuje dijagram toka pozicioniranja kamere.



**Slika 3.9.** Blok dijagram inkrementalnoga pozicioniranja.

Ardunio u *setup()* petlji otvara COM3 serijski port na 9600 bauda i postavlja servo motore na 90 stupnjeva. U *loop()* petlji prima podatke sa računala i obrađuje ih.

Obrada podataka je sljedeća: u prvom se IF-bloku uspoređuju podaci koji su primljeni. Ovisno o primljenome simbolu servo motor se inkrementalno podešava. Simboli su odabrani proizvoljno i svaki od njih predstavlja jednu od četiri pozicija u kojemu se može nalaziti lice.

if (readInput.charAt(0) == 'x') startPositionHorz += 1;

else if (readInput.charAt(0) == 'c') startPositionHorz -= 1;

else if (readInput.charAt(0) == 'y') startPositionVert += 1;

else if (readInput.charAt(0) == 'a') startPositionVert -= 1;

Drugi IF-blok provjerava da li se servo motor nalazi u dopuštenome području zakretanja. Zbog sigurnosnih razloga, kako ne bi došlo do mogućih oštećenja postavljen je ovaj IF-blok. Servo motori koji su korišteni u ovome projektu mogu se zakretati od 0 do 180 stupnjeva.

if (startPositionHorz < 180 && startPositionHorz > 0 ) horz.write(startPositionHorz);

else if(startPositionHorz >= 178) horz.write(180);

else if(startPositionHorz <= 2) horz.write(0);

if (startPositionVert < 180 && startPositionVert > 0) vert.write(startPositionVert);

else if(startPositionVert >= 178) vert.write(180);

else if(startPositionVert <= 2) vert.write(0);

## Ekonomska bilanca izgradnje rješenja

Komponente koje su kupljene: Arduino, stalak za kameru i servo motore, te dva servo motora. Kameru smo posudili od kolege, a računalo i protoboard smo posjedovali od prije tako da nismo morali uložiti novac u te komponente. Ukupni troškovi prikazani su u tablici.

|  |  |  |  |
| --- | --- | --- | --- |
| **Komponenta** | **Komada** | **Cijena/kom (kn)** | **Cijena (kn)** |
| Arduino | 1 | 86,03 | 86,03 |
| Stalak za kameru | 1 | 24,70 | 24,70 |
| Kamera | 1 | 0 | 0 |
| Računalo | 1 | 0 | 0 |
| Protoboard | 1 | 0 | 0 |
| Servo motor | 2 | 33,52 (zajedno 2 kom) | 33,52 |
| **Ukupno** |  |  | **144,25** |

**Tablica 3.3.** Pregled financijskih troškova.

# Ispitivanje i evaluacija sustava

U sljedećim poglavljima su opisane metode i dane su karatkeristike algoritma za praćenje lica. Važno je napomenitu da maska cijelog *framea* nije primjenjivana na *backprojection* sliku kako bi se što bolje vidjele razlike uspoređivanih metoda.

## Postupci, metode i kriteriji ispitivanja i evaluacije

Ispitana su dva primjera realizacije programa. Prvo je ispitana realizacija algoritma koja bi uzimala u obzir jednokanalni ili dvokanalni histogram. Te je nakon toga ispitana realizacija algoritma u kojemu se koristi dinamični ili statični histogram.

Kod prvoga ispitivanja, svakih pet *framea* smo spremali backprojection sliku jednokanalnog i dvokanalnoga histograma te sliku framea na kojoj je kvadratom označeno lice za svaku od backprojection slika pojedinačno. Promatrali smo koliko dobro pojedinačni kvadrati opisuju lice te koliko šuma imaju i koje kvalitete su backprojection slike. Boljom metodom smo proglasili onu koja je u backprojection slici ima manje šuma i koja u konačnici bolje opisivala lice kvadratom.

Drugo ispitivanje smo provodili svakih pet sekundi nakon što bi upalili metodu dinamičnoga histograma (desni klik miša). Spremali smo backprojection sliku statičnog i dinamičnoga histograma te sliku *framea* na kojoj je kvadratom označeno lice. Promatrali smo ponašanje kvadrata koji predstavlja svoju metodu računanja histograma i backprojection slike pojedinačno. Metoda koja bi imala bolju backprojection sliku i ona koja bi bolje opisivala lice kvadratom, proglasili smo boljom.

## Usporedba praćenja sa jednokanalnim i dvokanalnim histogramom

Histogram se računa za sliku lica koja je u HSV prostoru boja. Jednokanalni histogram sadrži Hue kanal slike, dok dvokanalni histogram sadrži Hue i Saturation kanale. Na sljedećim slikama su prikazani *backprojection* slike za navedene histograme i slika *framea.* Slika je spremana svakih 50 *frameova*.

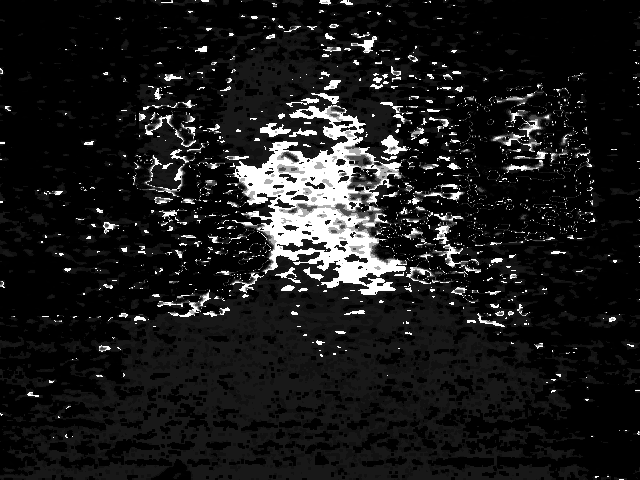
   
Frame 0 Frame 50

   
Frame 100 Frame 150

   
Frame 200 Frame 250  
**Slike 4.1.** Prikazuju *backprojection* za dvokanalni histogram.

   
Frame 0 Frame 50

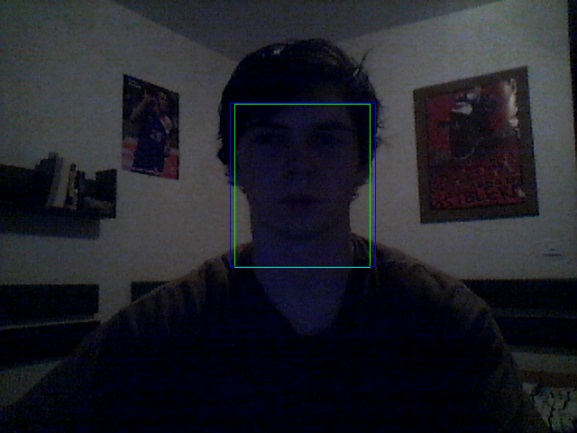
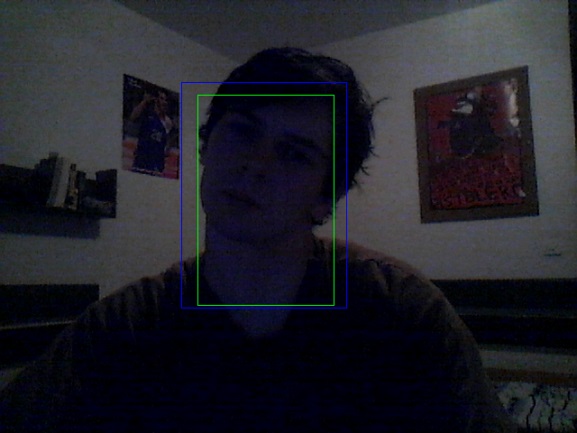
   
Frame 100 Frame 150

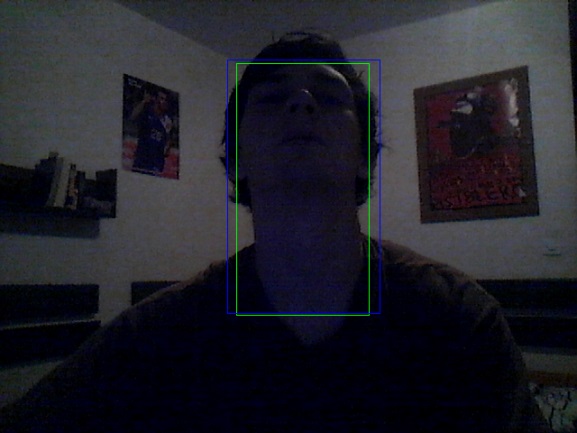
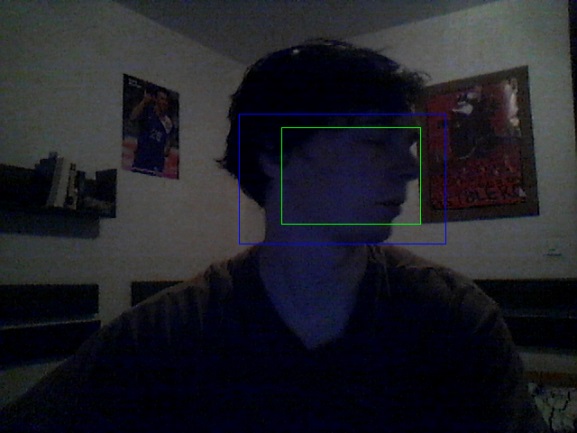
Frame 200 Frame 250

**Slike 4.2.** Prikazuju backprojection za jednokanalni histogram.

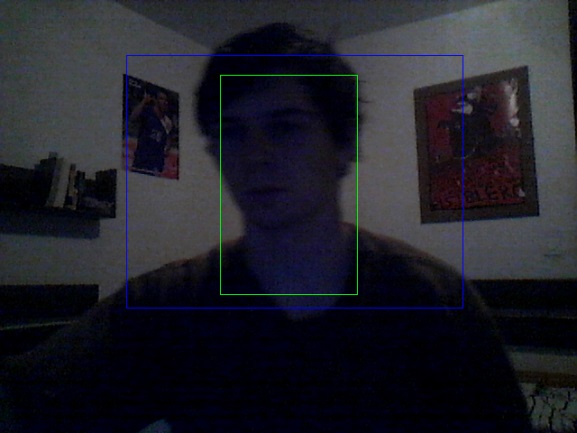
Na slikama *framea*, kvadratom je označeno lice, **zeleni kvadrat** predstavlja dvokanalni histogram, a **plavaki kvadrat** jednokanalni histogram.

Frame 0 Frame 50

Frame 100 Frame 150

Frame 200 Frame 250

**Slike 4.3.** Prikazuju frame slike za gore prikazane back projection slike.

Iz gore pokazanih slika možemo zaključiti da je dvokanalni histogram bolja opcija za praćenje lica od jednokalanoga histograma. Backprojection slike za dvokanalni histogram imaju znatno manje šuma koji utječe na bolje označavanje lica.

## Usporedba praćenja sa dinamičkim i statičkog histogramom

Na poćetku pračenja lica, histogram je isti za obje metode. Svakih pet sekundi metoda temeljena na dimaničkome histogramu računa novi histogram slike lica, dok statični histogram ostaje onaj početni. Sljedeće slike prikazuju backprojection slike za dinamički histogram (Slike 4.4.), statični histogram (Slike 4.5.) i frame slike (Slike 4.6.) na kojoj je kvadratom označeno lice za svaku od metoda praćenja.

Frame u 0 sekundi Frame u 5 sekundi

Frame u 10 sekundi Frame u 15 sekundi

Frame u 20 sekundi Frame u 25 sekundi

**Slike 4.4.** Prikazuju backprojection za metodu praćenja sa promjenjivim histogramom.

Frame u 0 sekundi Frame u 5 sekundi

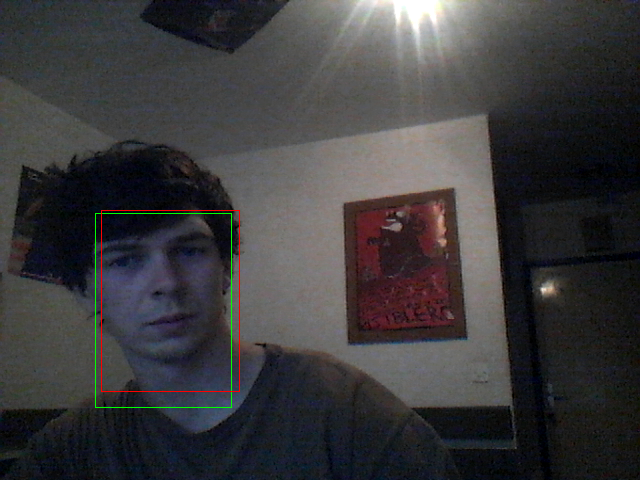
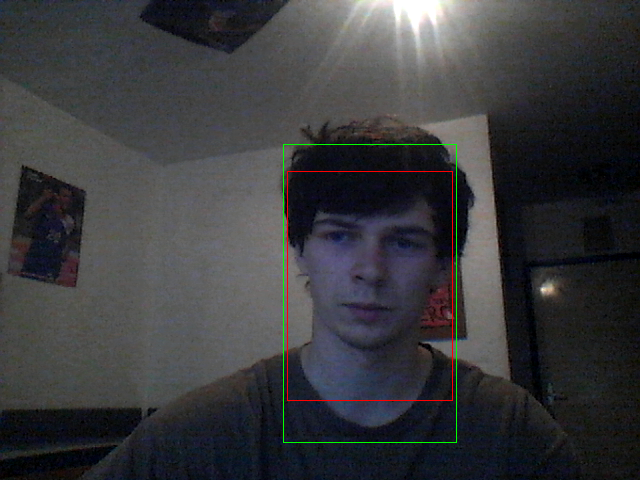
Frame u 10 sekundi Frame u 15 sekundi

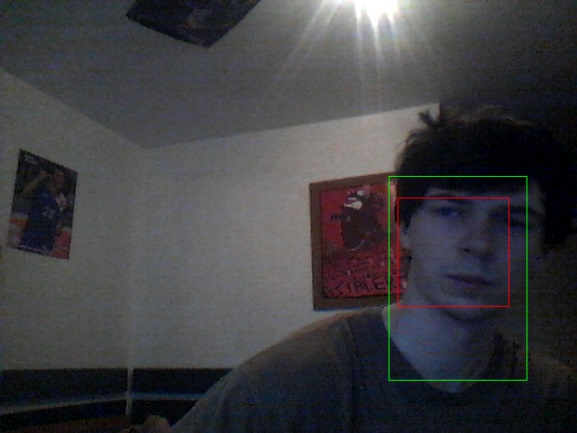
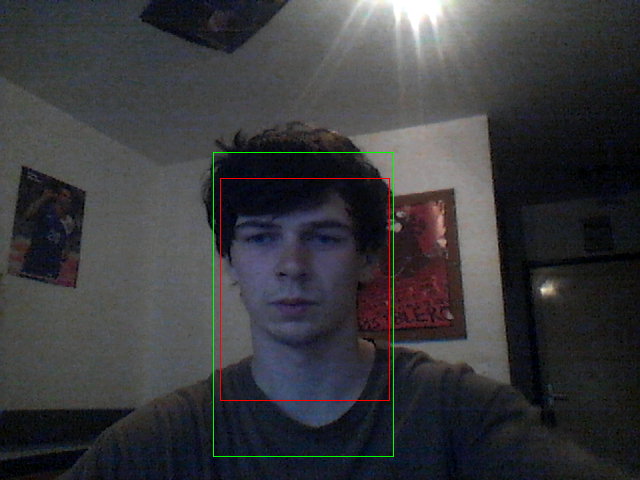
Frame u 10 sekundi Frame u 15 sekundi

**Slike 4.5.** Prikazuju backprojection sliku za nepromjenjivi histogram

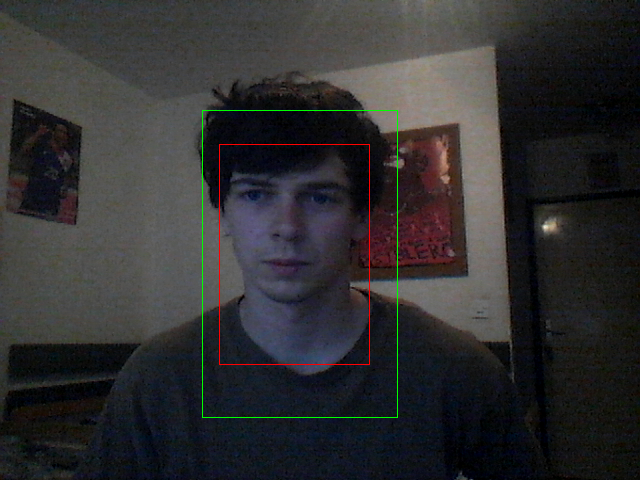
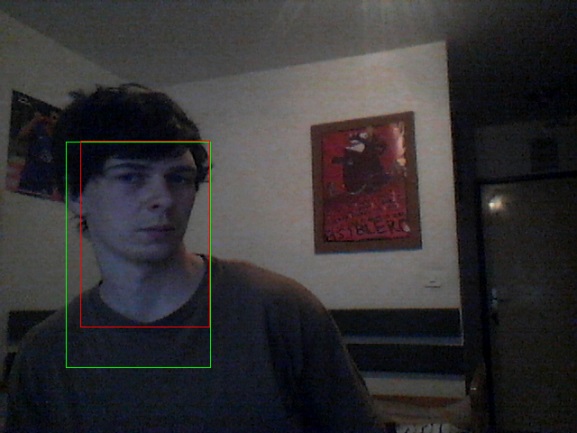
Sljedeće slike prikazuju frame slike na kojoj je označena kvadratom svaka od metoda pračenja. Crvenim kvadratom predstavljena je metoda statičnoga histograma, a zelenim kvadratom metoda dinamičnoga histograma

Frame u 0 sekundi Frame u 5 sekundi

Frame u 10 sekundi Frame u 15 sekundi

Frame u 20 sekundi Frame u 25 sekundi

**Slike 4.6.** Prikazuju frame slike za gore prikazane back projection slike.

Iz gore pokazanih slika, možemo vidjeti da kvadrat metode statičnoga histograma bolje opisuje lice, ali predpostavljamo da nije otporan na promjene inteziteta svjetla koje pada na lice. Dok dinamički histogram malo lošije opisuje lice, ali ostaje u dozvoljenim granicama. Teoretski gledano, možemo predpostaviti da bi metoda dinamičkoga histograma bila otporna na promjene inteziteta svjetlosti koje pada na lice.

# Zaključak

Nedugo nakon predstavljanja algoritma za detekciju objekata Paula Viole i Michaela Jonesa, pojavile su se mnoge realizacije praćenja koje su koristile taj algoritam. U ovome seminaru je opisan algoritam koji za detekciju objekta koristi algoritam Viole-Jonesa, dok za praćenje objekta koristi metode opisane u seminaru. Cilj je bio napraviti algoritam koji bi pratio objekt bolje nego realizacija praćenja koja koristi samo algoritam Viole-Jonesa. Ispitivanjem algoritma došli smo do sljedećih zaključaka. Algoritam praćenja bolje prati objekt ako se koristi dvokanalni histogram boja objekta, te ako koristimo statički ili dinamički način računanja histograma. Statičko računanje histograma je bolje za uvjete gdje se ne mjenja osvjetljenje objekta, dok je dinamičko računanje histogram bolji za slučajeve gdje se osvjetljenje objekta mjenja. Usprkos svemu, algoritam nije savršen, još uvijek postoji prostora za njegovo unapređivanje.

# Literatura

[1] Paul Viola i Michael Jones, “Rapid Object Detection using a Boosted Cascade of Simple Features”, 2001, <https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf>

[2] Backprojection, [http://docs.opencv.org/doc/tutorials/imgproc/histograms/back\_proje ction/back\_projection.html](http://docs.opencv.org/doc/tutorials/imgproc/histograms/back_proje%20ction/back_projection.html)

[3] Camshift, [http://docs.opencv.org/trunk/doc/py\_tutorials/py\_video/py\_meanshift/py\_m eanshift.html](http://docs.opencv.org/trunk/doc/py_tutorials/py_video/py_meanshift/py_m%20eanshift.html)

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

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

[6] OpenCv, <http://opencv.org/>

[7] Interfacing Arduino and C++ (for Windows), <http://playground.arduino.cc/Interfacing/> [CPPWindows](http://playground.arduino.cc/Interfacing/)

# Prilozi

Dokumenti:

* EA seminar.docx
* EA seminar.pdf

Datoteke:

* EA project – 19.06.2014

# Elektronička verzija