UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET
ODSJEK ZA AUTOMATIKU I ELEKTRONIKU

Korištenje irisa oka kao virtuelnog pokazivača

ZAVRŠNI RAD

- PRVI CIKLUS STUDIJA -

Student:
Amila Sabljica

Mentor:
Doc. dr Emir Sokić, dipl. ing. el.

Sarajevo,
septembar 2019.

Sažetak

Korištenju oka u interakciji sa računarom je posvećeno mnogo pažnje, stoga je veliki broj metoda razvijen za praćenje kretanja oka i korištenje dobijenih informacija. Međutim, većina istih je ograničena ambijentalnim uslovima, facijalnom ekspresijom korisnika, i korištenjem invazivnih metoda koje mogu utjecati na čovjekovo zdravlje. U ovom radu je pokušano zaobići takve pristupe, te se bazirati na pristupu koji će omogućiti korisniku što jednostavnije korištenje uređaja za praćenje pogleda. Ciljana grupa su ljudi sa kognitivnim poteškoćama, stoga je i razvoj uređaja prilagođen toj namjeni. Korištenje neinvazivne metode praćenja pogleda se zasniva i oslanja na obradu slike, stoga je opisan način određivanja korištenog hardvera, te nadalje metode obrade slike, ekstrakcije područja od interesa, te estimacija irisa oka na osnovu rezultata obrađene slike. Dati su rezultati svakog od korištenih pristupa, te osvrt na korištene metode, moguća poboljšanja i rješenja problema.

Ključne riječi: praćenje pogleda, intereakcija sa računarom, iris oka, obrada slike, regija od interesa

Abstract

Using an eye in interaction with computers has attracted a lot of attention, so plenty of different methods were developed for that use. Most of those are restricted by ambient conditions, user's facial expression, and using invasive methods that can affect human health. This project strives to develop an approach that will avoid mentioned methods, and focus on creating a device that will be as simple as possible for using as an eye tracker. The target group are people with cognitive disorders, so the device was developed bearing that in mind. Using noninvasive method for eye tracking is based on image processing, which is explained in this work, along with the way of deciding what type of hardware should be used, extraction of region of interest, and estimation of iris position based on processed image. Results and review of used approaches are given, as well as possible improvements and solutions of existing problems.

Keywords: eye tracking, computer interaction, iris of an eye, image processing, region of interest

Elektrotehnički fakultet, Univerzitet u Sarajevu
Odsjek za automatiku i elektroniku
Doc. dr Emir Sokić, dipl.el.ing
Sarajevo, 19.12.2018.

Završni rad I ciklusa studija

Korištenje irisa oka kao virtuelnog pokazivača

Sažetak:
Za osobe koje su privremeno ili trajno u potpunosti motorički hendikepirane, najjednostavniji zadaci kao što su pomjeranje ruke, klik na dugme ili izgovor kratke riječi može predstavljati veliku poteškoću. Kako bi se olakšala komunikacija ljekara/porodice sa osobama sa poteškoćama, jedna od mogućnosti je korištenje oka kao "virtuelnog" pokazivača. U ovom radu će biti potrebno istražiti mogućnost detekcije, lokalizacije i praćenja irisa oka, kako bi se mogli sintetizirati odgovarajući signali za pokretanje i upravljanje virtuelnih markera na ekranu mobilnog telefona, tableta ili računara. Ovo bi u konačnici omogućilo da se pomjeranje oka hendikepiranih osoba iskoristi kao "virtuelni" uređaj za upravljanje odgovarajućim aplikacijama na računarskom sistemu.

Koncept i metode rješavanja:
Rad treba da se sastoji iz sljedećih cjelina:

- pregleda literature,
- analize metoda detekcije i lokalizacije irisa oka,
- implementacije odgovarajućih algoritama u C++/OpenCV okruženju,
- analize eksperimentalnih rezultata.

Polazna literatura:

[1] Batchelor, Bruce G., and Paul F. Whelan. Intelligent vision systems for industry. Springer Science & Business Media, 2012.
[2] Torras, Carme, ed. Computer vision: theory and industrial applications. Springer Science & Business Media, 2012.
[3] Demant, Christian, C. Demant, and Bernd Streicher-Abel. Industrial image processing. Springer-Verlag, 1999.
[4] Szeliski, Richard. Computer vision: algorithms and applications. Springer Science & Business Media, 2010.
[5] Sonka, Milan, Vaclav Hlavac, and Roger Boyle. Image processing, analysis, and machine vision. Cengage Learning, 2014.
[6] Russ, John C. The image processing handbook. CRC press, 2016.
[7] Costa, Luciano da Fontoura Da, and Roberto Marcondes Cesar Jr. Shape analysis and classification: theory and practice. CRC Press, Inc., 2000.

[8] Sohail Maryam, Geelani M. Nafees, Ghani M. Usman and Chaudhry Sarah. GazePointer: A Real – Time Mouse Pointer Control Implementation Based on Eye Gaze Tracking. Comsats Institute of Information Technology, 2012.
[9] Enrique Cáceres, Miguel Carrasco, Sebastián Ríos. Evaluation of an eye-pointer interaction device for human-computer interaction. Heliyon 4, 2018.

Doc. dr Emir Sokić, dipl. ing. el.

Univerzitet u Sarajevu
Elektrotehnički fakultet
Odsjek za automatiku i elektroniku

Izjava o autentičnosti radova

Završni rad
I ciklusa studija

Ime i prezime: Amila Sabljica
Naslov rada: Korištenje irisa oka kao virtuelnog pokazivača
Vrsta rada: Završni rad Prvog ciklusa studija
Broj stranica: 62

Potvrđujem:

- da sam pročitao dokumente koji se odnose na plagijarizam, kako je to definirano Statutom Univerziteta u Sarajevu, Etičkim kodeksom Univerziteta u Sarajevu i pravilima studiranja koja se odnose na I i II ciklus studija, integrirani studijski program I i II ciklusa i III ciklus studija na Univerzitetu u Sarajevu, kao i uputama o plagijarizmu navedenim na web stranici Univerziteta u Sarajevu;
- da sam svjestan univerzitetskih disciplinskih pravila koja se tiču plagijarizma;
- da je rad koji predajem potpuno moj, samostalni rad, osim u dijelovima gdje je to naznačeno;
- da rad nije predat, u cjelini ili djelimično, za stjecanje zvanja na Univerzitetu u Sarajevu ili nekoj drugoj visokoškolskoj ustanovi;
- da sam jasno naznačio prisustvo citiranog ili parafraziranog materijala i da sam se referirao na sve izvore;
- da sam dosljedno naveo korištene i citirane izvore ili bibliografiju po nekom od preporučenih stilova citiranja, sa navođenjem potpune reference koja obuhvata potpuni bibliografski opis korištenog i citiranog izvora;
- da sam odgovarajuće naznačio svaku pomoć koju sam dobio pored pomoći mentora i akademskih tutora/ica.

Sarajevo, septembar 2019.

Potpis:

Amila Sabljica

Sadržaj

Popis slika

Popis tabela

Poglavlje 1

Uvod

Sa izumom računara sredinom prošlog stoljeća, došlo je do potrebe kreiranja interfejsa za upravljanje istim. U početku je korišten teletajp, da bi sa naglim napretkom u oblasti računarske tehnologije došlo do potrebe kreiranja naprednijih upravljačkih interfejsa. Upotreba računara je postala svakodnevnica za veliki dio populacije, stoga je bilo potrebno razviti različite načine korištenja računara obzirom da se mogućnosti korisnika razlikuju od korisnika do korisnika. Uz sve mogućnosti koje danas računari mogu ponuditi, ljudi i njihova interakcija sa računarima su postali limitirajući faktor, što je dalo povoda za istraživanjem u polju *čovjek-računar interakcije* (*engl. human computer interaction*, u nastavku teksta HCI). Glavni zadatak HCI-a je omogućavanje lakše, jednostavnije, intuitivnije i efikasnije interakcije čovjeka sa računarom. Ograničenja u interakciji koja su prije postojala, a koja su se bazirala samo na tastaturama i printerima, sada su prevaziđena. Različite vrste pokazivačkih uređaja, površina osjetljivih na dodir, displeja velikih rezolucija, mikrofona i zvučnika su postali sastavni dio uređaja za interakciju sa modernim računarima. Sve ovo je dovelo do toga da danas interakciju sa računarom možemo ostvariti i putem govora, gestikulacije ili pomoću različitih objekata sa senzorima. Jedan od načina HCI-a je i korištenje pogleda za interakciju čovjeka sa računarom, što će u nastavku biti razmatrano. [4]

1.1 Motiv za korištenje pogleda u HCI

Da bi uopće bilo razmatrano korištenje praćenja pogleda (*engl. eye tracking*) u HCI-u, potrebno je prvo shvatiti motivaciju za snimanje ljudskog pokreta očima. Ono u što osoba gleda se pretpostavlja da indicira misao koja je ”na vrhu gomile” (*engl. on top of the stack*) koginitvnih procesa ([5]). Ljudi pokreću oči da bi što preciznije mogli vidjeti ono što se nalazi u centralnom dijelu njihovog pogleda. Također, često skreću pažnju na tu tačku kako bi se mogli skoncentrisati (makar i na trenutak) na objekat ili regiju od interesa. Upravo iz tog razloga se pretpostavlja da je iz praćenja ljudskog pogleda moguće zaključiti koja je putanja pažnje posmatrača. To također može dati informaciju šta posmatrač smatra interesantnim, odnosno, šta je odvratilo njegovu pažnju, te možda čak i kako je osoba percipirala sliku koju je gledala. [6]

Dio populacije ostvaruje interakciju sa računarom cjelodnevno, kako za potrebe posla, tako i u slobodno vrijeme. Kako je većina te interakcije ostvarena sa tastaturom i mišem, uz korištenje ruku, to može dovesti do toga da neki ljudi pate od prenaprezanja određenih dijelova ruku, što obično rezultuje sindromom karpalnog tunela. Kako korištenje računara postaje sveprisutno, dolazi do potrebe za razvijanjem drugih metoda interakcije između čovjeka i računara, koje će stvarati manje fizičkih problema. Oči su dobar kandidat obzirom da se svakako pokreću koište-

njem računara, stoga se stvara mogućnost korištenja pokreta očiju kako bi se dobila informacija koja bi u određenoj mjeri nadoknadila interakciju baziranu na korištenju ruku, pogotovo onda kada korištenje ruku nije uopće moguće zbog određenih fizičkih problema. [4]

Pored problema koji mogu nastati svakodnevnim i dugotrajnim korištenjem računara, a koji se odražavaju na fizičke probleme ruke, javlja se problem komunikacije sa računarom kod osoba čije su motoričke sposobnosti u velikoj mjeri oslabljene ili u potpunosti onesposobljene. Za pacijente sa određenim stepenom motoričkih poteškoća, kao što su amiotrofna lateralna skleroza, totalna paraliza, ili ekstrapiramidalni sindrom (bolest motornog neurona), a koji imaju normalne kognitivne sposobnosti, pogled je najbolja dostupna opcija za komunikaciju. Nedostatak mehanizma koji pomenutim pacijentima omogućava da pišu i lako čitaju, između ostalog, predstavlja veliku barijeru u pristupu znanju i smanjuje njihove šanse za samostalnost i lični razvoj.[2]

Iako je fokus ovog rada na izradi interfejsa za HCI za osobe sa ozbiljnim motoričkim poteškoćama, testiranje je izvođeno na osobama koje imaju prosječne vizuelne, kognitivne i motorne vještine. Razlog tome je u činjenici da razvoj rada nije bio ograničen na testiranje na osobama sa motoričkim poteškoćama, iako su one uzete u obzir prilikom projektovanja rješenja problema. Motoričke sposobnosti ne utječu u velikoj mjeri na samo rješavanje problema, osim što postavlja određena ograničenja koja su uzeta u obzir.

1.2 Cilj rada

Cilj rada je kreiranje funkcionalnog uređaja za komunikaciju sa računarom korištenjem ljudskog pogleda. U postupku izrade, pažnja je posvećena nekim osnovnim zahtjevima:

- uređaj treba koristiti neinvazivnu metodu za praćenje ljudskog pogleda,
- planirano je da uređaj koriste osobe sa ozbiljnim ili parcijalnim motoričkim poteškoćama, stoga je potrebno da uređaj ne zahtijeva korištenje motoričkih sposobnosti,
- uređaj treba biti prilagođen korištenju osoba različitih fizičkih propozicija, stoga treba biti prilagodljiv,
- preciznost i ponovljivost metode praćenja pogleda kod uređaja trebaju biti zadovoljavajuće,
- cijena uređaja treba biti ško prihvatljivija kako bi mogla parirati već postojećim skupim rješenjima,
- potrebno je da uređaj što manje bude ograničen na određene ambijentalne uslove,
- uređaj treba u što većoj mjeri biti otporan na pokrete glavom,
- uređaj treba biti siguran za korištenje od strane djece,
- obzirom da se uređaj pravi za čovjek-računar interakciju, potrebno je obezbijediti što veći dijapazon tipova računara s kojim će uređaj moći funkcionirati ispravno,
- uređaj ne treba stvarati nelagodu kod osoba koji ga koriste.

Pored zahtjeva navedenih gore, postavljaju se i zahtjevi tehničke prirode, koji će biti obrađeni tokom izlaganja realizacije rješenja.

Poglavlje 2

Praćenje pogleda

U prethodnih 30 godina, mnoge nove tehnologije koriste ljudski pogled kao ulaznu informaciju za računare i kontrolne sisteme. Spomenuta tehnologija je razvijana u različitim područjima, kao što su zdravstvene usluge, sigurnost i biometrika, interfejsi, interaktivni sistemi, neuroznanost i kognitivnost. [2] Kao što je spomenuto, praćenje pogleda ima posebnu veliku primjenu kod osoba sa određenim motoričkim poteškoćama, time im olakšavajući korištenje kursora i/ili virtualne tastature, pružajući pacijentima alternativni pistup tokom integracije i rehabilitacije.

Uprkos razvoju, postoji nekoliko faktora koji sprečavaju preciznost u ovakvim sistemima, kao što je npr. nedostatak standarda za kvalitetu podataka prikupljenim uređajima baziranim na praćenju pogleda. [2] Posebno je bitna tačna evaluacija tačaka od interesa ili pogleda posmatrača, obzirom da iste moraju biti precizno predviđene kako bi mogao biti određen prostor u koju posmatrač gleda. Sposobnost predviđanja ovisi od tipa aplikacije koja se razvija. [2]

2.1 Pozadina praćenja pogleda

Zahvaljujući velikom napretku tehnologije praćenja pogleda, moguće je usavršiti razumijevanje ljudskog pogleda u poljima kao što su psihologija, dizajn proizvoda, biologija, kognitivna-neuronauka, i računarski vid. Jedna od glavnih upotreba praćenja pogleda (*engl. eye tracking*) tehnologije je detekcija pokreta oka kako bi se konvertovala pozicija oka u pokretne obrasce. Jedna od najčešćih primjena ovih uređaja je u prikupljanju informacija za statističke svrhe, bilo da je to za određivanje glavne tačke gledišta osobe koja gleda u bilbord ili neki drugi objekat, ili za snimanje fokusa osobe kada gleda prema nečemu konkretnom. Mnogo kompleksnija aplikacija koja mnogo dobija na značaju, je istraživanje osoba-računar interakcije (HCI) u smislu korištenja eye tracking uređaja za interakciju, kroz novorazvijene platforme koje omogućavaju korištenje određenih aplikacija uz pomoć eye tracking uređaja ili direktno inkorporirajući iste u već postojeće operativne sisteme računara ili mobilnih uređaja. [2]

Koncept praćenja pogleda se odnosi na set tehnologija i postupaka koji omogućavaju praćenje pogleda osobe koja fiksira svoju vizuelnu pažnju na određenu scenu ili sliku.

Važno je razlikovati dva glavna pristupa u izvođenju observacije [2]:

- mjerenje pozicije i orijentacije oka u odnosu na glavu,
- određivanje orijentacije oka u odnosu na prostor.

Određivanje orijentacije oka u odnosu na prostor je poznato i kao ”tačka gledišta” (*engl. point of regard*).

Razvijene su četiri tehnike za mjerenje pokreta oka [2]:

- elektrookulografija (*engl. electrooculography (EOG)*),
- korištenje usisnih čaša ili kontaktnih sočiva,
- korištenje foto ili video okulografije,
- video detekcija zasnovana na odsjaju zjenice i rožnjače.

Danas postoje tri vrste uređaja na bazi eye trackinga koji koriste kombinacije nekih od navedenih pristupa [2]:

- uređaji kod kojih je glava poduprta i uvijek je u nepomičnom položaju,
- uređaji koji su pričvršćeni za glavu korištenjem neke vrste kacige ili sočiva, dopuštajući pokretanje glave,
- uređaji koji uključuju i pokrete lica (praćenje lica) kako bi bilo moguće kretati glavu bez kalibracionih gubitaka.

Bez obzira na tip, osnovna funkcija ovih uređaja je snimanje slika proizvedenih optičkim odsjajem prvog sloja rožnjače, i korištenje vektora formiranog između centra zjenice i centra reflektovanih tačaka dobijajući tačne pozicije zjenice za eye tracker. Ovaj vektor, a ne apsolutna pozicija zjenice, je korišten kao vrijednost jer je invarijantan u odnosu na nenamjerne pokrete uređaja ili glave. Kalibracijom ovi uređaji mogu detektovati tačku u koju korisnik gleda. [2]

Na slici 2.1 je prikazan primjer različitih eye tracking uređaja.

Slika 2.1: Primjer različitih eye tracking uređaja ([1])

Poglavlje 3

Osnove praćenja usmjerenosti pogleda

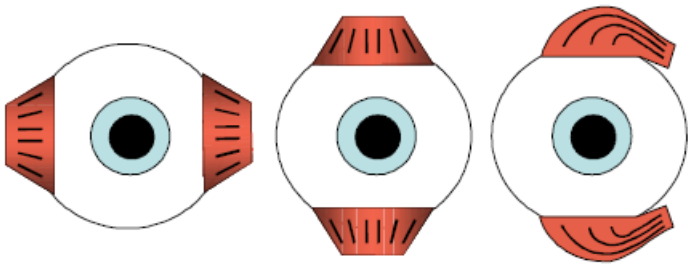
U prethodnim godinama, mnoge tehnologije su razvijene za potrebe korištenja ljudskog oka, odnosno pogleda, kao ulazne informacije za računar. Posebna pažnja je posvećena razvijanju uređaja za osobe sa određenim motoričkim poteškoćama što se pokazalo korisnim, obzirom da je na taj način olakšano korištenje kursora miša i/ili virtuelne tastature na alternativni način tokom procesa prilagođenja ili rehabilitacije. [2][1]

3.1 Oko i pokreti oka

Da bi oko uopće moglo biti razmatrano za korištenje u ulozi virtuelnog pokazivača, potrebno je razumjeti kako oko, tako i uređaj za praćenje oka. U ovom poglavlju će biti razmatrana anatomija oka, te mogući problemi u kalibraciji i tačnosti.

3.1.1 Anatomija i pokreti oka

U terminima inženjeringa, oko i njegovi mišići se mogu posmatrati kao kamera sa stabilizacijom slike. Šest mišića spajaju oko sa glavom, i to je prikazano na slici 6.1. Mišići su organizovani u parove tako da svaki par oku pruža jedan stepen slobode, čineći da oko ima ukupno 3 stepena slobode. Jedan par je odgovoran za horizontalno pomijeranje, drugi za vertikalno, dok treći omogućava rotacione pokrete oko pravca pogleda. Na ovaj način je obezbijeđena kompenzacija pokretanja glave u slučaju da je okom moguće gledati isključivo u jednom pravcu. Nervi koji kontrolišu očne mišiće i omogućavaju pokretanje oka su usko vezani za organ zaslužen za ravnotežu koji se nalazi u uhu. [4]



Slika 3.1: Tri para mišića oka koji kompenzuju pomijeranje glave [Drewes, 2010, str. 33]

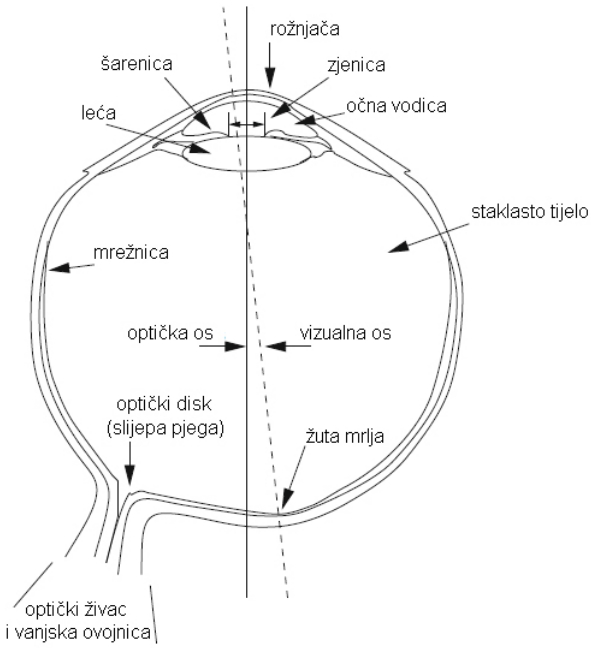
Oko često dobija naziv ”najgora kamera na svijetu” obzirom da ”pati” od nekoliko optičkih nesavršenosti, kao što su:

- sferična odstupanja: efekat prizme koji stvaraju periferalni dijelovi leće,
- hromatska odstupanja: kraće talasne tužine (plava) se prelamaju više od dužih talasnih dužina (crvena),
- zakrivljenje vidnog polja: ravni objekti dovode do zakrivljene slike.

S druge strane, oko posjeduje nekoliko mehanizama koji smanjuju negativne efekte:

- da bi se smanjila sferična odstupanja, iris djeluje kao stop, time ograničavajući perferni ulazak svjetlosnih zraka,
- da bi se prevazišao problem hromatskog odstupanja, oko se obično fokusira na kreiranje izoštrenih slika srednjih talasnih dužina,
- da bi se uspješno prevazišao problem zakrivljenja vidnog polja, retina je zakrivljena time kompenzirajući posljedice ovog efekta. [6]

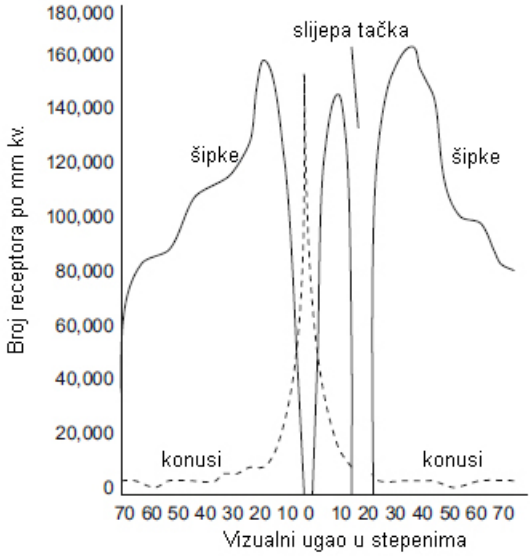
Oko je shematski prikazano na slici 3.2.



Slika 3.2: Oko adaptirano vizuelnoj percepciji [Duchowski, 2007, str. 19]

Kao što je spomenuto, konstrukcija oka je slična konstrukciji kamere. Oko posjeduje membranu, koja se naziva iris oka, i koja služi za postavljanje otvora blende. Otvor u irisu se naziva zjenica. Za razliku od kamere koja koristi sočiva sa fiksnom fokalnom dužinom i koja fokusiranje postiže mijenjanjem pozicije sočiva, sočivo oka može promijeniti fokalnu dužinu mijenjajući oblik sočiva. Dio osjetljiv na svjetlost se nalazi u zadnjem dijelu unutrašnjosti oka i naziva se retina. [4]

Kompletna površina osjetljiva na svjetlost je elipsoid sa horizontalnim produžetkom od 180° i sa vertikalnim produžetkom od 130°. Postoje dva različita tipa fotoreceptora - šipke i konusi. Šipke su osjetljive na osvijetljenost i imaju veliku osjetljivost te omogućavaju gledanje u mraku. Konusi su manje osjetljivi i detektuju hromatsko svjetlo. Šipke i konusi nisu jednako raspoređeni na retini. Veći dio retine ima veoma nisku gustoću fotoreceptora i samo mala kružna površina prečnika 1°do 2° koja se naziva fovea sadrži mnogo konusa. Fovea sadrži oko 150000 konusa po mm^2 i samo mali broj šipki. Van fovee, broj konusa pada na 20000 konusa po mm^2. Većina vidnog polja van fovee ima nisku rezoluciju i njen glavni zadatak je percepcija ambijetalnog kretanja. [6] Na lici 3.3 je prikazana gustoća raspodjele šipki i konusa, dok slika 3.4 prikazuje efekte te raspodjele fotoreceptora na sliku koju prenosi retina. [4]



Slika 3.3: Gustoća raspodjele šipki i konusa [Duchowski, 2007, str. 32]

Slika 3.4: Efekti raspodjele fotoreceptora na sliku koju prenosi retina [Drewes, 2010, str. 34]

Oči se većinom kreću na neobičan način u odnosu na ostale dijelove tijela. Prvenstveno, oči se kreću sinhronizovano. Oko treba stabilnu projekciju na retinu za prepoznavanje slika. Da bi se to ostvarilo, tri para mišića nadoknađuju pokrete glave, s tim što je to kretanje oka suprotno (glatkom) pokretu glave. Slično kretanje se pojavljuje i kad oko prati objekat i takvi pokreti su slični ostalim pokretima tijela. Ipak, kada se objekat ne kreće i glava stoji u stabilnoj poziciji, pogled se mijenja u naglim pokretima. Ovo ”skakutanje” nazivamo sakade. Sakade su brzi, simultani pokreti oba oka u istom smjeru koje kontrolira frontalni režanj mozga. Nakon sakada, oči jedno vrijeme miruju što se naziva fiksacija. Ostatak je vrijeme potrebno za stabilnu projekciju na retinu. Objašnjenje za pojavu sakada leži u veličini fovee. Fovea je mala reda veličine oko 1° vidni ugao, i ima mogućnost davanja slike velike rezolucije. Da bi bilo moguće vidjeti nešto jasno, oko se mora okrenuti prema objektu od interesa kako bi isti bio projektovan na foveu, a to sve čini brzi pokret sakada. Vrijeme mirovanja, fiksacija, je vrijeme potrebno za procesiranje slike, što znači da je mozgu potrebno vrijeme da prepozna objekat u koji je oko usmjereno. U ovisnosti od toga šta je mozak prepoznao, oči izvode ponovo sakadu prema drugom objektu od interesa. [4]

Poglavlje 4

Opis razvojnog rješenja

Rečeno je da postoje četiri tehnike za mjerenje pokreta oka, te tri vrste uređaja na bazi eye trackinga. Od pomenutih je bilo potrebno odabrati bazni pristup na kojem će biti bazirano rješenje. Za tehniku mjerenja pokreta oka odabran je pristup koji omogućava neinvazivnost u određivanju pozicije oka, odnosno, korištenje foto ili video okulografije, dok je za vrstu uređaja odabran princip koji omogućava da uređaj bude pričvrščen na glavu korištenjem neke vrste kacige ili sočiva, dopuštajući pokretanje glave, te je također uključeno i praćenje pokreta lica kako bi bilo moguće okretati glavu bez kalibracionih gubitaka. Korištenjem ovih pristupa bi bilo moguće ukloniti ili u određenoj mjeri smanjiti restrikcije koje se postavljaju inače u eye trackingu. [1]

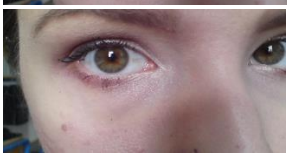
Obzirom da je odabrana neinvazivna metoda korištenjem video okulografije, bilo je potrebno osmisliti strukturu uređaja koji će omogućiti realizaciju pomenute metode. Iz tog razloga je izvedeno eksperimentalno snimanje oka u kontrolisanim uvjetima, te je iz rezultata snimanja donesena odluka o izgledu strukture uređaja.

4.1 Eksperimentalno snimanje oka

Snimanje oka je izvedeno u kompjuterskoj laboratoriji pod različitim uvjetima kako bi bilo lakše utvrditi koji uvjeti snimanja su najbolji. U nastavku je opisan eksperimenti, te rezultat istog.

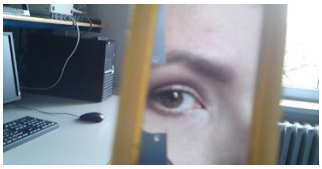
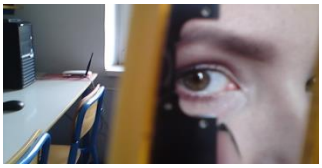
Za eksperiment su korišteni kamera, sočiva i ogledalo.
U tabeli 4.1 prikazani su rezultati prve postavke eksperimenta.

Tabela 4.1: Prva postavka eksperimenta

	Situacija	Postavka lica i kamere	Rezultat	Opis rezultata
Postavka 1	Prva	> kamera je usmjerena direktno u oko > lice nije direktno okrenuto prozoru već laptopu > bez korištenja ogledala i sočiva > snimano desno oko > kamera udaljena cca. 10 cm od oka		Pozicija osvjetljenja u odnosu na poziciju glave je takva da stvara mnogo sjene koja može uzrokovati pogrešna očitanja položaja irisa oka.
	Druga	> lice okrenuto prozoru	 	Oko je jasno vidljivo te sve značajke istog, što ukazuje na to da bi pozicija irisa oka mogla biti dobro ekstraktovana ukoliko je osvjetljenje takvo da svjetlost ide ka licu. Jedini problem stvara odsjaj prozora u oku.
	Treća	> lice okrenuto prema tački koja se nalazi cca. 45 stepeni od prozora prema laptopu		Ovakvo osvjetljenje također daje dobre rezultate obzirom da je iris oka jasno vidljiv te su sjene minimizirane. Samim tim je moguće odrediti okvirni ugao za koji će osoba moći okrenuti glavu, a da rezultati ekstrakcije pozicije zjenice budu zadovoljavajući.
	Četvrta	> lice okrenuto 180 stepeni od prozora		Sa slike je jasno vidljivo kako ovaj način osvjetljavanja nije uopće dobar obzirom da je oko pretamno i značajke oka nisu jasno vidljive.
	Peta	> lice okrenuto 135 stepeni od prozora		Moguće je u određenoj mjeri primijetiti značajke oka, ali je prisutno značajno zatamnjenje u odnosu na situaciju kada je lice okrenuto izvoru svjetlosti.

U tabeli 4.2 prikazani su rezultati druge postavke eksperimenta gdje je uključeno i korištenje ogledala.

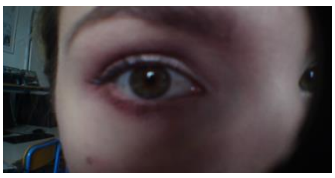
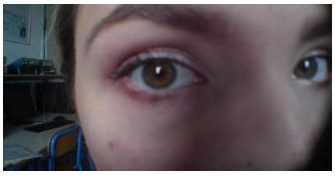
Tabela 4.2: Druga postavka eksperimenta

	Situacija	Postavka lica, kamere i ogledala	Rezultat	Opis rezultata
Postavka 2	Prva	> lice okrenuto laptopu > svjetlost dolazi sa desne strane > snimano lijevo oko > udaljenost ogledala od kamere cca. 5 cm > udaljenost ogledala od oka cca. 10 cm		Prikaz oka nije jasno vidljiv, te značajke oka nisu jasno uočljive zbog prisustva sjene.
	Druga	> lice okrenuto prozoru > ostale karakteristike iste kao u prvoj situaciji	 	Prikaz oka je znatno poboljšan u odnosu na prethodnu situaciju. Očigledni su problemi sa fokusom na obje slike. ali je ovim eksperimentom otvorena mogućnost korištenja ogledala koji će omogućiti lakše snimanje oka, te neinvazivniju metodu.

U tabeli 4.3 prikazani su rezultati treće postavke eksperimenta u kojoj su korištena sočiva,

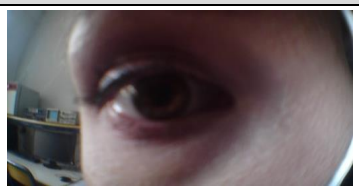
bez korištenja ogledala.

Tabela 4.3: Treća postavka eksperimenta

	Situacija	Postavka lica, kamere i sočiva	Rezultat	Opis rezultata
Postavka 3	Prva	> lice okrenuto laptopu > korištenje MACRO + 0.67 X WIDE sočiva > snimano desno oko > udaljenost kamere cca. 6 cm od oka		Osvjetljenje izaziva velike sjene na slici oka, što može dovesti do pogrešnog očitanja pozicije irisa. Također je slika veoma mutna, bez mogućnosti izoštrenja obzirom da blur efekat izaziva samo sočivo.
	Druga	> lice okrenuto izvoru svjetlosti, odnosno prozoru > korištena sočiva kao u prethodnoj situaciji > ista udaljenost kamere kao u prethodnoj situaciji	 	Rezultat ovog pristupa je ponovo zamućena slika koju izaziva korištenje sočiva.
	Treća	> lice okrenuto na 45 stepeni između prozora i laptopa		Kao i u prethodnom slučaju, rezultat ovog pristupa je ponovo zamućena slika koju izaziva korištenje sočiva.

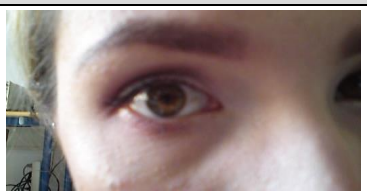
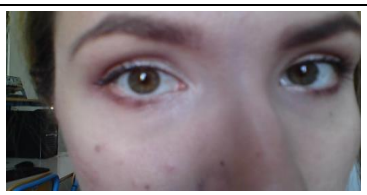
U tabeli 4.4 prikazani su rezultati četvrte postavke eksperimenta, također korištenjem sočiva, s tim da su korištena sočiva drugog tipa.

Tabela 4.4: Četvrta postavka eksperimenta

	Situacija	Postavka lica, kamere i sočiva	Rezultati	Opis rezultata
Postavka 4	Prva	> sočivo Fish Eye lens je korišteno u svim situacijama > lice okrenuto ka laptopu > snimano desno oko > udaljenost kamere od oka između 3 cm i 4 cm		Za svaku od situacija je moguće zaključiti da korištenje sočiva nepovoljno utječe na kvalitet slike
	Druga	> lice okrenuto ka prozoru		
	Treća	> lice okrenuto 45 stepeni od laptopa ka prozoru		

U tabeli 4.5 prikazani su rezultati pete postavke eksperimenta, bez korištenja sočiva i ogledala, s tim da su auto contrast i auto whitebalance uključeni.

Tabela 4.5: Peta postavka eksperimenta

	Situacija	Postavka lica i kamere	Rezultat	Opis rezultata
Postavka 5	Prva	> auto contrast i auto whitebalance uključeni u svim situacijama > lice okrenuto laptopu		U odnosu na prvu postavku, primijećen je značajan problem s fokusom.
	Druga	> lice okrenuto prozoru		Slika je mnogo više zamućena nego u prvoj postavci, stoga ovaj metod neće biti korišten.
	Treća	> lice okrenuto 45 stepeni od prozora ka laptopu		

4.1.1 Zaključak:

Bitno je primijetiti nekoliko činjenica koje proizilaze iz ovog eksperimenta. Osvjetljenje itekako utječe na sliku, a dobro osvjetljenje stvara odsjaj u oku, stoga će to stvarati problem prilikom očitanja pozicije zjenice u ovisnosti od metode koja bude korištena za nalaženje iste. Osvjetljenje treba biti takvo da je lice osobe osvijetljeno, te da izvor najsjajnije svjetlosti u prostoriji bude takav da je nasuprot osobe. Eksperiment je izvršen u uslovima gdje je postojao samo prirodni izvor svjetlosti, čemu se i teži, obzirom da bi bilo koja vrsta vještačkog svjetla koje osvjetljava lice ili oko čovjeka, bila zamarajuća za pacijenta. Cilj je koristiti neinvazivnu metodu, a zamjena IR svjetlosti nekom drugom svjetlosti nije dobro rješenje za zdravlje pacijenta. Dobro osvjetljenje oka je ključno za ekstrakciju značajki oka.

4.2 Mehanička konstrukcija

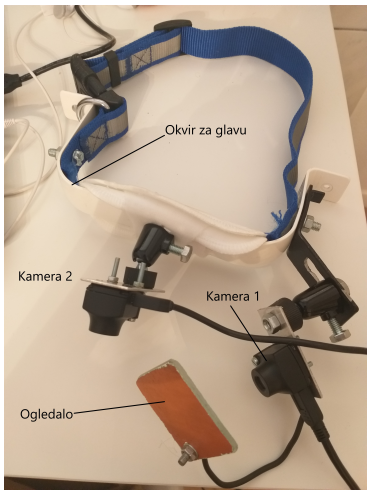
Na osnovu sprovedenog eksperimentalnog dijela, odlučeno je na koji način će biti realizovan fizički dio uređaja. Odabran je pristup sa ogledalom obzirom da isti omogućava da kamera bude fiksirana na okviru, umjesto da bude usmjerena ka oku što bi napravilo težinski dibalans u strukturi. Također je na ovaj način omogućena pokretljivost ogledala čime je smanjena sjena koja pada na oko i koja može ugroziti očitanje pozicije zjenice. Ogledalo je mnogo lakše od kamere, te je fleksibilno u smislu pozicioniranja čime je omogućeno da ogledalo bude postavljeno tako da ne zaklanja vidno polje korisnika, a ujedno prikazuje oko i omogućava ekstrakciju zjenice.

Potrebno je omogućiti da osoba pokreće glavu, odnosno, ne postavlja se uslov na konstantnu fiksaciju glave tokom gledanja u ekran. Da bi to bilo realizovano, potrebno je omogućiti automatsku rekalibraciju u slučaju da osoba pomjeri glavu. Za tu svrhu je dodana još jedna kamera koja će omogućiti detekciju pomijeranja glava te automatsko skaliranje u odnosu na novi koordinatni sistem. Dodatna kamera treba biti postavljena tako da snima ekran, te da na osnovu orijentacije i pozicije istog bude moguće odrediti kretanje oka u odnosu na novonastali

koordinatni sistem.

Na osnovu pomenutih uslova realizovan je fizički sistem koji se sastoji od dvije kamere, okvira za glavu, te ogledalca. Veličinu okvira je moguće podešavati shodno veličini glave osobe koja koristi uređaj. Ogledalce je postavljeno na bakarni nastavak čime je omogućeno namiještanje pozicije ogledala shodno poziciji oku korisnika.

Na slici 4.1 je prikazana slika uređaja, te kako izgleda na korisnikovoj glavi.



(a) Fizička struktura uređaja

(b) Uređaj na glavi korisnika

Slika 4.1: Mehanička konstrukcija uređaja

Kamere korištene za snimanje su uEye UI - 1008 XS (informacije u prilogu). Za okvir je korištena plastična struktura primarno namijenjena korištenju u instalaciji cijevi za ventilaciju, dok je kao remen korišten dio okovratnika za kućne ljubimce. Uređaj je samo prototip, stoga je dodano i nekoliko jastučića u unutrašnjosti koji omogućuju udobnije korištenje uređaja.

Poglavlje 5

Realizacija softverskog dijela

Kao što je pomenuto, snimanje je obavljano sa dvije kamere. Kamera 1 se nalazi na lijevoj strani uređaja i namještena je tako da u ogledalcu snima lijevo oko korisnika. Kamera 2 se nalazi na centralnom dijelu uređaja i zadužena je za snimanje ekrana te vršenje automatske rekalibracije.

U nastavku će prvo biti obrađeno snimanje kamere 1, a nakon toga snimanje kamere 2.

Kamera 1 izvršava najviše posla, obzirom da je pomoću nje potrebno snimiti oko te odrediti pravilnu poziciju zjenice oka. Razvijeno je više metoda za određivanje pozicije oka ([1], [7], [8]), ali je u ovom radu korištena metoda koja se zasniva na određivanju centra najtamnijeg dijela na slici. Činjenica je da je iris najtamniji dio dijela oka koji predstavlja beonjaču sa irisom, stoga je na slici na kojoj je prikazano samo pomenuto teoretski moguće dobiti informaciju o tome gdje se nalazi centar irisa. To je moguće dobiti iz same informacije o vrijednosti piksela, tako što će na osnovu sume svake kolone biti određena kolona sa najvećom sumom, dok će za redove biti urađeno isto. Ovim postupkom je moguće dobiti aproksimiranu poziciju zjenice na osnovu reda elementa najveće sume po kolonama i redovima.

5.1 Ekstrakcija oka

Budući da je potrebno odrediti poziciju centra irisa, prvenstveno je potrebno na slici locirati oko. U tu svrhu je korišten metod kaskadne klasifikacije (*engl. Cascade Classification*).

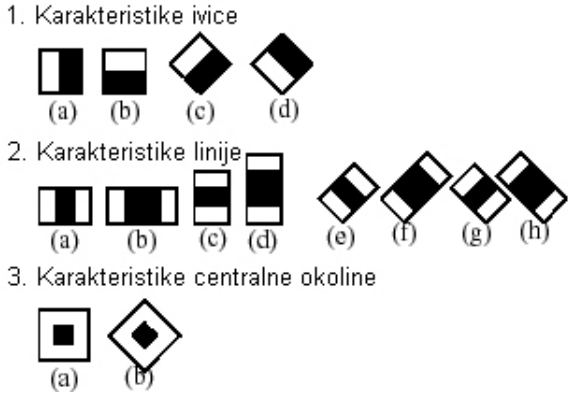
5.1.1 Kaskadni klasifikator za detekciju objekata baziran na Haarovim karakteristikama

Engl. Haar Feature-based Cascade Classifier for Object Detection klasifikator za detekciju objekata objašnjen u nastavku je inicijalno predložen od strane Paula Viole, dok je Rainer Lienhart poboljšao algoritam. [3]

Klasifikator (imenovan kao kaskada pojačanih klasifikatora na bazi karakteristika sličnih Haar-u (*engl. cascade of boosted classifiers working with Haar-like features*) je treniran sa nekoliko stotina uzoraka određenog objekta (kao što je npr. lice ili automobil), koji se nazivaju pozitivni uzorci i skalirani su na istu veličinu, i isto tako sa nekoliko stotina negativnih uzoraka koji predstavljaju proizvoljne slike iste veličine.

Nakon što je klasifikator istreniran, može biti apliciran na regiju od interesa (iste veličine kao i ona korištena tokom treniranja) ulazne slike (*engl. input image*). Klasifikator kao izlaz daje ”1” ako je velika vjerovatnoća da regija pokazuje objekat (npr. lice ili automobil), i ”0” u suprotnom. Da bi objekat bio pronađen u cijeloj slici potrebno je pomijerati prozor za pretragu preko slike i jednostavno provjeriti svaku lokaciju koristeći klasifikator. Klasifikator je dizajniran tako da njegova veličina može jednostavno biti promijenja kako bi se pronašli objekti od interesa na različitim veličinama, što je efikasnije nego mijenjanje veličine same slike. Iz navedenog se da zaključiti da, ukoliko je potrebno pronaći objekat nepoznate veličine na slici, proces skeniranja treba obaviti nekoliko puta sa različitim veličinama.

Riječ ”kaskada” u imenu klasifikatora znači da se rezultirajući klasifikator sastoji od nekoliko jednostavnijih klasifikatora (faza) koje su uzastopno primijenjene na regiju od interesa sve dok u jednoj fazi kandidat ne bude odbijen ili su sve faze uspješno provedene. Riječ ”pojačani” (*engl. boosted*) znači da su klasifikatori u svakoj fazi kaskade sami od sebe kompleksi i sastavljeni su od osnovnih klasifikatora koristeći jednu od četiri različite boosting tehnike (pondenirano glasanje (*engl. weighted voting)*. Trenutno Discrete Adaboost, Real Adaboost, Gentle Adaboost i Logitboost su podržani. Karakteristike slične Haaru su ulazi za osnovne klasifikatore i računaju se kako je objašnjeno u nastavku. Trenutni algoritam koristi sljedeće karakteristike prikazane na slici 5.1 slične Haaru:



Slika 5.1: Haarove karakteristike

Karakteristika korištena u konkretnom klasifikatoru je specificirana oblikom (1a, 2b etc.), pozicijom unutar regije od interesa i skaliranjem (skaliranje nije uvijek isto kao skaliranje korišteno u fazi detekcije, mada su ova dva skaliranja pomnožena). Na primjer, u slučaju karakteristike treće linije (2c) rezultat se računa kao razlika između sume piksela slike unutar pravougaonika koji pokriva cijelu karakteristiku (uključujući i dvije bijele trake i crnu traku u sredini) i sume piksela slike unutar crne trake pomnožene sa 3 kako bi se kompenzirala razlika u veličinama područja. Suma vrijednosti piksela na pravougaonim regijama se računa brzo koristeći integralne slike (*engl. integral images*).

Kaskadni klasifikator korišten u programu se naziva *detectMultiScale* i detektuje objekte različitih veličina u ulaznoj slici. Detektovani objekti su vraćeni kao lista pravougaonika.

Funkcija u OpenCVu koja omogućava korištenje ovog klasifikatora se definira kao:

```
void CascadeClassifier::detectMultiScale(const Mat& image, vector<Rect>& objects,
double scaleFactor = 1.1, int minNeighbors=3, int flags=0, Size min_size=Size(),
Size maxSize=Size())
```

Korištenje funkcije je opisano u prilogu.

Linija koda koja izvršava detekciju oka:

```
eyeCascade.detectMultiScale(frame, eyes, 1.1, 2, 0 | CV_HAAR_SCALE_IMAGE,
cv::Size(150, 150))
```

Ukoliko je oko detektovano oko istog se iscrtava pravougaonik koji ozačava područuje oka na osnovu pravougaonih okvira koje vraća pozivanje funkcije `detectMultiScale`.

Primjer pozivanja prethodne funkcije prikazan je na slici 5.2.

(a) Slika koju snima kamera

(b) Slika na kojoj je detektovano oko

Slika 5.2: Primjer detekcije oka

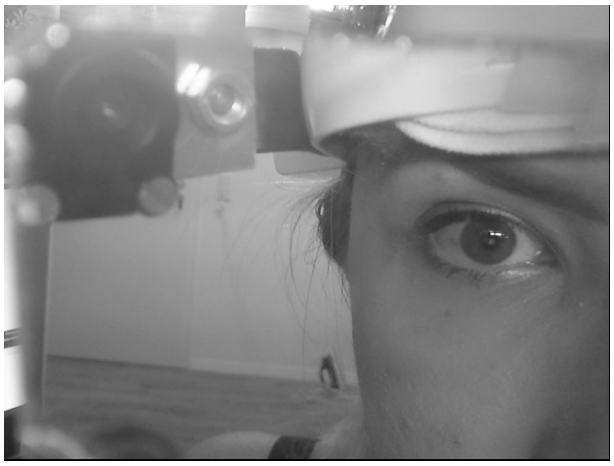
5.1.2 Kreiranje maske

Upravljanje programom uređaja zahtijeva određeni nivo ekspertize, stoga se uvodi pojam korisnika i administratora. Korisnik je obično pacijent, dok je administrator osoba sa iskustvom dovoljnim za ispravno korištenje uređaja. Obzirom da je uređaj primarno namijenjen osobama sa motoričkim poteškoćama, jasno je da je potrebno imati administratora koji će izvršavati sve potrebne dijelove programa za ispravan rad uređaja.

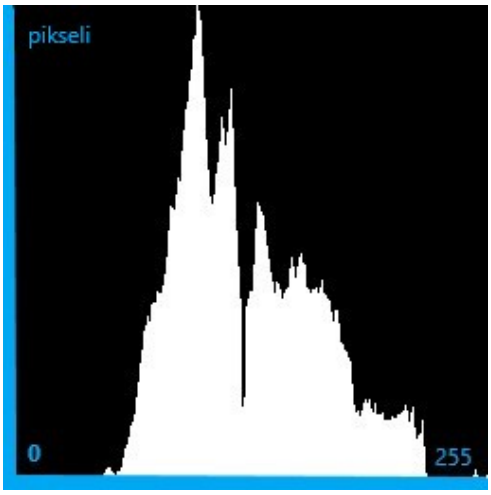
Na slici je potrebno izdvojiti regiju od interesa (*engl. Region of Interest*) koja predstavlja beonjaču oka sa irisom. Prije nego to bude urađeno, potrebno je obraditi sliku koju snima kamera 1. Slika koju snima kamera se konvertuje u sivu monohromatsku sliku, te se na istoj primjenjuje izjednačavanje histograma (*engl. Histogram Equalization*). Potrebno je izvršiti izjednačavanje histograma kako bi se dobila slika sa više kontrasta. Izjednačavanje histograma se radi u slučaju kada je slika takva da su njeni pikseli predstavljeni bliskim kontrastnim vrijednostima. Obzirom da je potrebno na slici oka vršiti mnoge operacije, potrebno je što bolje dobiti tačne podatke, te se iz tog razloga izjednačavanje histograma i radi nakon što je slika prebačena u monohromatsku sliku.

Izjednačavanje histograma

Histogram slike predstavlja grafičku reprezentaciju distribucije intenziteta slike. Kvantificira broj piksela za svaki nivo intenziteta koji se razmatra. Na slici 5.3 prikazana je monohromatska slika, te na desnoj strani iste njen histogram.



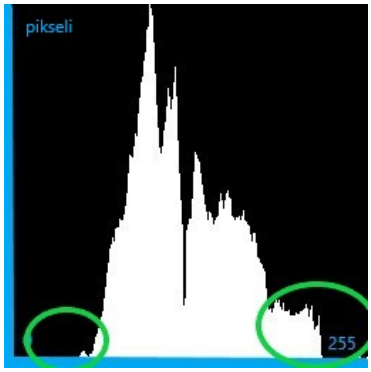
(a) Monohromatska slika



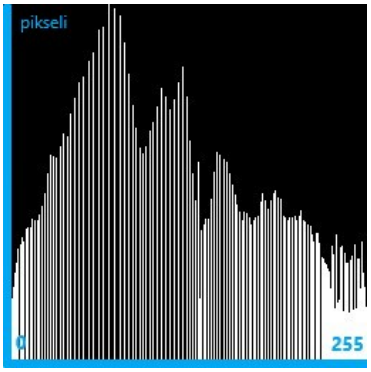
(b) Histogram monohromatske slike

Slika 5.3: Primjer histograma slike

Histogram equalization predstavlja metod koji poboljšava kontrast slike kako bi rastegnuo raspon intenziteta. Na slici 5.3 je moguće vidjeti kako svi pikseli izgledaju centrirani oko sredine dostupnog raspona intenziteta. Ono što radi izjednačavanje histograma je da ”rasteže” ovaj raspon. Na slici 5.4 prikazani su rezulati primjene izjednačavanja histograma na prethodnu sliku. Zelene kružnice indiciraju nedovoljno skoncentriranih (*engl. underpopulated*) intenziteta. Nakon primjene izjednačavanja, dobijamo histogram kao što je na centru slike. Rezultujuća slika je prikazana na slici desno.



(a) Histogram prije primjene izjednačavanja histograma



(b) Histogram nakon primjene izjednačavanja histograma

(c) Slika nakon primjene izjednačavanja histograma

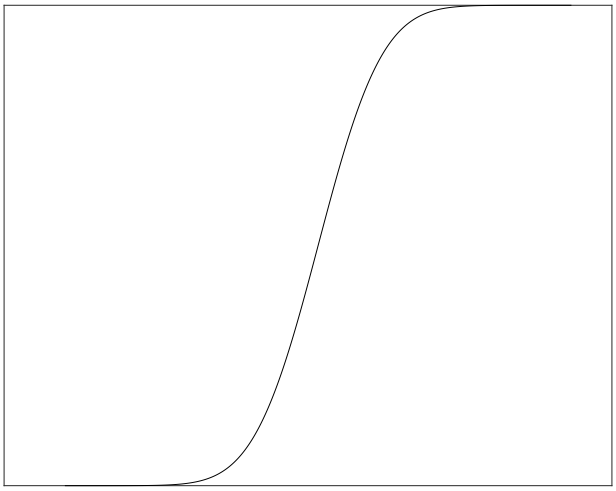
Slika 5.4: Primjer izjednačavanja histograma slike

Izjednačavanja implicira mapiranje jedne distribucije (datog histograma) u drugu distribuciju (širu i uniformniju distribuciju vrijednosti intenziteta) tako da su vrijednosti intenziteta raširene preko cijelog opsega.

Da bi se ostvario efekat izjednačavanja, remapiranje treba biti *funkcija kumulativne distribucije* (*engl. cumulative distribution function (cdf)*). Za histogram **H(i)** kumulativna distribucija **H'(i)**:

$$H'(i) = \sum_{0 \leq j < i} H(j)$$

Za korištenje ove funkcije, moramo normalizirati **H'(i)** tako da je maksimalna vrijednost 255 (ili maksimalna vrijednost za intenzitet slike). Za primjer iznad, kumulitivna funkcija je prikazana na slici 5.5:



Slika 5.5: Kumulativna funkcija

Na kraju je još potrebno primijeniti jednostavno remapiranje kako bi bile dobijene vrijednosti intenziteta izjednačene slike:

equalized(x,y) = H'(src(x,y))

Izjednačavanje histograma slike oka je izvršeno pozivanjem funkcije `equalizeHist` koja prima dva parametra; prvi parametar je slika čije se izjednačavanje histograma vrši, dok je drugi parametar slika koja se dobije kao rezultat izjednačavanja histograma. Potrebno je spomenuti da su slike varijable tipa `Mat`, što predstavlja klasu koja omogućava predstavljanje slika kao matrica, sadržavajući i druge informacije koje opisuju samu matricu.

5.1.3 Karakteristične tačke oka

Potrebno je na neki način izdvojiti beonjaču oka zajedno sa zjenicom, što je teško uraditi ukoliko nisu poznate karakteristične tačke oka. Čak i ako su poznate karakteristične tačke oka, postavlja se pitanje kolika veličina luka koji aproksimira oko treba biti da ti bila tačnost što bolja. Jasno je da luk tada ne može biti fiksirane veličine i mora ovisiti od drugih parametara. Upravo zbog toga je pribjegnuto korištenju manuelnog označavanja četiri karakteristične tačke oka čime se povećava preciznost, a i vrijeme izvršavanja obzirom da sama detekcija značajki oka zahtijeva resurse koji usporavaju rad programa. Manuelno se označavanje karakterističnih tačaka oka vrši tako što se označavaju sredine gornjeg i donjeg kapka oka, te uglovi oka.

Tokom izvođenja eksperimenata je primijećeno da su koordinate pravougaonika, koji označavaju područje sa okom, skoro uvijek drugačije, iako se pozicija samog oka nije mijenjala u odnosu na kameru. Iz tog razloga je odlučeno da maska bude kreirana na cjelokupnoj slici koju snima prva kamera, a ne samo na ekstraktovanoj slici oka. Na ovaj način je iskorištena informacija da je relativna pozicija kamere u odnosu na oko, i obratno, uvijek ista, obzirom da bi u protivnom bilo potrebno porediti trenutnu poziciju pravougaonika sa prvobitnom, što je redundantno.

Korištenjem informacija o četiri karakteristične tačke, moguće je kreirati masku tako da je iz slike oka izdvojena samo regija od interesa, dok je ostatak slike bijele boje.

5.1.4 Segmentacija pragom

Da bi bilo moguće primijeniti algoritam određivanja zjenice oka, potrebno je prvo novonastalu sliku prebaciti u format binarne slike, odnosno slike koja sadrži samo crne i bijele piksele. Za tu svrhu je potrebno odrediti granicu (*engl. threshold*) koja predstavlja vrijednost s kojom se poredi svaki piksel slike, te ako je vrijednost piksela slike veća od postavljenog thresholda, piksel se posmatra kao bijeli piksel (vrijednost 255), dok je u suprotnom to crni piksel (vrijednost 0). Objašnjeni postupak se naziva segmentacija pragom (*Image Thresholding*).

U nastavku će biti objašnjeno nekoliko vrsta image thresholdinga, što podrazumijeva jednostavno određivanje praga (*engl. Simple Thresholding*), adaptivno određivanje praga (*engl. Adaptive Thresholding*), te Otsu određivanje praga (*engl. Otsu's thresholding*).

5.1.5 Jednostavno određivanje praga

Simple thresholding je upravo ono što mu i samo ime kaže, jednostavni thresholding kod kojeg važi da ukoliko je vrijednost piksela veća od granične vrijednosti, pikselu se dodjeljuje jedna vrijednost (npr. bijela), dok se u protivnom pikselu dodjeljuje druga vrijednost (npr. crna).

Za postizanje jednostavnog thresholdinga koristi se funkcija `cv::threshold(src, dst, thresh, maxval, type)` objašnjena u prilogu.

Očigledno je kao tip thresholdinga potrebno odabrati `THRESH_BINARY`, obzirom da su potrebni samo crni i bijeli pikseli, bez sivih, i to tako da tamniji pikseli odgovoraju crnim, a svjetliji bijelim pikselima. Ovo je potrebno uraditi jer se algoritam estimacije pozicije zjenice bazira na veličini sume piksela po redovima i kolonama.

5.1.6 Adaptivno određivanje praga

U prethodnom slučaju je korištena globalna vrijednost kao threshold vrijednost, što može biti loše u slučajevima kada slika ima različite uvjete osvjetljenja u različitim dijelovima. U tom slučaju je dobro koristiti adaptivni thresholding. U ovom slučaju algoritam računa threshold za male dijelove slike, stoga je moguće dobiti različite thresholde za različita područja iste slike, što daje bolje rezultate za slike sa varijativnim osvjetljenjem.

Detalji adaptive thresholdinga su opisani u prilogu.

Primjer adaptivnog thresholdinga na slici oka je prikazan na slici 5.6.

(a) Originalna slika

(b) Global thresholding (v = 127)

(c) Adaptive Mean Thresholding

(d) Adaptive Gaussian Thresholding

Slika 5.6: Primjer primjene adaptivnog thresholdinga sa različitim tipovima na slici oka

Otsu binarizacija

U simple thresholdingu je korištena proizvoljna vrijednost za threshold vrijednost, ali je teško znati da li je ta vrijednost odabrana ispravno. Jedini način za provjeru ispravnosti izabrane vrijednosti je kroz metod pokušaja i pogreške. Međutim, ukoliko se posmatra **bimodalna slika** (***engl. bimodal image***), što predstavlja sliku čiji histogram ima dva vrha, tada je za takvu sliku moguće uzeti vrijednost u sredini tih vrhova kao vrijednost thresholda, što je upravo ono što radi **Otsu binarizacija**. Jednostavno rečeno, Otsu binarizacija automatski računa threshold vrijednost i vraća je kao drugi parametar, **retVal** za funkciju u simple thresholdingu. Ako nije korištena Otsu binarizacija, retVal je ista kao i threshold vrijednost koja je korištena.

Princip rada Otsu binarizacije

Otsu algoritam pokušava naći threshold vrijednost (t) koja minimizira **ponderisanu varijansu unutar klase (*engl. weighted within-class variance*)** datu relacijom:

$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t) \tag{5.1}$$

gdje je

$$q_1(t) = \sum_{i=1}^{t} P(i) \qquad \& \qquad q_2(t) = \sum_{i=t+1}^{I} P(i) \tag{5.2}$$

$$\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{q_1(t)} \qquad \& \qquad \mu_2(t) = \sum_{i=t+1}^{I} \frac{iP(i)}{q_2(t)} \tag{5.3}$$

$$\sigma_1^2(t) = \sum_{i=1}^{t} [i-\mu_1(t)]^2 \frac{P(i)}{q_1(t)} \qquad \& \qquad \sigma_2(t) = \sum_{i=t+1}^{I} [i-\mu_1(t)]^2 \frac{P(i)}{q_2(t)} \tag{5.4}$$

Zapravo, ovaj algoritam pronalazi vrijednost t koja se nalazi između dva vrha tako da je varijansa obje klase minimalna.

5.1.7 Estimacija pozicije zjenice

Nakon što je izvršena obrada slike oka, spremna je za primjenu algoritma koji će aproksimizirati poziciju zjenice. Jasno je da je oko, odnosno iris, najtamniji dio dijela oka sa beonjačom, stoga je iz tog razloga logično posmatrati najtamniji dio obrađene slike oka.

Sabiranjem svih elemenata po svakoj koloni matrice slike dobija se suma svih piksela za svaku kolonu. Pozicija najveće sume po kolonama u vektoru svih suma daje y vrijednost estimirane koordinate zjenice, dok je x koordinatu moguće dobiti sabiranjem svih elemenata po redovima, te je rezultat toga vektor sa svim sumama po redovima. Pozicija najvećeg elementa tog vektora daje estimiranu x koordinatu zjenice. Uzimajući u obzir da je slika oka nerijetko takva da je linija koja spaja krajeve oka nakrivljena pod nekim uglom, zakretanjem slike za taj ugao u odgovarajućem smjeru, dobija se slika nad kojom je potrebno izvršiti estimaciju pozicije zjenice.

Obzirom da je obrađena slika takva da je svaki piksel ili vrijednosti 255 (bijela boja), ili vrijednosti 0 (crna boja), prvo je potrebno obrnuti svaki piksel tako da je crni piksel sada bijeli i obratno. Ovo je dobro uraditi zbog pozitivne logike koja je prirodnija, tražeći maksimalnu, a ne minimalnu sumu.

5.1.8 Detekcija ekrana

Detekciju ekrana je potrebno vršiti kako bi se ostvarila korelacija između koordinatnog sistema oka, te koordinatnog sistema računara, odnosno ekrana. Jasno je da je pomak oka kada osoba gleda u dvije tačke na ekranu mnogo manji nego stvarna udaljenost tih tačaka. Upravo iz tog razloga je potrebno imati vezu između koordinatnog sistema u kojem se kreće oko kada gleda u ekran, te koordinatnog sistema ekrana. Da bi bilo moguće napraviti tu vezu, potrebno je izvršiti

detekciju ekrana. Detekciju ekrana je potrebno vršiti i zbog automatske rekalibracije, obzirom da je potrebno porediti novonastalu poziciju ekrana sa pozicijom ekrana tokom kalibracije. Potrebno je naglasiti da se pod terminom ”pozicija ekrana” podrazumijeva pozicija ekrana u odnosu na osobu, odnosno u odnosu na kameru 2 koja je i zadužena za snimanje ekrana.

Za detekciju ekrana je najbolje koristiti gometrijske oblike obzirom da je iste najlakše detektovati i procijeniti njihovu poziciju. Iz tog razloga, na zaslonu ekrana je potrebno u uglovima prikazati oblike koje će detektovati druga kamera, a izabrani su krugovi. Za detekciju krugova je u OpenCVu razvijena funkcija `HoughCircles`, čija je upotreba objašnjena u prilogu.

5.1.9 Usmjerenost pogleda

Možda i najbitnija stavka cijelog projekta je određivanje usmjerenosti pogleda osobe na osnovu informacije o poziciji zjenice. Za tu svrhu je dobro napraviti podjelu ekrana na nekoliko dijelova i razmatrati kako se mijenja pozicija zjenice u odnosu na mijenjanje pogleda osobe od dijela do dijela. Jasno je da je prostor pomijeranja oka veoma mali iako udaljenost dijelova na ekranu može biti velika, stoga je logično očekivati da podjela ekrana neće moći biti izvršena na mnogo dijelova.

Za svakog korisnika, kreira se funkcija transformacije koja odgovara kretanju oka i posmatranoj slici. Pomenuta transformacija se određuje kroz kalibracijski proces neovisan za svakog korisnika. Nakon definiranja i primjenjivanja odgovarajuće transformacijske funkcije, eye tracker snima svaki pokret oka za sliku ili scenu. [6]

Kalibracijski proces spomenut iznad se obično izvršava tako što se od korisnika traži da gleda u nekoliko tačaka programskog zaslona, praveći vezu između pozicije zjenice fokusirane na svaku od tačaka sa koordinatama tih tačaka, time kreirajući transformacijsku matricu između dva referentna sistema. Ukoliko je proces kalibracije izvršen netačno, programski rezultati procjene transformacijske matrice će također biti netačni, što može dovesti do odstupanja u rezultatima za nekoliko piksela.

Koordinate pogleda korisnika moraju biti estimirane u odnosu na zaslon koji korisnik posmatra. Matematički, cilj je izvršiti transformaciju tačke iz jednog koordinatnog sistema u drugi. Da bi to bilo izvršeno, potrebna je verifikacija da korisnik gleda prema ekranu kako bi bilo moguće postaviti pokazivač na te koordinate. Iz tog razloga je potrebno vršiti detekciju ekrana koji se nalazi u vidnom polju korisnika. Obzirom da ljudski pogled varira konstantno, većina algoritama za praćenje nisu u stanju izvršiti ovu detekciju u realnom vremenu, posebno za promjenjivo osvjetljenje, te veličinu i udaljenost od ekrana. Nadalje, većina takvih postupaka su kompjuterski zahtjevni i zahtijevaju previše vremena za prikaz rezultata, pri čemu su i prilično netačni u sredinama sa mnogo objekata. Međutim, detekcija ekrana također zahtijeva da ekran bude postavljen ispred korisnika, i da su jasno prikazane referentne tačke ekrana koje treba snimati.

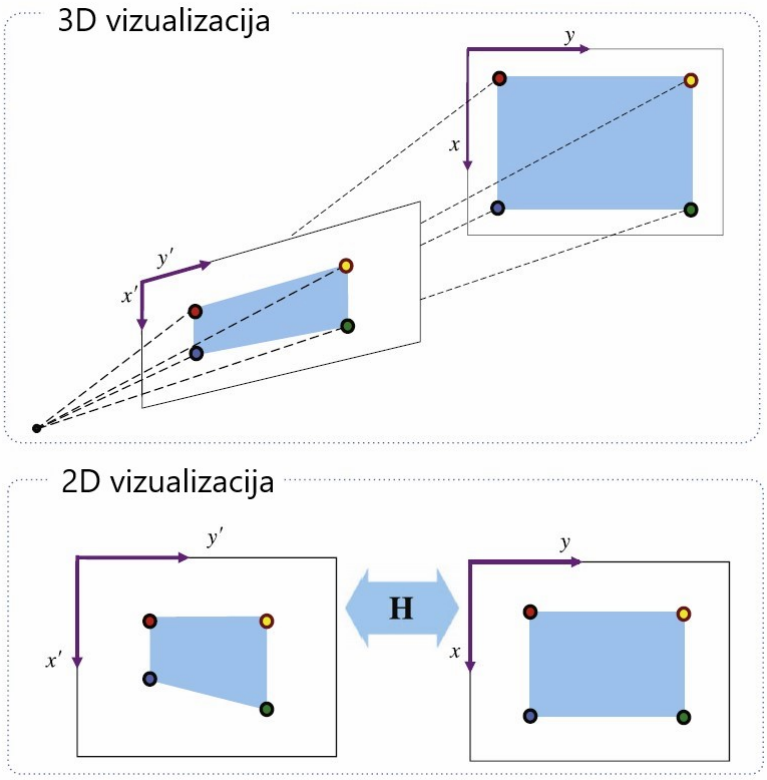
Kako bi se riješili pomenuti problemi, uvodi se dodatna kamera (u ovom slučaju kamera 2) koja je zadužena za snimanje ekrana i koja je montirana na uređaj na glavi, te joj je jedina uloga detekcija četiri oblika na ekranu koji okružuju sliku koju korisnik gleda. Druga kamera je komplementarna kameri koja snima oko i omogućava registraciju bilo kojih pokreta glave koji utječu na sliku koju snimaju prva i druga kamera.

Da bi se definisala relacija između dvije različite kamere (samim tim i referentni odnos dva koordinatna sistema), koriste se četiri detektovane tačke, tako da je interno poznat marker, odnosno oznaka, svake od njih.

Estimacija matrice homogene transformacije

Obzirom da postoje dvije referentne tačke (korisnikovo vidno polje i ekran) koje sistem posjeduje, potrebno je ostvariti homografsku vezu između njih tako da su koordinate jedne kamere ”prebačene” u koordinate druge kamere. Ovaj proces se obavlja **matricom homogene transformacije** (poznata kao **H** matrica), koja predstavlja linearnu operaciju između bilo koje dvije tačke dva sistema. Ovaj proces se vrši u dva koraka; (1) prvo je potrebno izvršiti kalibraciju između frame-a slike koju snima prva kamera i slike koju snima druga kamera (ona koja snima ekran), i (2), vrši se kalibracija između kamere koja snima ekran i četiri referentne tačke ekrana. [2]

Prvo je potrebno odrediti relativnu poziciju koordinatnih sistema slike koju snima kamera 1 i slike koju snima kamera 2. Generalno je potrebno poznavati barem četiri tačke, koje odgovaraju svakom referentnom sistemu, kako bi bilo moguće estimirati matricu **H**; što znači da su potrebne četiri koordinate po svakoj slici koje odgovaraju istoj tački ili objektu. Da bi bilo moguće tačno odrediti vezu među koordinatnim sistemima, potrebno je u isto vrijeme posmatrati sliku koju snima kamera 1 i sliku koju snima kamera 2, čime se omogućava određivanje četiri tačke od interesa za određivanje homogene transformacije. Četiri odabrane tačke su koordinate zjenice kada oko gleda ka četiri kalibracijske tačke na ekranu i četiri tačke koje detektuje kamera 2. Prethodno rečeno je grafički prikazano na slici 5.7.



Slika 5.7: Transformacija u 3D i 2D prostoru [2]

Za estimaciju matrice **H** koristi se ideja projiciranja tačke **m** = [x', y', 1] jednog koordinatnog sistema, u odgovarajuću tačku drugog koordinatnog sistema **n** = [x, y, 1]. Ova veza se definiše jednačinom 5.5.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{5.5}$$

Gdje je $\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$ matrica **H** homogene transformacije veličine $3x3$.

Ovaj problem se može predstaviti i kao linearan sistem jednačina (jednačina 5.6) za svaku korespondirajuću tačku $(x', y') \mapsto (x, y)$ između dvije slike.

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & y'_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & y'_2y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y'_3x_3 & y'_3y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & y'_4y_4 \end{bmatrix} = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} \tag{5.6}$$

Gdje je **A** = $\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & y'_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -y'_2x_2 & y'_2y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -y'_3x_3 & y'_3y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -y'_4x_4 & y'_4y_4 \end{bmatrix}$, **h** = $\begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix}$, i **b** = $\begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix}$.

Tačka (x_i, y_i) je i-ta korespondirajuća tačka. Linearna jednačina 5.6 u obliku **Ah=b** može biti riješena koristeći metod najmanjih kvadrata, čije je rješenje dato sa $\mathbf{h} = [\mathbf{A^T A}]^{-1}\mathbf{A^T b}$.

Opisana procedura se primjenjuje koristeći sliku kamere 1 i sliku kamere 2, i informacija dobijenih iz njih. Računanjem parametara matrice **H** korištenjem jednačine 5.6 i uvrštavanjem u jednačinu 5.5, moguće je dobiti reprezentaciju bilo koje tačke iz jednog u drugom koordinatnom sistemu, pri čemu je odabir referentnog proizvoljan. Ova matematička reprezentacija omogućava reprezentaciju koordinata vizualne pažnje korisnika iz sopstvenog referentnog sistema u referentni sistem kamere 2. Potrebno je naglasiti da su koordinatni sistemi kamere 1 i kamere 2, a samim tim i njihovih slika, fiksirani jedan u odnosu na drugi, obzirom da su kamere pozicionirane na okviru uređaja i nepomične su u odnosu na oko.

Drugi korak koristi koordinate oblika (detektovanih od strane kamere 2) i korisničkog zaslona, na kojem korisnik fiksira pogled. Da bi se koristila referenca korisničkog zaslona, estimira se nova matrica **H**. U koordinatnom sistemu kamere 2, koordinate detektovanih tačaka ekrana će biti ažurirane kontinuirano, obzirom da su iste određene opisanim sistemom, dok je je u referentnom sistemu ekrana pozicija ovih koordinata fiksirana i predstavljena pikselima. [2]

Drugi način na koji se mogu posmatrati dostupni koordinatni sistemi je opisan u nastavku.

Naime, koordinatni sistem (u nastavku KS) prostora u kojem se osoba nalazi (odnosno KS prostorije) može biti označen sa K_0. Poziciju i orijentaciju oka u tom sistemu je moguće označiti kao tačku A (sa orijentacijom). Kamera koja snima oko (pojednostavljujući činjenicu da se to u ovom slučaju vrši posredstvom ogledala) se nalazi u nekoj nekoj tački B (sa orijentacijom) u sistemu K_0. Ona vidi oko u koordinatnom sistemu slike, koji će biti označen sa K_1, i koji je zapravo ravan (nema Z osu). Kamera koja snima ekran je analogno pretpostavljena u tački C. Kamera koja snima ekran i kamera koja snima oko su na poznatom fiksiranom rastojanju i u međusobnoj orijentaciji (geometriji), dakle tačke B i C su fiksne međusobno. Nadalje, tačke B i C su fiksne u odnosu na oko. Kamera koja snima ekran i sam ekran mogu biti posmatrani kao dva koordinatna sistema. Iz koordinatnog sistema kamere 2 (tačka C) nazvanim K_2, vidi se ekran (koordinatni sistem K_3, pravougaonik D1-2-3-4) na određeni način, a što je sve vidljivo iz koordinatnog sistema K_4 koji predstavlja sliku druge kamere.

5.1.10 Automatska rekalibracija

Za automatsku rekalibraciju je potrebno imati relaciju koja povezuje prostor oka sa prostorom ekrana, te konstantno računati matricu transformacije ukoliko dođe do pomijeranja pozicije ekrana u odnosu na početnu. To može biti zahtjevan posao u pogledu vremena izvršavanja, što može značajno usporiti sam program, te je potrebno utvrditi isplativost implementacije takvog postupka.

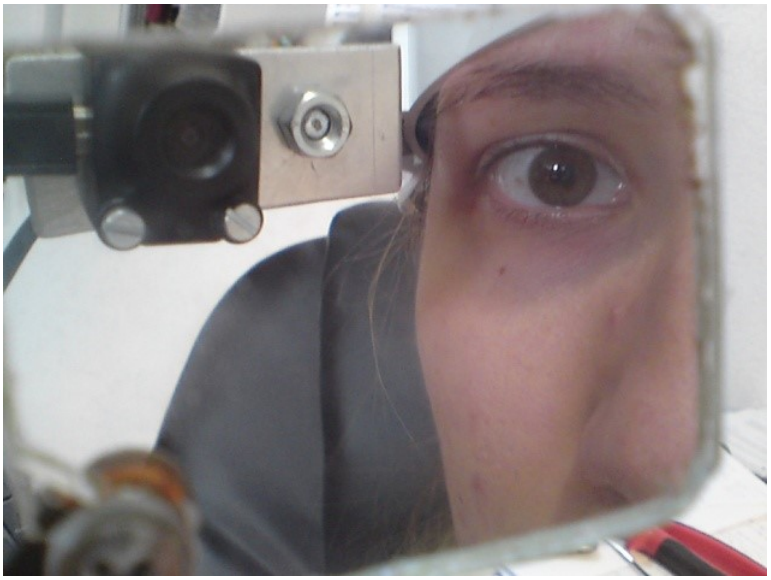
Poglavlje 6

Implementacija

U nastavku će biti opisana implementacija prethodno spomenutih metoda za obradu slike, te način realizacije istih u programu.

6.0.1 Ekstrakcija oka

Ekstrakcija oka se vrši sa slike koju snima prva kamera, a jedan prikaz takve slike je dat na slici 6.1.



Slika 6.1: Primjer slike koju snima prva kamera

Ovakva slika se prvo konvertuje u grayscale sliku pozivanjem funkcije:

```
cv::cvtColor(frame, grayscale, CV_BGR2GRAY)
```

Gdje je *frame* slika koju snima kamera 1, a *grayscale* slika u koju se sprema rezultat primjene konverzije slike prve kamere u grayscale sliku.

Na grayscale verziji slike se vrši izjednačavanje histograma pozivanjem funkcije:

```
cv::equalizeHist(grayscale, grayscale)
```

Pri čemu je iz poziva funkcije očigledno da se rezultat pozivanja sprema u varijablu grayscale.

Iz tako pripremljene slike je potrebno ekstraktovati poziciju oka, za što je potrebno prvo definisati vektor pravougaonika u koji će se pohranjivati detektovani pravougaonici. Nakon toga se primjenjuje funkcija `detectMultiScale` čiji je rezultat pozivanja smješten u pomenuti vektor, što je predstavljeno linijom koda:

```
eyeCascade.detectMultiScale(frame, eyes, 1.1, 2, 0 | CV_HAAR_SCALE_IMAGE,
                            cv::Size(150, 150))
```

Potrebno je izvršiti provjeru da li je oko detektovano, što je lako izvršiti provjerom veličine vektora u koji se smiješta rezultat pozivanja funkcije `detectMultiScale`. Očekuje se da će biti detektovano samo jedno oko, stoga veličina vektora treba biti jednaka jedinici. Ukoliko nije, flag greške se postavlja na jedinicu, te se prekida dalje izvršavanje funkcije, i ponovo se vrši detekcija oka. Flag greške se koristi za detekciju zatvorenog oka ili pomijeranja uređaja, zbog čega je potrebno izvršiti fizičko namještanje uređaja ili rekalibraciju.

Potrebno je sačuvati poziciju pravougaonika obzirom da je plan da se selekcija karakterističnih tačaka vrši na maloj slici, odnosno, slici na kojoj je prikazano samo oko, a ne kompletnoj slici prve kamere. Iz tog razloga je moguće na osnovu pozicije pravougaonika odrediti gdje se selektovane tačke nalaze na slici prve kamere.

6.0.2 Kreiranje maske

Kao što je objašnjeno u prethodnom poglavlju, nad slikom oka je potrebno vršiti operaciju aproksimacije pozicije zjenice, stoga je potrebno sliku oka konvertovati u grayscale sliku i izvršiti izjednačavanje histograma istim funkcijama kao što je to učinjeno nad slikom koju snima kamera 1. Kao rezultat se dobija slika prikazana na slici 6.2.

Slika 6.2: Primjer slike oka ekstraktovane iz slike koju snima prva kamera

Funkcija kojom se vrši omogućavanje selekcije tačaka na slici oka je:

```
setMouseCallback("Oko", CallBackFunc, NULL)
```

Funkcija je objašnjena u prilogu.

Potrebno je napomenuti da je implementacija takva da četiri karakteristične tačke budu određene na ekstraktovanoj slici oka obzirom da je lakše izvršiti označavanje na slici na kojoj je jasno prikazano samo i isključivo oko. Prije procesa označavanja dijela od interesa (beonjača), potrebno je nad slikom izvršiti konverziju iste u sivu monohromatsku sliku, te izvršiti izjednačavanje histograma, iz istih razloga zbog kojih je to izvršeno za sliku koju snima kamera 1. Na osnovu poznavanja informacija o poziciji pravougaonika na slici kamere, te poznavanju informacija o koordinatama karakterističnih tačaka oka na slici oka, moguće je odrediti dva luka koji aproksimiraju oko, pri čemu je značajan dio unutrašnjosti dva luka upravo beonjača oka.

Nakon što se barem jednom izvrši selekcija karakterističnih tačaka, vrši se skiciranje dvije parabole određene tim tačkama. Parametri parabola su određeni iz vrijednosti tačaka, te je izvršen prikaz parabola na obrađenoj slici koju snima prva kamera. Rezultat selekcije tačaka prikazan je na slici 6.3.

Slika 6.3: Primjer prikaza parabola na slici oka za odabir tačaka oka

Međutim, obrada slike se vrši na monohromatskoj slici. Jedan takav primjer parabola tad je na slici 6.4.

Slika 6.4: Primjer prikaza parabola na slici oka

Na osnovu formiranih lukova vrši se obrada slike, odnosno piksela slike, tako da su svi pikseli koji se nalaze na vanjskoj strani lukova vrijednosti koja predstavlja bijelu boju, odnosno vrijednosti 255. Ovdje se ogleda još jedna važnost dobijanja monohromatske slike iz slike koju snima kamera, obzirom da je mnogo lakše manipulirati sa jednokanalnim slikama i informacijama koje nose. Na osnovu selektovanog dijela beonjače, kreirana je maska koja ima bijelu pozadinu, a kao ROI prikazuje beonjaču oka. Primjer maske je prikazan na slici 6.5. Potrebno je naglasiti da se ovdje termin ”maska” razlikuje od klasičnog termina maske koja ima crnu pozadinu, osim u dijelu od interesa gdje je bijela, te je binarna slika.

Slika 6.5: Primjer maske oka

6.0.3 Segmentacija pragom

Pomenuto je da u OpenCV-u postoji nekoliko načina da za transformaciju monohromatske slike u binarnu sliku, stoga je dobro pogledati kako svaki od načina djeluje na sliku oka.

Algoritam estimacije zjenice je implementiran i vršen nad slikama oka, stoga će u nastavku na slikama biti prikazana i estimacija pozicije zjenice.

Jednostavno određivanje praga

Snimanje i testiranje thresholda je obavljano u uvjetima gdje je lice okrenuto ka izvoru prirodne svjetlosti, odnosno, prozoru.

Prvo je eksperiment izvođen sa jednostavnim thresholdingom kod kojeg je granična vrijednost unaprijed zadana. U tabeli 6.1 su prikazani rezultati za nekoliko threshold vrijednosti.

Tabela 6.1: Simple thresholding eksperiment

Vrijednost threshold granice	Monohromatska slika oka	Binarna slika oka	Zaključak
28			Pozicija zjenice je relativno dobro estimirana, ali novonastala binarna slika nije zadovoljavajuća zbog viška bijelih piksela.
30			Ponovo je estimacija zjenice veoma dobra, međutim, višak bijelih piksela je očigledan obzirom da do izražaja dolazi i odsjaj u oku.
39			Još uvijek je utjecaj odsjaja prozora oka vidljiv, što dovodi do odstupanja u estimaciji pozicije zjenice.
45			Da se primijetiti da je područje odsjaja prozora u oku smanjeno, s tim da su rezultati veoma slični prethodnom slučaju.
60			Iris je mnogo jasnije izražen, na estimaciju pozicije značajno utječe odsjaj u oku.
68			Oko je mnogo bolje vidljivo, što daje naznaku da bi ovakav threshold bio veoma dobar.
80			Za ovaj threshold previše do izražaja dolazi linija kapka koja može značajno utjecati na rezultate estimacije pozicije zjenice.
107			Za ovaj threshold do izražaja dolaze dijelovi koji nisu od interesa, što nije poželjno, stoga nije vršeno testiranje za veći threshold kada već ni vrijednost 107 nije zadovoljavajuća.

Iz eksperimenta je vidljivo da određivanje thresholda nije najjednostavnija stvar, te da osvjetljenje itekako utječe na threshold. Ukoliko bi threshold bio fiksan, pojavljivali bi se problemi kada bi se osvjetljenje naglo promijenilo, ili kada bi neki dijelovi slike bili osvijetljeni više ili manje od drugih. Iz tog razloga je potrebno imati neku vrstu adaptivnog thresholda koji će threshold prilagođavati trenutnoj situaciji u kojoj je snimljena slika. Iz tog razloga su u nastavku prikazani rezultati testiranja adaptivnog thresholda.

Adaptivno određivanje praga

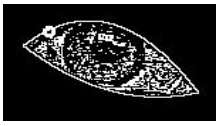
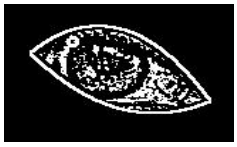
U sličnim uslovima je vršeno testiranje za adaptivni thresholding.

Obzirom da se obje vrste adaptivnog thresholdinga baziraju na sličnom principu, za očekivati je slične rezultate. Potrebno je napomenuti da je za ovu vrstu thresholdinga potrebno definisati određene parametre, što može predstavljati problem jer određivanje istih iziskuje određeni nivo ekspertize. Dakle, ova vrsta thresholdinga se bazira na metodi ”pokušaji i pogreške” što je vršeno ovim eksperimentom.

Eksperiment određivanja praga korištenjem ADAPTIVE_THRESH_MEAN_C

U tabeli 6.2 prikazani su rezultati eksperimenta za `ADAPTIVE_THRESH_MEAN_C`.

Tabela 6.2: Eksperiment određivanja praga korištenjem ADAPTIVE_THRESH_MEAN_C

Veličina područja	Konstanta C	Monohromatska slika oka	Binarna slika oka	Zaključak
3 x 3 piksela	5			Primijetimo da je korisna informacija potpuno izostavljena, te je nemoguće aproksimirati poziciju zjenice.
3 x 3 piksela	0			U odnosu na prethodnu sliku, ova ima potpuno crnu pozadinu što nije rezultat koji je trebalo postići. Nadalje, karakteristike oka nisu najjasnije vidljive, a samim tim je i zjenica pogrešno estimirana.
5 x 5 piksela	0			Rezultat primjene ovakvog thresholdinga je veoma sličan prethodnom, te je kao takav također neupotrebljiv za potrebnu svrhu.
7 x 7 piksela	0			Ponovo je rezultat sličan prethodnom, pa je dobro pomisliti da je problem u konstanti C koja u svim ovim slučajevima gdje je pozadina crna iznosi 0.
7 x 7 piksela	5			Rezultat je mnogo bolji nego u prethodnim slučajevima, premda još uvijek nije dovoljno dobar da bi zadovoljio uslov za dobru estimaciju pozicije zjenice.
11 x 11 piksela	5			U ovom slučaju počinju do izražaja dolaziti karakteristični dijelovi oka, međutim još uvijek rezultati nisu zadovoljavajući.
15 x 15 piksela	5			U odnosu na prethodni slučaj, mnogo je bolji kontrast i jasnije su vidljivi dijelovi oka, iako do izražaja dolaze dijelovi koji nisu od interesa za estimaciju zjenice.
15 x 15 piksela	20			Očigledno je konstanta C prevelika jer je mnogo više bijelih piksela u irisu, što onemogućuje ispravno određivanje pozicije zjenice.
15 x 15 piksela	-10			Za konstantu C jednaku -10 dobijamo rezultate slične onima kada je konstanta bila nula, što ukazuje na to da je ona sigurno veća od 0.

Iz sprovedenog eksperimenta je moguće zaključiti da ovakvo određivanje parametara uzima previše vremena, te očigledno ponovo ovisi od uslova snimanja, obzirom da nije u potpunosti automatizovano.

Eksperiment određivanja praga korištenjem ADAPTIVE_THRESH_GAUSSIAN_C

U nastavku su prikazani rezultati za adaptivnu metodu baziranu na Gaussovoj raspodjeli. Na slici 6.6 prikazani su rezultati za slučaj primjene pomenute metode thresholdinga za područje jednako 11 x 11 piksela, te za konstantu C jednaku 2.

(a) Monohromatska slika oka (b) Binarna slika oka

Slika 6.6: Rezultati prijeme Adaptive Gaussian Thresholding binarizacije

Očigledno je da su i ovim postupkom dobijeni rezultati slični onima kada se primijeni adaptive mean thresholding. Ovakav pristup ukazuje na to da je potrebno sprovesti eksperimente kojima bi se odlučile vrijednosti parametara veličine područja u pikselima i konstante C. Nadalje, moguće je zaključiti da je velika vjerovatnoća da nijedni parametri ne bi u potpunosti zadovoljavali uslove, obzirom da ovakav način binarizacije dovodi do izražaja sve osvijetljenje tačke slike, što nije poželjno u području irisa oka.

Preostao je Otsu thresholding koji automatski računa graničnu vrijednost. Eksperiment sa Otsu binarizacijom je prikazan u nastavku.

Otsu binarizacija

Testiranje Otsu binarizacije je vršeno u sličnim uvjetima u kojima su vršena i prethodna testiranja.

Na slici 6.7 prikazan je rezultat primjene Otsu binarizacije. Moguće je primijetiti da je rezultat veoma dobar, te je pozicija zjenice veoma dobro estimirana.

(a) Monohromatska slika oka (b) Binarna slika oka

Slika 6.7: Rezultati prijeme Otsu binarizacije

U slučaju kad oko nije idealno prikazano, već u ROI ulazi i dio kapka, Otsu binarizacija ponovo daje dobre rezultate estimacije zjenice, što je moguće vidjeti na slici 6.8.

(a) Monohromatska slika oka (b) Binarna slika oka

Slika 6.8: Rezultati prijeme Otsu binarizacije kada oko nije idealno prikazano

Potrebno je spomenuti da je estimacija zjenice vršena na slici na koju je prethodno primijenjen Gaussov filter objašnjen u nastavku.

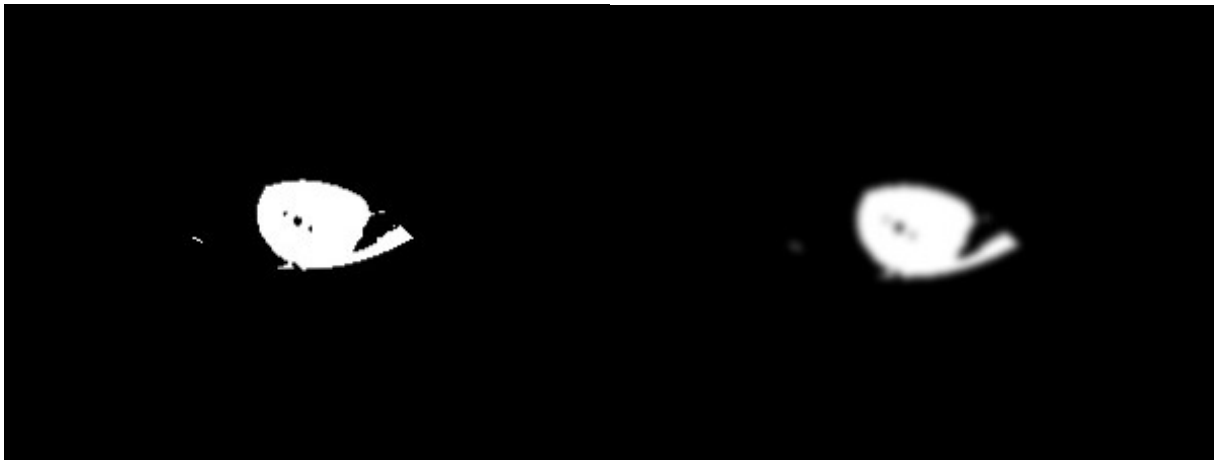
Korištenje Gaussovog filtera za eliminaciju lažnih ivica na slici oka

Gaussov filter predstavlja linearni filter implementiran u OpenCV-u koji služi za ”izglađivanje” (*engl. smoothing*) slika. Smoothing, odnosno blurring, je jednostavna i često korištena operacija u procesiranju slika. Postoji mnogo razloga za smoothing, od kojih je možda najbitnija ta što smanjuje šumove.

Detaljnije o Gaussovom filteru je moguće naći u prilogu.

Gassov filter je primijenjen na binariziranu sliku oka, sa Gaussovim kernelom veličine 9 x 9, i standardnim devijacijama u X i Y smjerovima objema jednakim 2.

Gaussov filter je primijenjen na inverznu binarnu sliku oka, obzirom da je algoritam zasnovan na najvećem zbiru, a vrijednost piksela bijele boje je 255, dok je vrijednost piksela crne boje 0, stoga je logično uraditi inverziju slike i nad takvom slikom vršiti algoritam. Slika oka prije nego je primijenjen blur na nju, te nakon što je primijenjen, prikazana je na slici 6.9.



(a) Binarna slika oka prije Gaussovog filtera (b) Binarna slika oka nakon Gaussovog filtera

Slika 6.9: Primjena Gaussovog filtera nad binarnom slikom oka

Sa prethodnih slika je vidljivo da se filtriranjem postiže u nekoj mjeri eliminisanje područja koja nisu od interesa, kao što su mali bijeli dijelovi u crnom dijelu slike, i sitni crni dijelovi u bijelom dijelu slike koji predstavlja iris. Na ovaj način se dobija slika nad kojom je bolje vršiti primjenu algoritma zbog svojih karakteristika.

Zaključak

Na osnovu sprovedenih eksperimenata, najboljom se pokazala upotreba Gaussovog filtera nad slikom u kombinaciji sa Otsu binarizacijom slike. Na ovaj način threshold se automatski računa, pa nagla promjena svjetlost ili različito osjetljenje slike po dijelovima, može imati manje utjecaja.

6.1 Estimacija pozicije zjenice

Sa prethodnih slika se jasno moglo vidjeti da dobrim odabirom metoda transformacije monohromatske slike oka u binarnu, te primjenom algoritma za estimaciju pozicije zjenice, dobija se relativno dobra aproksimacija zjenice.

Estimacija pozicije zjenice ovisi i od toga gdje se oko nalazi, kao i od osvjetljenja. Osvjetljenje je najbolje da bude takvo da je izvor svjetlosti okrenut ka licu, stoga je ta praksa zadržana kroz sve sprovođene eksperimente. U nastavku su prikazani rezultati estimacije zjenice na oku koje gleda u različitim smjerovima. Estimacija je primijenjena nad slikom oka nad kojom je prvo primijenjena Otsu binarizacija, te nakon toga Gaussov filter.

Rezultati estimacije pozicije zjenice prikazani su u tabeli 6.3.

Tabela 6.3: Estimacija pozicije zjenice

Usmjerenost pogleda	Monohromatska slika oka	Binarna slika oka
pravo		
udesno		
ulijevo		
nadolje		

Od velikog značaja je vidjeti ponašanje estimacije zjenice kada se iris nalazi u uglovima oka. Za očekivati je da je tada estimacija pozicije zjenice teža, a razlog za odabir pomenutog algoritma određivanja pozicije zjenice leži upravo u činjenici što se oslanja na maksimalnu sumu piksela po redovima i po kolonama, a ne na izdvajanje karakterističnog oblika irisa, što znači da algoritam treba raditi i u uvjetima kada oko dostiže granice pokretljivosti.

Rezultati estimacije zjenice kada je iris u lijevom uglu oka, prikazana je na slici 6.10.

(a) Monohromatska slika oka (b) Binarna slika oka

Slika 6.10: Estimacija zjenice kada je iris u lijevom uglu oka

Da se primijetiti da je estimacija pozicije zjenice prilično dobra obzirom da je iris u uglu oka.

Rezultati estimacije zjenice kada je iris u desnom uglu oka prikazana je na slici 6.11.

(a) Monohromatska slika oka (b) Binarna slika oka

Slika 6.11: Estimacija zjenice kada je iris u desnom uglu oka

Iris nije u potpunosti u desnom uglu obzirom da je fizički veoma teško gledati sa takvom pozicijom oka. Potrebno je napomenuti da desni i lijevi ugao odgovaraju orijentaciji slike, a ne stvarnog oka.

Iz prethodnih slika je vidljivo da je estimacija zjenice veoma dobra, premda postoje odstupanja koja su uzrokovana upravo metodom estimacije zjenice, te veličinom maske i onoga što ona obuhvata.

6.2 Rotacija slike oka

Obzirom da se pozicija zjenice računa na osnovu najveće sume piksela po kolonama i po redovima, orijentacija oka na slici utječe na određivanje pozicije zjenice. Na slikama prethodno prikazanim to nije došlo do prevelikog izražaja obzirom da je maska bila veoma dobro odabrana i u većini slučajeva je bilo moguće vidjeti samo beonjaču oka. U slučaju kada u masku ulaze i druge oblasti oka poput trepavica, ili u slučaju kada sjena padne na oko tako da stvara problem i daje lažne rezultate, može doći do problema pri aproksimaciji pozicije zjenice.

Iz tog razloga je poželjno rotirati sliku beonjače i nad rotiranom slikom vršiti estimaciju pozicije zjenice. Rotacija se vrši određivanjem ugla rotacije iz karakterističnih tačaka oka, tačnije, iz koordinata uglova oka. Na osnovu izračunatog ugla nagiba oka, oko se rotira tako da je linija koja spaja krajeve oka, ravna linija bez nagiba.

Primjer rotacije jedne binarizirane slike oka dat je na slici 6.12.

(a) Nakrivljena slika oka (b) Ispravljena slika oka

Slika 6.12: Rotacija slike oka

U tabeli 6.4 su prikazani rezultati estimacije zjenice u slučaju kada se vrši rotacija slike oka.

Tabela 6.4: Estimacija pozicije zjenice za slučaj rotacije slike oka

Monohromatska slika oka	Nerotirana binarna slika oka	Rotirana binarna slika oka sa procjenom pozicije zjenice
		
		
		
		
		

Na posljednjoj slici je maska takva da ne obuhvata cijelu zjenicu, ali je vidljivo da je ipak procijenjena zjenica unutar irisa. Već je iz svih navedenih rezultata estimacije pozicije zjenice vidljivo da metod nije baš najbolji te bi princip procjene pozicije zjenice trebao biti dosta rigorozniji kako bi se izbjegle greške nastale zbog utjecaja dijelova oka koji ne bi trebali biti u ROI. Teško je procijeniti da li je rotacija slike dovela do značajnog poboljšanja u estimaciji pozicije zjenice obzirom da je generalno estimacija na pomenuti način veoma nepredvidljiva i nesigurna.

6.2.1 Poređenje estimacije pozicije zjenice sa i bez rotacije slike

U nastavku će biti prikazani rezultati estimacije zjenice kada je slika rotirana, te kada nije. Ujedno će biti prikazana i stvarna slika oka kako bi se procijenila tačnost određivanja pozicije zjenice.

U tabeli 6.5 prikazane su monohromatska slika oka, nerotirana binarna slika oka sa procijenjenom pozicijom zjenice i rotirana binarna slika oka sa procijenjenom pozicijom zjenice.

Tabela 6.5: Estimacija pozicije zjenice za slučaj rotirane i nerotirane slike oka

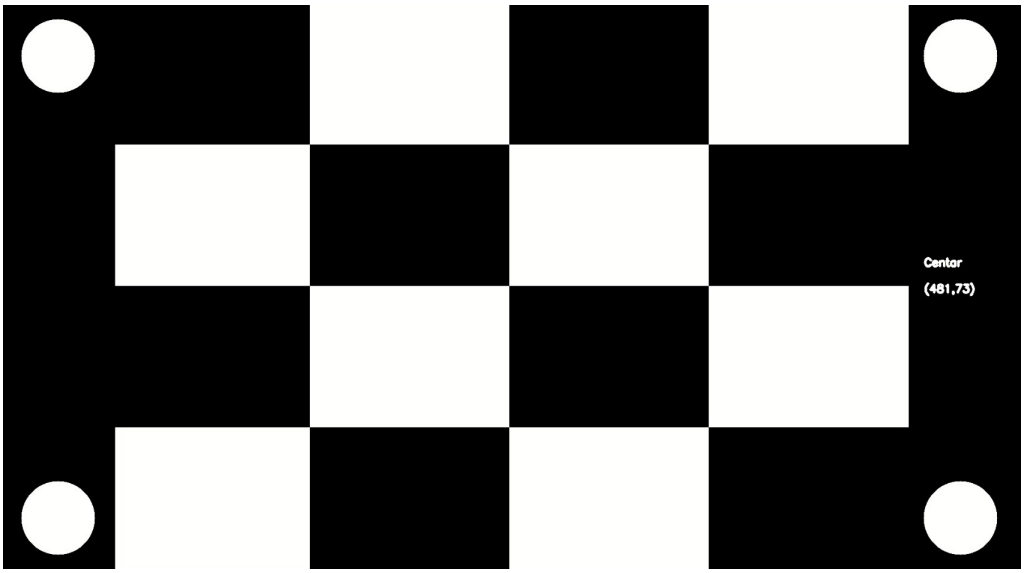
Monohromatska slika oka	Nerotirana binarna slika oka sa procjenom pozicije zjenice	Rotirana binarna slika oka sa procjenom pozicije zjenice
		
		
		
		
		
		

Prikazane slike najbolje pokazuju neizvjesnost u procjeni pozicije zjenice oka. Naime, jasno je vidljivo da je u nekim situacijama rotacija dobro rješenje, kao što je to u slučaju pete i šeste slike, dok se iz druge slike može pomisliti upravo suprotno. Na ostalim slikama je odstupanje od stvarne pozicije zjenice približno isto za rotiranu i nerotiranu sliku, s tim da je odstupanje u različitu stranu. Kod je prilagođen situaciji sa rotiranom slikom oka, iako je potrebno još jednom naglasiti da nije dokazano kako je taj postupak upotpunosti bolji od onoga bez rotacije, ali je logičniji.

6.3 Detekcija ekrana

Detekcija ekrana je zamišljena kao detekcija četiri kružnice bijele boje u uglovima ekrana na crnoj pozadini. Razlog za odabir bijelih kružnica leži u činjenici da kamera 2 neće biti u potpunosti paralelna ekranu, stoga će pod utjecajem osvjetljenja neki dijelovi ekrana biti manje, a neki više osvijetljeni. Iz tog razloga je veoma teško detektovati krugove koji su različite boje od bijele, odnosno teško je raditi detekciju krugova zasnovanu na boji kruga, a ne na samom obliku upravo zbog osvjetljenja.

Slika na kojoj se bazira detekcija ekrana se nalazi na čitavom ekranu i prikazana je na slici 6.13.



Slika 6.13: Slika za detekciju ekrana

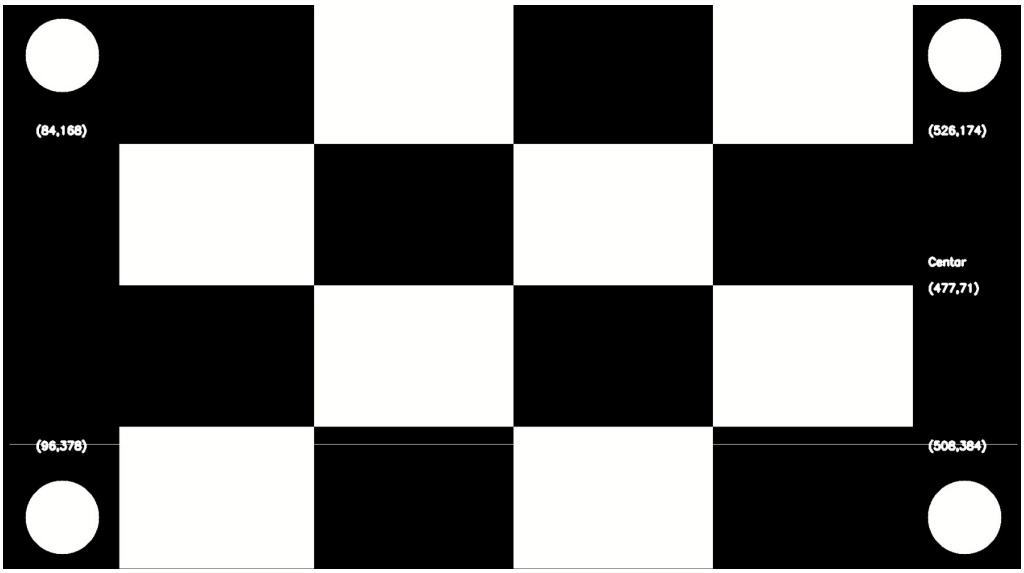
Da bi ova četiri kruga bila dobro i tačno detektovana, potrebno je podesiti parametre funkcije `HoughCircles`. Metodom pokušaja i pogrešaka, došlo se do zaključka da je najbolji odabir parametara sljedeći:

- **dp** = 1
- **min_dist** = 100
- **param_1** = 200 (gornja granica za Canny detektor)
- **param_2** = 30 (granica za detekciju centra)

Parametri za minimalni i maksimalni radijus su nula obzirom da nije poznata ta veličina zbog udaljenosti osobe od ekrana. Fleksibilnost u veličini krugova dozvoljava da se osoba pomijera nekoliko centimetara naprijed i nazad, a da krugovi i dalje budu detektovani.

Za detekciju krugova je prvo potrebno obraditi sliku koju snima kamera 2, odnosno, sliku je prvo potrebno konvertovati u grayscale sliku, a zatim na tako dobijenu sliku primijeniti Gaussov filter koji će izvršiti blurring slike i omogućiti lakšu detekciju krugova.

Ukoliko dođe do detekcije sva četiri kruga, na slici 6.13 će ispod svakog kruga biti prikazana koordinata njegovog centra na slici koju snima kamera 2. Rečeno je prikazano na slici 6.14.

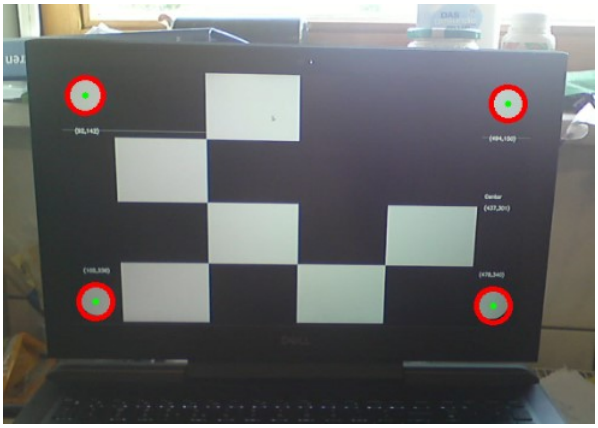


Slika 6.14: Slika za detekciju ekrana sa prikazanim koordinatama centara krugova

Ukoliko nisu detektovana sva četiri kruga, ispod krugova neće biti napisana pozicija njihovog centra.

Procjena koja koordinata, odnosno, koji detektovani krug pripada kojem dijelu ekrana obavlja se jednostavnom provjerom centara kružnica, znajući dimenziju slike koju snima kamera 2.

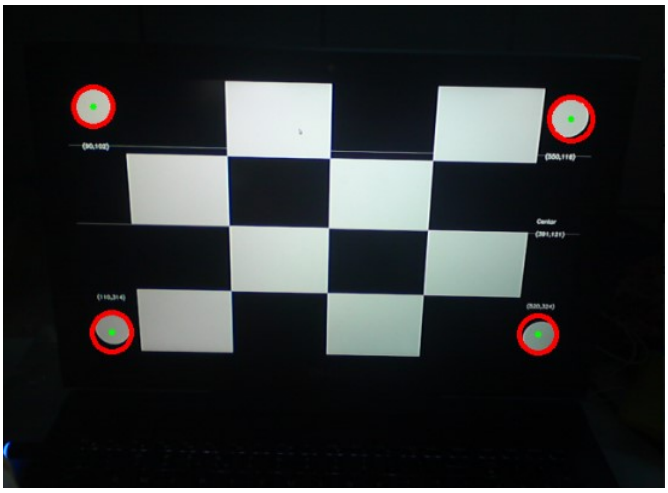
Na slici 6.15 prikazana je slika koju snima kamera 2, na kojoj se jasno vidi detekcija četiri kruga prikazanih na slici 6.14.



Slika 6.15: Slika koju snima kamera 2 na kojoj su detektovana sva četiri kruga

Koordinate ovako detektovanih krugova se ispisuju na ekranu ispod svakog od krugova.

Na slici 6.16 vidljivo je koliko osvjetljenje utječe na detekciju krugova stvarajući sliku sa više kontrasta. U pomenutom slučaju je u potpunosti uklonjeno osvjetljenje prema ekranu, te samo postoji osvjetljenje iza ekrana.



Slika 6.16: Slika koju snima kamera 2 bez direktnog osvjetljenja ekrana

6.4 Estimacija usmjerenosti pogleda

Da bi bilo uopće moguće procijeniti gdje osoba gleda potrebno je naći vezu između pozicije zjenice i kvadrata. Na slikama 6.13 i 6.14 moguće je vidjeti, pored krugova, i kvadrate crne i bijele boje koji tvore sliku nalik na šahovsku ploču dimenzija 4 x 4.

Pretpostavka je da je podjela na 16 dijelova premala za detekciju usmjerenosti pogleda, ali je potrebno to i eksperimentalno dokazati. Snimana je pozicija zjenice oka dok je osoba gledala kvadratiće redom odozgo nadolje i slijeva nadesno, počevši od prvog u gornjem lijevom uglu. Koordinate karakterističnih tačaka oka date su u tabeli 6.6.

Tabela 6.6: Estimacija koordinate zjenice kada je pogled usmjeren ka prvom kvadratu

Koordinate karakterističnih tačaka oka (okvir maske)
Krajnja lijeva tačka oka: (46, 89)
Krajnja desna tačka oka: (180, 111)
Sredina luka gornjeg kapka: (97, 63)
Sredina luka donjeg kapka: (124, 118)

Za pomenute koordinate karakterističnih tačaka oka, snimana je estimacija pozicije zjenice dok osoba gleda u 16 različitih kvadratića prikazanih na zaslonu. Rezultati estimacije pozicije zjenice dati su u tabeli 6.7.

Tabela 6.7: Rezultati eksperimenta estimacije zjenice tokom gledanja 16 kvadratića

Kvadrat	Estimacija pozicije zjenice na slici oka sa koje se ista estimira
Prvi	(417, 302), (416, 302), (418, 302)
Drugi	(413, 301), (419, 302), (420, 302), (419, 301)
Treći	(421, 302), (422, 301)
Četvrti	(425, 301), (426, 301), (424, 302), (427, 300)
Peti	(416, 302), (417, 302)
Šesti	(418, 302), (419, 302)
Sedmi	(421, 302), (420,302)
Osmi	(424, 302), (423, 302)
Deveti	(438, 302), (417, 302)
Deseti	(441, 302), (442, 302), (440, 302)
Jedanaesti	(444, 302), (443, 302)
Dvanaesti	(400, 304), (447, 301), (447, 302), (448, 302), (449, 302)
Trinaesti	(440, 301), (439, 302)
Četrnaesti	(442, 301), (442, 302)
Petnaesti	(445, 301), (444, 301), (446, 301)
Šesnaesti	(448, 301), (449, 301), (448, 300), (447, 301)

Iz tabele 6.7 jasno je moguće vidjeti probleme koji nastaju prilikom estimacije pozicije zjenice. Naime, jasno je moguće vidjeti kako estimirana koordinata zjenice pripada više skupova kvadratića. Iz tabele je također jasno moguće vidjeti opravdanost podjele ploče umjesto na 16 na 4 veća kvadrata, obzirom da zjenica ima slične koordinate za prvi, drugi, peti i šesti kvadratić. Analogno vrijedi i za ostale kvadratiće koji čine četvorku koja čini veći kvadrat. Očigledno je da nema većih promjena u y koordinati zjenice, što može predstavljati problem prilikom određivanja u koji kvadrat osoba gleda.

U slučaju kada osoba gleda u uglove ploče, rezultati za estimaciju pozicije zjenice su dati u tabeli 6.8.

Tabela 6.8: Estimirana vrijednost koordinata zjenice kada osoba gleda u uglove najvećeg kvadrata

Ugao	Estimirana vrijednost koordinate zjenice
Gornji lijevi	(416, 302), (416, 303)
Gornji desni	(447, 301), (448, 301)
Donji lijevi	(436, 302), (439, 302)
Donji desni	(450, 300), (449, 300)

Iz tabela 6.7 i 6.8 moguće je bolje primijetiti kojim vrijednostima, odnosno, intervalima pripadaju y vrijednosti koordinata zjenice. Također, bitno je primijetiti mnogo veću razliku u x koordinatama za tabelu 6.8, nego u tabeli 6.7. Ovakav eksperiment samo potvrđuje činjenicu da je određivanje pozicije zjenice veoma neprecizno, pogotovo ukoliko osoba gleda prema donjim uglovima, time dovodeći oko u položaj u kojem je teško estimirati poziciju zjenice.

Bitno je napomenuti da je eksperiment izvođen u uslovima u kojim je vrat osobe bio potpuno prav, odnosno, ekran je bio u visini očiju, čime je postignuto da je oko većinom bilo otvorenije.

Iz sprovedenog eksperimenta je moguće zaključiti da je podjela na četiri kvadrata bolja nego podjela na 16 kvadrata, međutim, i manji broj od 4 bi vjerovatno bio još bolji. Najbolji slučaj je onaj u kojem su područja u koja osoba gleda veoma razmaknuta, tada dolazi do izražaja različitost koordinata zjenice.

6.4.1 Određivanje dijela ekrana prema kojem je usmjeren pogled osobe

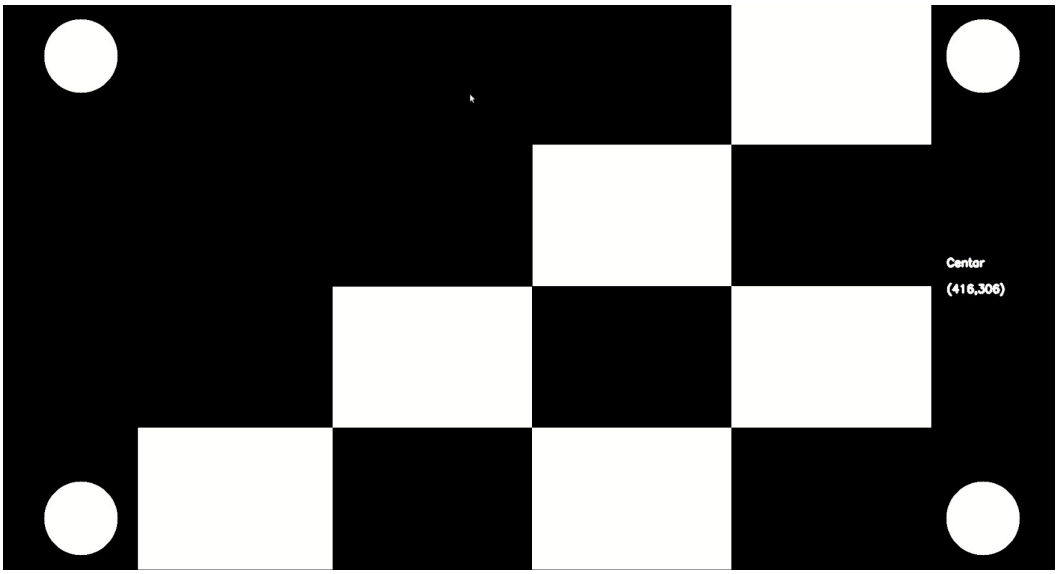
Metod opisan u prethodnom poglavlju nije korišten obzirom da zahtjeva računanje inverznih matrica velikih dimenzija, što značajno usporava već dovoljno spor program. Također je pomenuti algoritam dobro primjenjivati samo u slučaju kada je pozicija zjenice određena precizno, što ovdje nije slučaj. Opisana metoda korištenjem matrice homogene transformacije se primjenjuje i za slučaj autorekalibracije, koja također nije urađena iz istog razloga.

Matematička pozadina određivanja dijela ekrana prema kojem je usmjeren pogled zasnovana je na činjenici da se na početku programa vrši kalibracija, te svaki put kada korisnik želi promijeniti karakteristične tačke oka.

Tokom kalibracije, osoba gleda u uglove slike nalik šahovskoj ploči, te se prilikom gledanja u svaki od uglova kreira relacija između prostora slike kamere 2 i estimirane pozicije zjenice. Granica koja se provjerava prilikom određivanja tačnog kvadrata u koji osoba gleda iznosi polovinu udaljenosti između koordinata koje ima zjenica tokom gledanja dva susjedna ugla (uglovi su susjedni ukoliko se između njih nalaze dva crna i dva bijela kvadratića). Uzimanje polovine udaljenosti za granicu leži u činjenici da je granica promjene gledišta sa jednog velikog kvadrata na drugi upravo na sredini udaljenosti između dva ugla. Očigledno je da nije implementirana matrica homogene transformacije iz razloga objašnjenog ranije, ali je potrebno napomenuti kako je kalibracija implementirana u programu, te se ostavlja prostor za kasnijom implementacijom procjene tačke gledišta korištenjem homogene transformacije.

Postavljanjem jednostavnih uslova na koordinate zjenice moguće je procijeniti prema kojem velikom kvadratu je trenutno usmjeren pogled osobe. Da bi ovakav algoritam funkcionisao, potrebno je pokazati da je aproksimativna granica u prostoru oka između dva velika kvadrata na sredini udaljenosti bilo x ili y koordinata zjenice, bilo da se radi o promjeni x ili y koordinate, tokom kalibracije. Pomenuto se može zaključiti iz prethodnog eksperimenta, što je opravdanje za korištenjem takvog algoritma za provjeru usmjerenosti pogleda.

Ukoliko je rezultat algoritma takav da je zaključeno da osoba gleda u gornji lijevi kvadrat kojeg čine četiri manja kvadratića, taj dio će u potpunosti postati crn, što je prikazano na slici 6.17.



Slika 6.17: Estimacija usmjerenosti pogleda ka velikom kvadratu u gornjem lijevom uglu

Na slici 6.17 nisu prikazane koordinate krugova obzirom da je položaj kamere 2 bio takav da nije bilo moguće detektovati sva četiri kruga.

Analogan postupak je urađen i za preostala tri velika kvadrata. Rezultati eksperimenta pokazuju da je velika greška pri određivanju usmjerenosti pogleda ka trećem i četvrtom kvadratu (dva donja kvadrata), što je vjerovatno rezultat pogrešnog određivanja pozicije zjenice tokom kalibracije. Rezultati nisu 100% zadovoljavajući, ali je moguće primijetiti da postoji potencijal za razvijanje i unapređivanje pomenute metode.

6.4.2 Automatska rekalibracija

Automatska rekalibracija nije implementirana prvenstveno iz razloga što samo određivanje usmjerenosti pogleda ne radi idealno ni kada osoba u potpunosti miruje. Nadalje, vrijeme izvršavanja programa je mnogo usporeno, stoga bi implementacija automatske rekalibracije mogla rezultirati dodatnim usporavanjem programa. Autorekalibracija je bazirana na računanju matrice transformacije, a za samo određivanje koeficijenata iste potrebno je vršiti računarski zahtjevne operacije, kao što je pronalazak inverzne matrice dimenzija 4 x 4. Eksperimentima je utvrđeno da pomijeranje osobe od nekoliko centimetara u svim smjerovima ne predstavlja veliki problem i procjena usmjerenosti pogleda će biti relativno tačna ukoliko je tačna i kada osoba miruje. Premda automatska rekalibracija nije implementirana, detekcija ekrana nije uklonjena zbog mogućih kasnijih poboljšanja algoritma koji će koristiti istu.

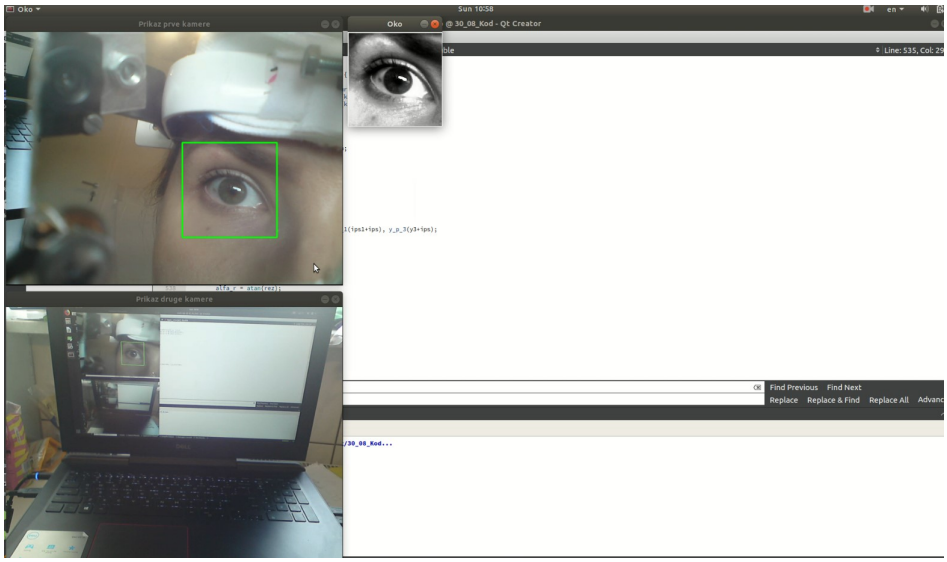
Poglavlje 7

Rezultati i primjena

7.1 Primjena

U prethodnom odjeljku je objašnjenja implementacija ideje uvedene ranije. U ovom odjeljku će biti pokazana praktična primjena svih pomenutih aspekata, te generalno rad osmišljenog uređaja.

Pokretanjem programa, na zaslonu ekrana se pojavljuju tri slike. Jedna predstavlja sliku kamere 1, druga sliku kamere 2, dok je treća i najmanja od tri slike slika oka koje se detektuje sa slike koju snima kamera 1. Primjer izgleda ekrana nakon pokretanja programa prikazan je na slici 7.1.



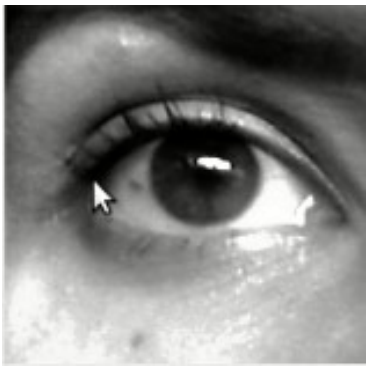
Slika 7.1: Izgled ekrana nakon pokretanja programa.

Kao što je pomenuto u eksperimentima izvođenim ranije, poželjno je da lice osobe bude usmjereno ekranu tako da vratni stub korisnika ispravljen, odnosno, lice je u visini ekrana. U slučaju rezultata prikazanih u ovom poglavlju, pozicija lica viđenog od strane kamere laptopa u vrhu ekrana, prikazana je na slici 7.2.

Slika 7.2: Lice osobe viđeno kroz objektiv kamere laptopa

Detekcija oka se puno brže izvršava nego prikazivanje male slike oka, stoga se u nekim momentima slika oka u zelenom pravougaoniku na slici koju snima kamera 1 i slika oka je izdvojena zasebno u monohromatskoj verziji, mogu razlikovati. Iz tog razloga je bitno pratiti izgled oka na slici koju snima kamera 1 obzirom da se detekcija i prikaz brže na istoj izvršavaju, te u pogodnom trenutku, kada detektovano oko izgleda spremno za selektovanje karakterističnih tačaka (oko je spremno za selektovanje karakterističnih tačaka onda kada je na slici prikazano cijelo oko sa jasno vidljivim tačkama koje treba selektovati), potrebno je pritisnuti na tastaturi tipku 'SPACE' nakon čega se trenutna slika zaustavlja kako na prikazu kamere 1 tako i na prikazu kamere 2, i monohromatske slike oka, pri čemu su sada monohromatska slika oka i slika oka u zelenom pravougaoniku na slici kamere 1, jednake.

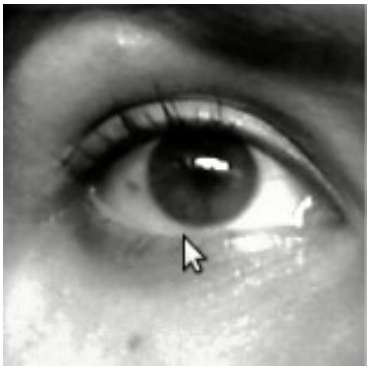
Nakon zaustavljanja željenog izgleda oka (pri čemu se preporučuje da osoba neko vrijeme mirno gleda u jednu tačku na taj način obezbjeđujući dobru sliku za selekciju karakterističnih tačaka), potrebno je izvršiti selekciju četiri karakteristične tačke oka, krenuvši od ugla beonjače oka koji se nalazi krajnje lijevo na ekranu, pri čemu je to prva tačka, dalje se krečući prema sredini gornjeg kapka, pri čemu to predstavlja drugu tačku, treća tačka je krajnji desni ugao beonjače oka, dok je četvrta tačka sredinja donjeg kapka oka. Odabir karakterističnih tačaka prikazan je na slici 7.3. Bitno je napomenuti da se odabir tačaka ispisuje tokom izvršavanja, te ukoliko se desi da osoba ne selektuje tačke ispravnim redoslijedom, program se zaustavlja. Razlog za zaustavljanje leži u činjenici da je moguće znati koja tačka je na redu za odabir. Prilikom odabira karakterističnih tačaka bitno je pažnju usmjeriti na odabir tačaka tako da je ROI beonjača oka, a što manje dijelovi van nje.



(a) Prva tačka

(b) Druga tačka

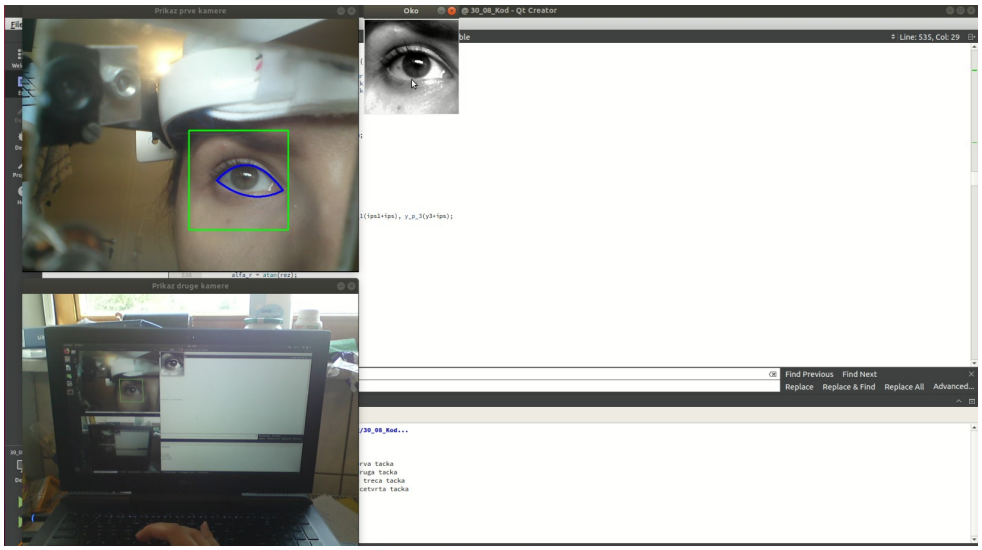
(c) Treća tačka



(d) Četvrta tačka

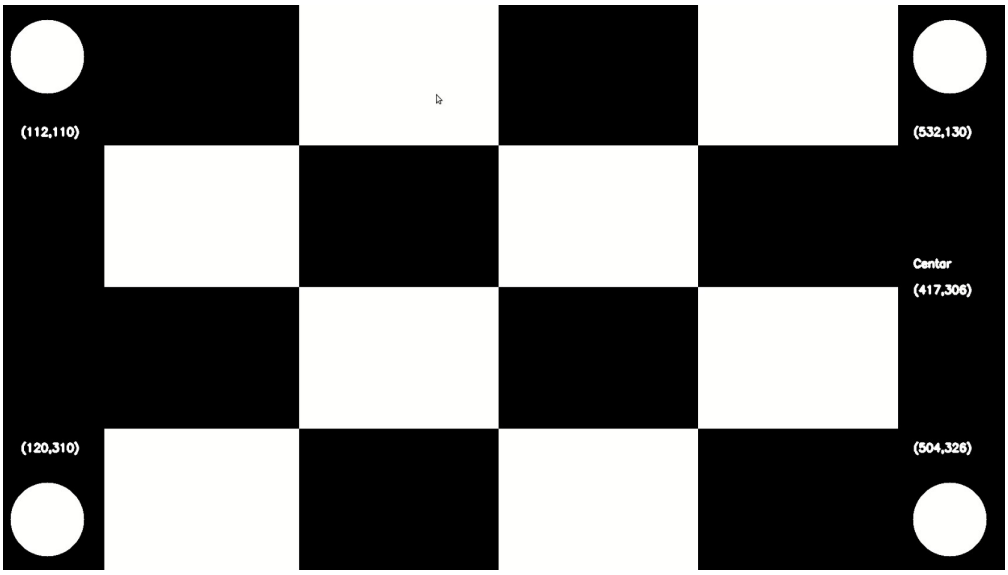
Slika 7.3: Selekcija četiri karakteristične tačke oka klikom na lijevi klik miša

Nakon odabira karakterističnih tačaka, kreira se maska na slici kamere 1, što produžava vrijeme izvršavanja programa, i prolongira prikaz slike sa krugovima i dijelom nalik na šahovsku ploču. Zaslon ekrana neto prije prikaza pomenute slike, vidljiv je na slici 7.4.



Slika 7.4: Zaslon ekrana netom nakon odabira svih karakterističnih tačaka

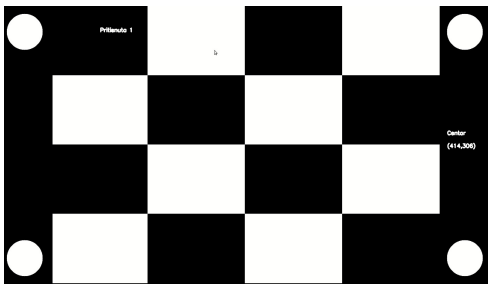
Na zaslonu se, nakon svega pomenutog, pojavljuje slika sa četiri kruga u uglovima, te središnjim dijelom nalik na šahovsku ploču, što je prikazano na slici 7.5. Bitno je primijetiti da su na pomenutoj slici detektovana sva četiri bijela kruga, stoga njihova pozicija na slici kamere 2 stoji napisana ispod svakog od njih.



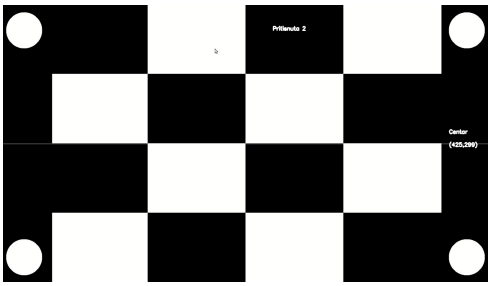
Slika 7.5: Zaslon ekrana nakon odabira svih karakterističnih tačaka

U ovom trenutku je potrebno izvršiti kalibraciju kako bi bilo moguće napraviti relaciju između pozicije zjenice i kvadrata u koji osoba gleda. Tokom kalibracije se od korisnika traži da gleda četiri ugla središnje slike, krenuvši od gornjeg lijevo i krečući se u smjeru kazaljke na satu ka donjem desno. Prilikom gledanja u jedan od uglova potrebno je zadržati pogled dovoljno dugo na istom tako da se estimacija centra irisa, koja je prikazana na ekranu na desnoj strani između dva kruga, mijenja što manje. Kada se to ostvari, potrebno je pritisnuti na tastaturi broj koji odgovara uglu u koji osoba gleda, pri čemu je gornjem lijevom uglu dodijeljen broj '1', a donjem desnom uglu broj '4', a ostalim uglovima analogno brojevi '2', odnosno '3', onim redom kako se pojavljuju. Između odabira brojeva na tastaturi treba postojati značajna pauza kako bi estimirana pozicija zjenice bila što vjerodostojnija.

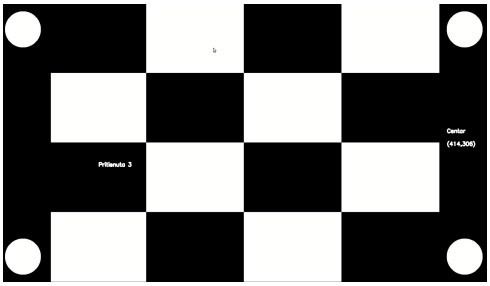
Prilikom kalibracije, nakon svakog pritiska na dugme tastature koje odgovara brojevima od 1 do 3, prikazuje se poruka koja govori koji je broj pritisnut, što je prikazano na slici 7.6.



(a) Prvi ugao - pritisnuta jedinica



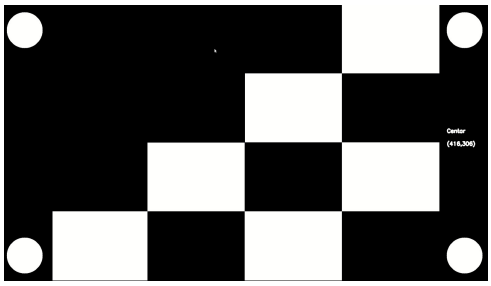
(b) Drugi ugao - pritisnuta dvica



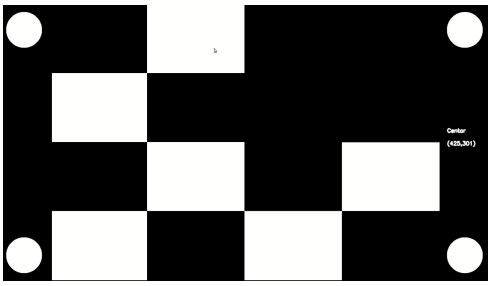
(c) Treći ugao - pritisnuta trica

Slika 7.6: Selekcija četiri karakteristične tačke oka klikom na lijevi klik miša

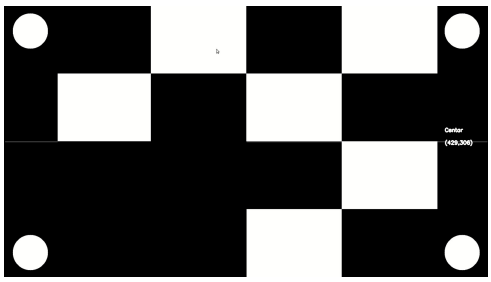
Nakon kalibracije, vrši se procjena usmjerenosti pogleda kao što je objašnjeno u prethodnim odjeljcima. Prilikom kalibracije i gledanja u četvrti ugao, nakon odabira broja četiri automatski započinje procjena usmjerenosti pogleda, a obzirom da je pogled već usmjeren na četvrti ugao, poruka o pritisnutom broju četiri se ne ispisuje jer je pokriva veliki crni pravougaonik koji označava da osoba gleda ka četvrtom velikom kvadratu. Svaki kvadrat se sastoji od 4 manja kvadrata, dva crne i dva bijele boje, stoga ukoliko osoba gleda ka nekom kvadratu, taj dio će u potpunosti biti crn, kao što je prikazano na slici 7.7.



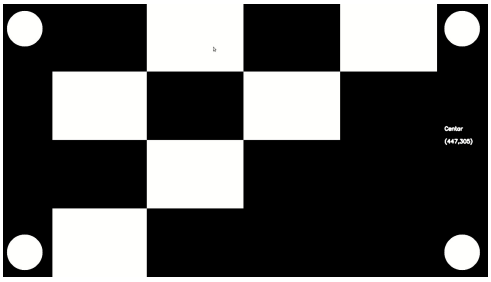
(a) Osoba gleda u prvi veliki kvadrat



(b) Osoba gleda u drugi veliki kvadrat



(c) Osoba gleda u treći veliki kvadrat



(d) Osoba gleda u četvrti veliki kvadrat

Slika 7.7: Određivanje usmjerenosti pogleda

7.2 Rezultati

Obzirom da rad programa zahtjeva obavljanje višestrukih operacija obrade slike, značajno se istima usporava izvršavanje programa. Iz tog razloga se tokom izrvšavanja programa nerijetko dešava da trenutni prikaz slike koju snima kamera 1 i prikaz oka ekstraktovanog iz te slike, ne budu isti.

Na rad algoritma za određivanje pozicije zjenice utječu mnogi faktori, od kojih su i taj kako će biti odabrana maska. Iz tog razloga je veoma bitno da osoba koja koristi program ima nivo ekspertize kako bi mogla selektovati tačke koje daju zadovoljavajuču masku.Tokom eksperimenata je primijećeno da lošija maska daje znatno lošije rezultate procjene pozicije zjenice oka.

Najbitniji faktor koji znatno utječe na rezultate je svakako osvjetljenje. Osvjetljenje je najveći nedostatak opisanog pristupa rješavanja problema, obzirom da je nešto na što se ne bi trebalo puno utjecati, ali je očigledno potrebno uzimati ga u obzir. U ovisnosti od osvjetljenja, oko će biti bolje ili lošije prikazano kao binarna slika, a samim tim se utječe na procjenu pozicije zjenice.

Tokom izvođenja iznad spomenutog eksperimenta, nakon kalibracije je bilo veoma teško pogled zadržati na trećem kvadratu. To govori da je i kalibracija od ključnog značaja, stoga je potrebno ostaviti vremena između dva odabiranja, te ujedno za kalibrisanu tačku odabrati onu koja nije izuzetak koji se pojavio tokom procjene pozicije zjenice oka, već uistinu raciju realistična tačka za poziciju zjenice oka u odnosu na tačku koju gleda.

Rezultati su znatno bolji ukoliko osoba tokom odabiranja nekog od kvadrata ne gleda u čitavu površinu kojoj pripada taj kvadrat, već u uglove u koje je gledala tokom kalibracije. Primijećeno je da što je veći razmak između tačaka u koje osoba gleda, to je veća vjerovatnoća da će estimirana pozicija zjenice oka biti tačna.

Obzirom da automatska rekalibracija nije implementirana, kretanje osobe je ograničeno na kretanje unazad ili unaprijed paralelno poziciji u kojoj je bila tokom kalibracije. Ovime se omogućava kretanje osobe u odnosu na referentnu tačku za cca. 10 cm, što je dovoljno s obzirom da je pretpostavljeno da se osoba neće značajno ni kretati. Također je moguće pomijerati glavu ustranu, s tim da je potrebno takve pokrete minimizirati obzirom da se gubi referenca utvrđena tokom kalibracije.

Na kraju je najbitnije biti u mogućnosti zadržati pogled na nekom od kvadratića dovoljno dugo, obzirom da takvog nešto može biti signal da osoba želi upravo taj kvadrat i da je taj kvadrat onaj u koji gleda, što je bilo moguće u sprovođenim eksperimentima (pod uslovom da su tokom određivanja maske i kalibracije bile pravljene minimalne greške).

Zaključak

Određivanje pozicije zjenice oka korištenjem neinvazivnih metoda nije nimalo lagan posao i oslanja se isključivo na obradu slike. Poznato je da na obradu slike utječe prvenstvno nivo ekspertize osobe koja obavlja obradu, te nakon toga vanjski utjecaji, kao što su kvaliteta kamere, ambijentalni uslovi, pozicije kamere u odnosu na dio koji je potrebno snimati, i slično.

U ovom radu je korištena metoda procjene zjenice oka koja nije pronađena niti u jednom drugom obrađenom dokumentu korištenja pokreta oka za upravljenje računarom. Bilo je potrebno pokazati da li u ovaj tip metode vrijedi ulagati napore u razvijanju boljih načina estimacije pozicije zjenice, ili ne. Iz eksperimenata koji su sprovođeni je jasno da je za bolje rezultate potrebno imati dvije osobe koje su uključene u proces rada programa, od kojih je jedna korisnik, a druga osoba koja upravlja programom, pri čemu osoba koja upravlja programom ima određeni nivo ekspertize. To može predstavljati znatno ograničenje obzirom da se korištenje uređaja limitira, te nije dostupno svakome. U konkretnom slučaju to ne predstavlja veliki problem, obzirom da je cilj korištenja i razvijanja jednog ovakvog uređaja bio pomoći osobama sa problemima u radu kognitivnih sposobnosti, a koje nerijetko imaju pomoć drugih osoba.

Na određivanje, odnosno, estimaciju pozicije zjenice oka utječe mnogo faktora, što je čini nepouzdanom metodom. Ukoliko bi bilo moguće ekstraktovati uglove oka, bez potrebe za ljudskom intervencijom, dobila bi se metoda u kojoj je teže na polju maske napraviti fatalne greške koje mogu utjecati na rad programa. Nadalje, problem koji najviše utječe na estimaciju pozicije zjenice predstavlja osvjetljenje. Problem osvjetljenja se može riješiti uvođenjem adaptivnih metoda koje će mijenjati čak i parametre kamere tokom snimanja, kako bi se dobila slika iz koje je veoma dobro moguće vidjeti oko sa zjenicom. Teško je reći da li bi ikada ovakav pristup mogao raditi bez uvođenja eksternog osvjetljavanja lica ili oka, koje bi se mijenjalo tokom rada programa, ali je jedna od mogućnosti rješavanja problema osvjetljenja.

Za najbolje rezultate ovakve metode je potrebno koristiti algoritam koji će ekstraktovati samo beonjaču sa zjenicom. Takav algoritam treba na osovu boje (obzirom da je beonjača bijela u odnosu na ostale dijelove oka) izdvojiti ROI, te omogućiti dalju obradu samo tog dijela oka. Ovakva metoda je mnogo naprednija i programu vremenski zahtjevnija za izvršavanje, ali bi omogućila odstranjivanje jednog od faktora koji utječu na estimaciju pozicije zjenice.

Procjena usmjerenosti pogleda ovdje nije vršena korištenjem matrice homogene transformacije, već empirijskom metodom određivanja pragova koji se testiraju i na osnovu kojih se procjenjuje usmjerenost pogleda. Korištenje matrice homogene transformacije nije korišteno zbog povećanja vremena izvršenja programa, te zbog činjenice da pozicija zjenice nije uvijek u potpunosti tačno određena, što dovodi do toga da estimirane koordinate zjenice budu vrijednosti takve da bi se pomoću njih teško mogla pronaći odgovarajuća i tačna matrica homogene transformacije. Ukoliko bi algoritam određivanja pozicije zjenice bio poboljšan do zadovolja-

vajuće tačnosti, matrica homogene transformacije za određivanje usmjerenosti pogleda, te za autorekalibraciju, postaje veoma dobro moguće rješenje.

Korištenje razvijenog uređaja je znatno ograničeno na ambijentalne uslove, stoga nije najbolje rješenje za korištenje irisa oka kao virtuelnog pokazivača. Pomenuta rješenja problema bi mogla značajno utjecati na estimaciju zjenice, i poboljšati rad programa. Ne može se reći da algoritam koji se zasniva na estimaciji pozicije zjenice koristeći sumu piksela po redovima i kolonama slike, nije dobar algoritam, ali primijenjen u radu ovog uređaja nije dao zadovoljavajuće rezultate i savjetuje se korištenje drugih metoda estimacije zjenice oka. Također je potrebno naglasiti da bi ovakav pristup mogao biti značajno poboljšan korištenjem metode mašinskog učenja, obzirom da bi tada bilo moguće neke od parametara prilagoditi trenutnom korisniku i ambijentalnim uslovima.

Ostvareni ciljevi završnog rada

Izvšreni su eksperimenti koji pokazuju tačnost estimacije pozicije zjenice i korištenih metoda, te ovisnosti od ambijentalnih uslova. Rezultati izvršenih eksperimenata su prikazani u svakom od poglavlja, a generalno su svi eksperimenti vezani za softverski dio programa izvršavani u sličnim ambijentalnim uslovima, kako bi se dobili korisni podaci za obradu.

Uređaj koristi neinvazivnu metodu, što je bio najbitniji uslov prilokom kreiranja uređaja, obzirom da su korištene kamare koje snimaju oko i nemaju nikakvog kontakta s okom, te je kompletna metoda estimacije pozicije zjenice zasnovana na obradi slike.

Uređaj je struktuiran tako da ga je moguće koristiti od strane bilo kojeg korisnika, bilo kojih propozicija glave, dodavanjem prilagodljivog remena za stezanje uređaja i korištenjem ogledala i kamera na fleksibilnim postoljima.

Eksperimenti su pokazali da preciznost i ponovljivost metode praćenja pogleda nije u potpunosti ostvarena, te izvršavanje estimacije ovisi od ambijentalnih uslova. Također je omogućeno pokretanje glave u određenoj mjeri, koja može biti dovoljna za korištenje u primjeni koja je planirana.

Uređaj ne koristi invazivnu metodu niti ima dijelove koji su u opasni za korištenje od strane korisnika bilo koje dobi. Moguće je izvršiti značajna poboljšanja udobnosti uređaja korištenjem materijala koji ne stvaraju nelagodu kod korisnika, ali je takav pristup na prototipu uređaja prilično ograničen.

Uređaj ne ograničava osobu na potpuno mirovanje ili određeni položaj tijela.

Obrađena literatura je većinom bazirana na korištenju invazivnih metoda ili metoda koje imaju mnogo više ograničenja od metode korištene u ovom radu. Uspjelo se pobjeći od mnogih nametnutih ograničenja, ali je mnogo njih i ostalo.

Kao zaključak, razvijena metoda ima potencijala za unapređivanje, te je vrlo vjerovatno da bi radila skoro savršeno uz izmjene nekih od pristupa korištenih u ovom radu, a navedenim u zaključku.

Prilozi

Prilog A

Korištene OpenCV funkcija

A.1 Kaskadni klasifikator

```
void CascadeClassifier::detectMultiScale(const Mat& image, vector<Rect>& objects,
double scaleFactor = 1.1, int minNeighbors=3, int flags=0, Size min_size=Size(),
Size maxSize=Size())
```

Parametri:

- **cascade** - Haar klasifikatorska kaskada. Može biti učitana iz XML ili YAML fajla koristeći Load(). Kad kaskada nije više potrebna, oslobađa se koristeći *cvReleaseHaarClassifiesCascade(&cascade)*.
- **image** - Matrica tipa CV_8U koja sadrži sliku gdje objekti trebaju biti detektovani.
- **objects** - Vektor pravougaonika gdje svaki pravougaonik sadrži detektovani objekat.
- **scaleFactor** - Parametar koji specificira koliko je veličina slike smanjena na svakoj skali slike.
- **minNeighbors** - Parametar koji specificira koliko susjeda svaki kandidat za pravougaonik treba imati da bi bio zadržan.
- **flags** - Parametar sa istim značenjem za staru kaskadu kao u funkciji *cvHaarDetectObjects*. Nije korišten za novu kaskadu.
- **minSize** - Minimalna moguća veličina objekta. Objekti manji od toga su ignorisani.
- **maxSize** - Maksimalna moguća veličina objekta. Objekti veći od toga su ignorisani.

Za pronalazak oka na slici prvo je potrebno definisati globalnu varijablu *eyeCascade* tipa klase *cv::CascadeClassifier*. U pomenutu varijablu je korištenjem *cv::CascadeClassifier::load* metode učitan potrebni XML Haar klasifikatorski fajl. Haar cascade (bos. *Haar kaskada*) pod nazivom ***haarcascade_eye_tree_eyeglasses.xml*** je kaskada dizajnirana od strane Shameem Hameed-a ([9]) za detekciju očiju, sa boljim rezultatima i u slučaju kada su korištene naočale. Nakon toga je detekcija oka izvršena korištenjem *cv::CascadeClassifier::detectMultiScale* metode, koja kao rezultat vraća pravougaone okvire za detektovane oči. XML fajl korišten u programu omogućava pronalazak oba oka, međutim obzirom da je za potrebe zadatka potrebno

samo jedno oko, vratit će samo jedan pravougaonik.

Linija koda koja izvršava detekciju oka:

```
eyeCascade.detectMultiScale(frame, eyes, 1.1, 2, 0 | CV_HAAR_SCALE_IMAGE,
cv::Size(150, 150));
```

Ukoliko je oko detektovano oko istog se iscrtava pravougaonik koji ozačava područuje oka na osnovu pravougaonih okvira koje vraća pozivanje funkcije *detectMultiScale*.

A.2 Jednostavno određivanje praga

```
cv::threshold(src, dst, thresh, maxval, type)
```

Parametri

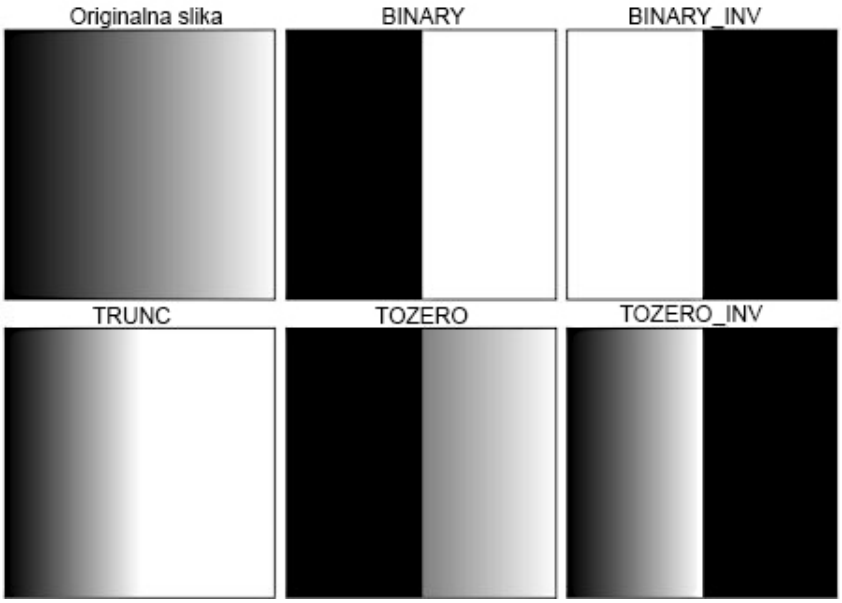
- **src** - ulazni niz
- **dst** - izlazni niz iste veličine i tipa sa istim brojem kanala kao i src
- **thresh** - granična vrijednost (*engl. threshold value*
- **maxval** - maksimalna vrijednost koja će biti korištena sa **cv.THRESH_BINARY** i **cv.THRESH_BINARY_INV** - thresholding tipovima (*engl. thresholding types*)
- **type** - thresholding tip

thresholding type - OpenCV pruža različite stilove thresholdinga, a četvrti parametar funkcije služi za odlučivanje koji će biti korišten. Različiti tipovi su:

- THRESH_BINARY
- THRESH_BINARY_INV
- THRESH_TRUNC
- THRESH_TOZERO
- THRESH_OTSU
- THRESH_TRIANGLE

Napomena: U slučaju **THRESH_OTSU** ili **THRESH_TRIANGLE** tipova, ulazna slika treba biti jednokanalna.

Primjer pozivanja svakog od navedenih tipova thresholda je prikazan na slici A.1.



Slika A.1: Primjer primjene simple thresholdinga sa različitim tipovima [3]

A.3 Adaptivno određivanje praga

Adaptivni thresholding ima tri 'specijalna' ulazna parametra i samo jedan izlazni argument.

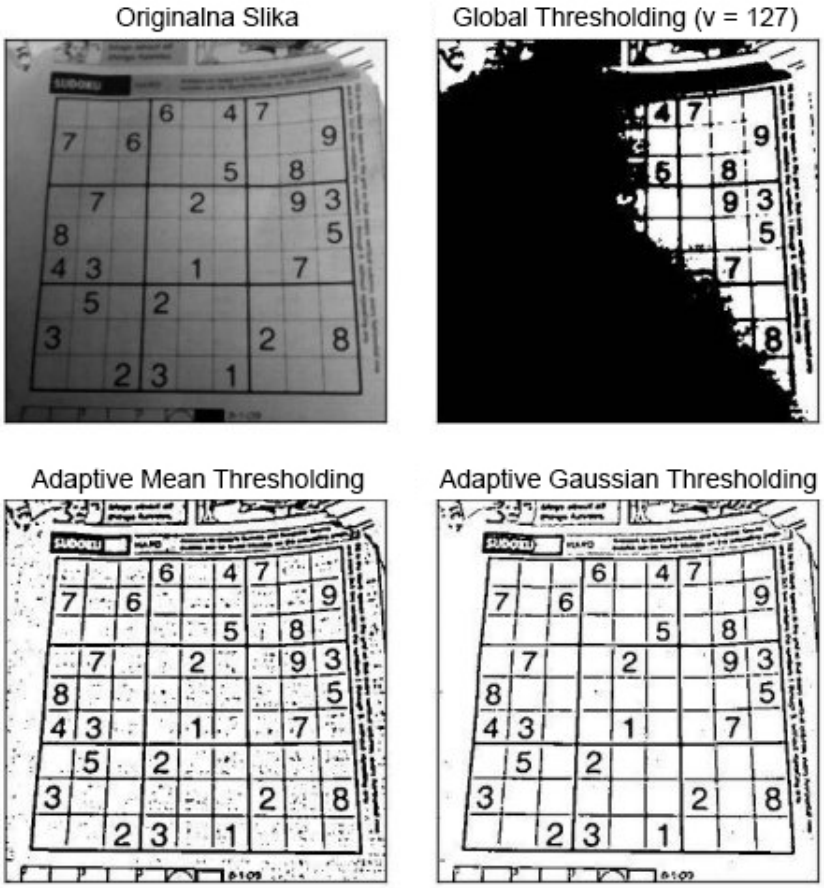
Adaptive Method (*bos. adaptivni metod*) - odlučuje kako se računa threshold vrijednost.

- ADAPTIVE_THRESH_MEAN_C - threshold vrijednost je srednja vrijednost susjednog područja.
- ADAPTIVE_THRESH_GAUSSIAN_C - threshold vrijednost je težinska suma susjednih vrijednosti gdje su težine Gaussova prozorska funkcija.

Block Size (*bos. veličina bloka*) - određuje veličinu susjednog područja.

C - konstanta koja se oduzima od srednje vrijednosti ili od težinske srednje vrijednosti.

Na slici A.2 prikazan je primjer primjene globalnog, odnosno, jednostavnog thresholdinga, adaptivnog thresholdinga sa srednjom vrijednošću i adaptivnog thresholdinga koji koristi Gaussove težinske koeficijente.



Slika A.2: Primjer primjene adaptivnog thresholdinga sa različitim tipovima [3]

A.4 Hough Circle Transform (*bos. Hough transformacija kruga*)

Potrebna su tri parametra za definisanje kruga:

$$C : (x_{center}, y_{center}, r)$$

gdje je (x_{center}, y_{center}) tačka koja definiše poziciju centra, a r je radijus, što omogućava da krug bude u potpunosti definisan, kao što je prikazano na slici A.3.

Slika A.3: Primjer Hough circle transform za detekciju kruga [3]

Funkcija `HoughCircles( src_grey, circles, CV_HOUGH_GRADIENT, dp, min_dist, param_1, param_2, min_radius, max_raidus)` ima 5 parametara:

- **src_gray** - ulazna slika u grayscale-u
- **circles** - vektor u koji je smješten set od 3 vrijednosti: x_c, y_c, r za svaku detekciju kruga
- **CV_HOUGH_GRADIENT** - definiše metodu detekcije, trenutno je ovo jedina dostupna metoda u OpenCV-u
- **dp** - inverzni odnos rezolucije, npr. ako je dp = 1, akumulator ima istu rezoluciju kao i ulazna slika, dok ako je dp = 2, akumulator ima upola veliku širinu i visinu
- **min_dist** - minimalna udaljenost između detektovanih centara
- **param_1** - gornja granica za interni Canny detektor
- **param_2** - granica za detekciju centra, što je manja granica, veća je vjerovatnoća da će doći do lažne detekcije
- **min_radius** - minimalni radijus za detekciju, vrijednost se postavlja na nulu ako je minimalni radijus nepoznat
- **max_radius** - maksimalni radijus za detekciju, vrijednost se postavlja na nulu ako je maksimalni radijus nepoznat

Pomenut je Canny detektor, za koji je bitno pomenuti da je detektor ivica. Postavlja se gornja granica za Canny detektor, što nadalje znači da ukoliko je gradijent piksela veći od gornje granice, piksel se smatra ivicom.

A.5 Gaussov filter

Za izvođenje smoothing operacije, aplicira se filter na sliku. Najčešće forme filtera su linearne, kod kojih je vrijednost izlaznog piksela (npr. $g(i,j)$), određena težinskom sumom ulaznih vrijednosti piksela (npr. $f(i+k, j+l)$).

$$g(i,j) = \sum_{(k,l)} f(i+k, j+l)h(k,l)$$

gdje je $h(k,l)$ takozvano *jezgro* (engl. *kernel*), što je ništa drugo do koeficijenti filtera. Pomaže ukoliko se filter vizualizira kao prozor sa koeficijentima koji klizi preko slike.

Gaussov filter je vjerovatno najkorisniji filter (iako ne najbrži). Gaussovo filtriranje je izvođeno konvoluiranje svake tačke u ulaznom nizu sa Gaussovim kernelom i sumiranjem svih kako bi se dobio izlazni niz.

Gaussov filter se definiše kao:

```
void GaussianBlur(InputArray src, OutputArray dst, Size ksize,
double sigmaX, double sigmaY=0, int borderType=BORDER_DEFAULT )
```

Gdje je:

- **src** - ulazna slika koja može imati bilo koji broj kanala, koji su procesirani neovisno, s tim da dubina treba biti CV_8U, CV_16U, CV_CV16S, CV_32F ili CV_64F
- **dst** - izlazna slika iste veličine i tipa kao i src
- **ksize** - veličina Gaussovog kernela, s tim da širina i dužina od ksize mogu biti različite, s tim da obje moraju biti pozitivne i neparne
- **sigmaX** - Gaussova kernelova standardna devijacija u X smjeru
- **sigmaY** - Gaussova kernelova devijacija u Y smjeru; ako je sigmaY nula, tada će biti jednaka sigmaX, ako su obje sigme nula, tada se računaju iz ksize.width i ksize.height, respektivno
- **borderType** - ekstrapolacijska metoda piksela

A.6 Dokumentacija i kod

A.7 Karakteristike kamere uEye UI - 1008 XS

Korištena kamera uEye UI - 1008 XS ([10]) ima sljedeće bitne tehničke karakteristike:

- USB 2.0
- 8 Megapixel CMOS senzor
- 3280 x 2464 u modu slikanja
- 1280 x 720 sa 15 slika u sekundi
- integrirana leća sa autofokusom
- elektronska stabilizacija slike
- automatska kontrola slike

A.8 C++ kod

C++ kod za programiranje uređaja je moguće vidjeti na Korištenje irisa oka kao virtuelnog pokazivača ili posjetom stranice:
`https://github.com/AmilaSab/Koristenje-irisa-oka-kao-virtuelnog-pokazivaca`.

Demonstracija rada uređaja je data na YouTube link demonstracije rada uređaja ili posjetom stranice: `https://www.youtube.com/watch?v=JJU3a2RcA2Q&feature=youtu.be`.

Literatura

[1] Chaudhry, M. S. M. N. G. M. U. G. S., “Gazepointer: A real-time mouse pointer control implementation based on eye gaze tracking”, 2012.

[2] Cáceres, E., Carrasco, M., Ríos, S., “Evaluation of an eye-pointer interaction device for human-computer interaction”, Heliyon, Vol. 4, No. 3, 2018, str. e00574.

[3] “Opencv dokumentacija”, dostupno na: https://docs.opencv.org/3.4.0/index.html

[4] Drewes, H., “Eye gaze tracking for human computer interaction”, Doktorski rad, lmu, 2010.

[5] Poole, A., Ball, L., Eye tracking in human-computer interaction and usability research: Current status and future prospects, 01 2006, str. 211-219.

[6] Duchowski, A. T., “Eye tracking methodology”, Theory and practice, Vol. 328, 2007, str. 614.

[7] Gizdić, M., “Detekcija usmjerenosti pogleda i upravljanje računalom primjenom računalnog vida”, 2015.

[8] Kaurić, I., “Sustav za detekciju usmjerenosti pogleda”, 2013.

[9] Hameed, S., “Detektor oka”, dostupno na: https://docs.opencv.org/3.4.0/index.html

[10] “Dokumentacija kamere”, dostupno na: https://www.m-service.de/seiten/gb/downloads_gb_only/gb_CV-UI-1008-XS-C.pdf