TALLINNA TEHNIKAÜLIKOOL

Martin Lensment (IAEM124481)

SILMAGA JUHITAV SÖÖMIST ABISTAV ROBOT PUUETEGA INIMESTELE (TARKVARA)

Magistritöö

ELEKTROONIKA JA BIOONIKA ÕPPEKAVA

Juhendaja:
Rauno Gordon
PhD, vanemteadur

Tallinn 2015

Autorideklaratsioon

Deklareerin, et käesolev magistritöö, mis on minu iseseisva töö tulemus, on esitatud Tallinna Tehnikaülikooli magistrikraadi taotlemiseks ja et selle alusel ei ole varem taotletud akadeemilist kraadi. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud või (avaldamata tööde korral) toodud autorlus välja põhitekstis.

.....

(töö autori allkiri ja kuupäev)

Sisukord

Mõisted ja lühendid

ARM - *Advanced RISC Machines* - vähendatud käsustikuga arvutiarhitektuur

CPU - *Central Processing Unit,* kesktöötlusseade - juhib arvuti tööd

CSI - *Camera Serial Interface,* kaamera järjestikliides - liides, mille kaudu ühenduvad protsessor ja kaamera

DARPA - *Defense Advanced Research Projects Agency,* Ameerika Ühendriikide Kaitseministeeriumi poolt loodud agentuur, mis on loodud uute (sõja)tehnoloogiate arendamiseks ja uurimiseks

DEKA - ***De****an* ***Ka****men,* Dean Kameni poolt loodud firma, mis arendab innovatiivseid tehnoloogiaid

DMA - *Direct Memory Access,* otsemällupöörduskanal - arvutisüsteemi võimalus pääseda ligi süsteemimälule ilma CPU sekkumiseta

FDA - *U.S. Food and Drug Administration,* Ameerika Ühendriikide föderaalne agentuur, mis reguleerib toidu ja ravimite kasutust riigis

GIL - *Global Interpreter Lock,* Globaalne Interpretaatori Lukk - Lukk, mis ei luba lõimedel protsessoril paralleelselt töötada

GPIO - *General Purpose Input / Output,* geneeriline sisend / väljund viik protsessori plaadil

HD - *High Definition*, kõrglahutus

I/O - *Input Output,* Sisend / Väljund

LED - *Light Emitting Diode,* valgusdiood

NOIR - *No Infra-Red,* Ilma infrapunata

PWM - *Pulse-Width Modulation,* pulsilaiusmodulatsioon - modulatsiooni liik, milles väljundpinge reguleerimiseks muudetakse impulsside laiust

RISC - *Reduced Instruction Set Computer,* kärbitud käsustikuga arvuti

RPI - *Raspberry Pi* mikroarvuti

SD card - *Secure Digital card,* SD-kaart - Välkmälukaart andmete turvaliseks talletamiseks

SSH - *Secure Socket Shell,* turvakest - võrguprotokoll, mis võimaldab andmeid saata üle turvalise kanali

USB - *Universal Serial Bus,* universaalne järjestiksiin - välissiini standard

VNC - *Virtual Network Computing,* graafiline töölaua jagamise süsteem

WIFI - *Wireless Fidelity,* juhtmevaba võrk

1. Sissejuhatus

Proteeside ajalugu sai alguse juba Vana Egiptuses. Egiptlaste eesmärk oli amputeeritud inimestele anda tagasi terve olemise tunne. Esimene funktsionaalne protees, mis on siiani leitud, pärineb 3000 aastat vanalt Egiptuse muumialt ja selleks oli nahast ja puust tehtud varvas.[1] Manchesteri ülikooli professorid leidsid, et see tehislik kehaosa ei kandnud mitte ainult esteetilise välimuse andmise ülesannet, vaid oli ka täiesti funktsionaalne ja parandas tunduvalt patsiendi mugavustunnet. Teadlased avastasid, et protees oli optimeeritud just sandaalidega kõndimiseks.

Kuni keskajani kandsid proteesid rohkem siiski kosmeetilist eesmärki. Alles 1500 aastat pärast Kristust hakkasid tekkima esimesed korralikumad proteesid, millel oli ka funktsionaalne eesmärk. Ambroise Paré leiutas esimese mehaanilise käe ja umbes sada aastat hiljem, aastal 1696, lõi Pieter Verduyn esimese põlvest liikuva jala. 1800 leiutas Londonis elav James Potts juba liikuva jalalabaga proteesi.[2]

Seoses elektroonika arenemisega tekkis varsti võimalus väga täpselt imiteerida bioloogilisi organeid või organisme. Aastal 1958 võttis Ameerika doktor Jack E. Steele kasutusele termini "bioonika."[3] Ta kombineeris kaks sõna bio (elu) ja oonika (elektroonika) ning tulemusena sündis sõna bioonika. Teadusharu on vähem kui saja aasta jooksul arenenud niivõrd kiiresti, et tänapäeval konstrueeritakse juba mõtte abil kontrollitavaid antropomorfseid robotkäsi. Üks taoline projekt on DEKA Research & Development

[1] http://science.howstuffworks.com/prosthetic-limb1.htm (30.04.2015)
[2] http://www.amputee-coalition.org/inmotion/nov_dec_07/history_prosthetics.html (30.04.2015)
[3] http://www.healthguideinfo.com/prosthetics-bionics/p9070/ (30.04.2015)

Corporation poolt arendatud ja DARPA kaudu rahastatud käe proteesi projekt "Luke Arm."[4] Käsi saab signaale patsiendi ajju viidud elektroodide abil ja proteesi on võimalik liigutada kümnes mõõtmes samaaegselt.[5] Loomulikult on selline tehnoloogia väga kallis ning seda on arendanud kümned teadlased juba rohkem kui üheksa aastat. Üle 100 miljoni dollari maksnud projekt sai lõpuks 2014. aasta keskel FDA heakskiidu,[6] et tehnoloogia on piisavalt täiuslik ja seda saab hakata kommertslikult müüma.

Tavainimese jaoks jääb kahjuks selline protees ilmselt kättesaamatuks veel pikkadeks aastateks eelkõige tema hinna ja asjaolu tõttu, et ajju peab paigaldama elektroodid, mis ajutegevust monitoorivad.

Christopher & Dana Reeve Foundation'i korraldatud uuringu[7] andmetel elab maailmas ligikaudu kuus miljonit osaliselt või täielikult halvatud inimest, kelle igapäeva toimetused on drastiliselt raskendatud. Lisaks halvatud inimestele on ka veel inimesi, kellel on mõni jäse amputeeritud või ei tööta normaalselt. Nõudlus bioonilise kehaosade järele on suur ja kasvab iga päevaga, kuna inimesi on maailmas järjest rohkem.

[4] http://www.engadget.com/2014/12/17/darpa-mind-control-robot-arm/ (30.04.2015)

[5] http://iopscience.iop.org/1741-2552/12/1/016011/article (30.04.2015)

[6] http://spectrum.ieee.org/automaton/biomedical/bionics/dean-kamen-luke-arm-prosthesis-receives-fda-approval (30.04.2015)

[7] http://www.christopherreeve.org/site/c.ddJFKRNoFiG/b.5900347/k.3B2/PARALYSIS_SURVEY_ALMOST_6_Million_affected.htm (30.04.2015)

2. Eesmärk

Käesoleva töö eesmärk on luua silmadega kontrollitav robotkäsi. Kuna sellise jäseme otstarve võib olla väga lai, siis tehakse kitsendus ja keskendutakse ainult ühele kasutusjuhule, milleks on söömine. Käsi on suunatud just eelkõige puudega inimestele, kes ei suuda iseseisvalt toitu manustada. Robot peaks aitama patsiendi hooldajal töökoormust vähendada. Kuna töömaht on väga suur, siis avaldatakse projektist kaks magistritööd. Käesolev töö kirjeldab detailselt roboti tarkvara, teine töö keskendub elektroonikale ja mehaanikale.[8] Projektile antakse koodnimi R(eye)bot, lühidalt Rebot.

Sissejuhatuses kirjeldatud Luke Arm protees on mõeldud erinevate tegevuste sooritamise jaoks, kuid peale selle on turul ka ka juba valmis karbitooteid ja mittekommertslikke robotkäsi, mis on juba keskendunud ainult toidu manustamisele. Kõik leitud lahendused on siiski kallid ning võivad jääda tavainimestele kättesaamatuteks.

Töö käigus:

- tehakse ülevaade sellest, kuidas juba olemas olevad seadmed töötavad ning võetakse nendest projektidest ideid;
- valitakse sihtgrupp, kellele lahendus on suunatud;
- selgitatakse välja, millised on eetilised aspektid;
- mõeldakse välja roboti tööpõhimõte;
- valitakse välja komponendid;

[8] Katrin Kibbal (2015) https://github.com/mlensment/rebot/raw/master/katrin_msc.pdf

- ehitatakse valmis reaalne roboti prototüüp;
- kirjutatakse valmis robotile mõeldud tarkvara;

Töö raamidesse jääb ainult tehnoloogia ja prototüübi väljaarendamine. Karbitoote loomine projekti ei mahu, kuna see eeldab sügavaid teadmisi materjalidest, tootmisliinidest ja nende projekteerimisest. See aga ei kuulu projektiga töötavate inseneride kompetentsi ning selleks ei ole eraldatud ka piisavaid rahalisi vahendeid.

3. Ülevaade sarnastest projektidest

3.1. Kommertslikud seadmed

3.1.1. Meal Buddy

Seoses tehnoloogiliste vidinate nõudluse pidevale kasvule on elektroonika tootmine järk järgult maailmas suurenenud. 2014. aasta septembris oli näiteks iPhone 6 tootmismaht 540000 telefoni päevas.[9] Suured tootmismahud, konkurents ning tehnoloogia kiire vananemine surub vanemate seadmete hindu alla ja see tingib selle, et peaaegu igal inimesel võimalik soetada mõistliku hinnaga erinevaid elektroonikakomponente.

Sellest võiks ju järeldada, et puuetega inimeste abistamiseks mõeldud seadmed odavnevad samuti, kuid interneti abil tehtud taustauuring selgitas välja, et näiteks halvatud inimeste toitmiseks mõeldud seade Meal Buddy[10] maksab 2015. aasta kevadel 3535 USA dollarit (umbes 3156 eurot) ning koos erinevate lisadega küündib hind lausa ligikaudu 4500 USA dollarini (umbes 4018 eurot) (USD - EUR kurss seisuga 30.04.2015). See on 5.4 Eesti mediaanpalka[11] ja tundub äärmiselt ebatõenäoline, et keskmine eestlane saaks seadet soetada ilma pikka aega raha kogumata.

Meal Buddy-l on täisvarustuses järgmised võimalused ning komponendid:

- ühe nupuvajutusega aktiveeritav eelprogrammeeritud lusika liigutamise funktsioon;

[9] http://9to5mac.com/2014/09/17/iphone-6-production-by-the-numbers-100-production-lines-200k-workers-540k-phones-a-day/ (30.04.2015)
[10] http://www.pattersonmedical.com/app.aspx?cmd=getProduct&key=IF_46883 (30.04.2015)
[11] http://uudised.err.ee/v/majandus/1a12afc7-590c-4345-a986-f3ef6a84b854 (30.04.2015)

- eelprogrammeeritud robotkäsi;
- kohandatavad toitmisrütmid (järjestikune, suvaline ja kasutaja valitud);
- programmeeritav suu asukoht;
- kolm kaussi (kolmekäigulise lõuna jaoks);
- üheleheline kasutusjuhend;
- konfigureeritav toidu hulk lusikal;
- konfigureeritav robotkäe kiirus;
- pühkimise programm, et vältida tilkumist;
- esteetiline viimistlus;
- aku ja laadija;

3.1.2. My Spoon

Umbes samas hinnaklassis on Meal Buddy konkurent My Spoon,[12] mille on välja arendanud Jaapani firma Secom.

Nimetatud autonoomse abivahendi võimalused ja komponendid on järgmised:

- nupuvajutusega aktiveeritav täielikult automaatne toitmise funktsioon;
- juhtkangi abil aktiveeritav poolautomaatne toitmise funktsioon (inimene valib sobiva kausi ning robot toidab valitud kausist automaatselt patsienti);
- juhtkangi abil teostatav manuaalne toitmise funktsioon (patsient kontrollib juhtkangi abil täielikult robotkätt);
- neli kaussi;
- esteetiline viimistlus;

My Spoon seadet ei saa kasutada inimesed, kellel on raskusi närimise või neelamisega, kes ei saa pead liigutada, kes ei saa püsti istuda (keha peab olema üle 60 kraadise nurga all), pimedad, sügava vaimse puudega inimesed ja kes ei saa aru kuidas seade töötab.

[12] http://www.secom.co.jp/english/myspoon/index.html (30.04.2015)

Robot saab hakkama nii tahkete toitude (näitkeks riis) kui ka pehmete toitude tõstmisega (erinevad pudrud). Samuti saab selle abil manustada jooki.

Robotil on olemas juhtkang, mille abil saab patsient valida toitu ja seda isegi juhtkangi abil üles võtta, kui ta ei soovi kasutada automaatset režiimi.

3.2. Mittekommertslikud seadmed

3.2.1. ICraft

Suuremat tähelepanu väärib Northeastern University välja arendatud projekt ICraft,[13] kuna erinevalt Meal Buddy-st ja My Spoon-ist kontrollitakse seda robotit silmadega. Kaheksa kolleegi sellest ülikoolist on juba enne käesoleva töö kirjutamist leidnud, et kommertslikult pakutavad tooted on liiga kallid ning püüdnud ICraft projekti raames hinda oluliselt minimeerida.

ICrafti funktsionaalsus on eeltoodud projektidega sarnane, suurim erinevus seisneb robotkäe kontrollimise põhimõttes. Kui Meal Buddy ja My Spoon puhul toimus seadme juhtimine nupu või juhtkangi abil, siis ICrafti kontrollitakse ainult silmadega. Selline lähenemine ei tähenda küll automaatselt tunduvalt suuremat auditooriumit võrreldes väljatoodud kommertstoodetega, kuid kindlasti tagab platvormi rohkemate ja keerukamate liigutuste väljaarendamiseks. Näiteks on ICrafti kasutajaliidese abil võimalik patsiendil valida endale meelepärane kauss, millest süüa. Sarnane optiline lahendus on eesmärgiks ka käesolevas töös loodavale projektile.

Kuigi ICrafti arendajad väidavad oma projekti kohta avaldatud dokumendis, et nende põhiline eesmärk on projekti kulude minimeerimine, läksid nad siiski kergema vastupanu teed ja investeerisid juba valmis robotkäppa. Ainuüksi juba robotkäe hind ületab selle töö raames loodava projekti eeldatavat kulu.
Järgnevalt on toodud ICraft'i ehitamiseks vajalikud komponendid ja nende hinnad:

[13] http://www.ece.neu.edu/personal/meleis/icraft.pdf (30.04.2015)

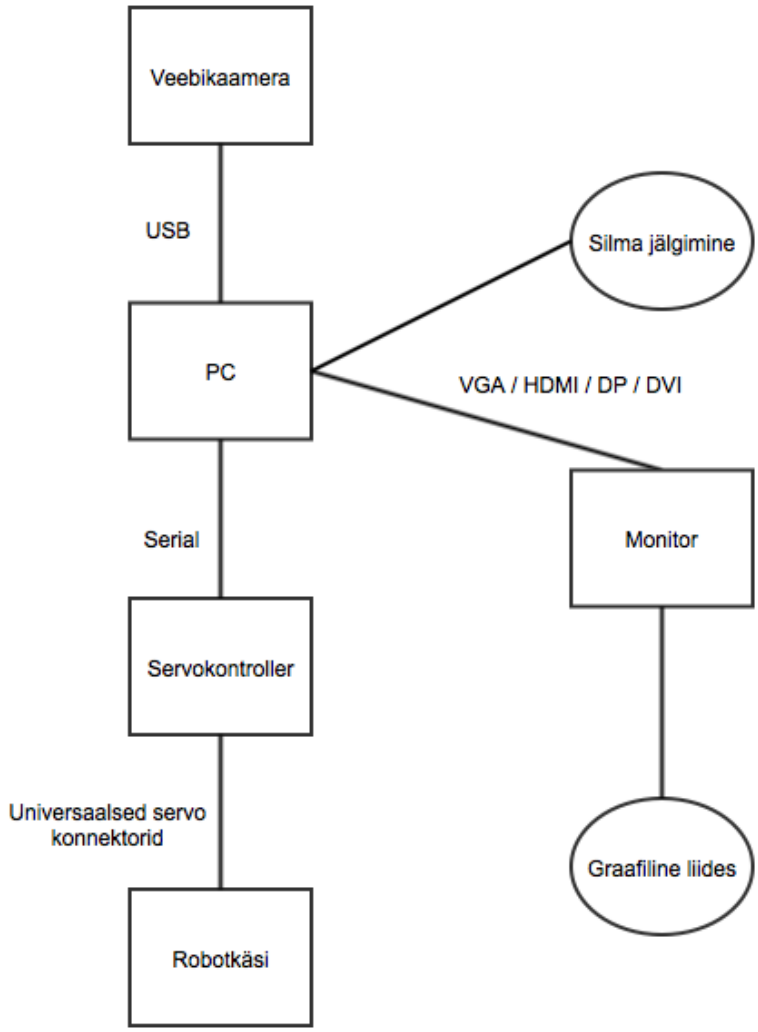
Osa	Hind
RobotShop M100RAK Robotkäpp	$598.85 (534.90€)[14]
Lynxmotion SSC-32 Servo kontroller Ei toodeta enam, asendatud SSC-32U kontrolleriga	$39.95 (35.68€)[15]
Lynxmotion SSC-32U Servo kontroller	$44.95 (40.07€)[16]
Kauss "Scooper Bowl"	$7.30 (6.51€)[17]
Pudel "The Hydrant"	$32.19 (28.7€)[18]
IR filtrita Creative kaamera ja infrapuna allikas	$114.74 (101.94€)[19]
Vooluallikas	$11.99 (10.71€)[20]
Muu	$120.14 (106.71€)[21]
Kokku	$930.16 (829.54€)

USD - EUR kurss seisuga 30.04.2015

Kuigi ICrafti dokumendis on kirjas ka projekti kulud, võetakse siiski parema võrdluse ja ettekujutuse saamiseks aluseks hinnad, mis kehtivad töö kirjutamise ajal poodides.

Lisaks robootika poolele vajab ICraft ka arvutit, kus töötab programmikood. Graafilise liidese kuvamiseks läheb vaja ka eraldiseisvat monitori, seega ainult sülearvutist ei piisa. Suurel ekraanil on pilgu asukoha tuvastamine lihtsam. Lisaks töötab nende kasutatud silma jälgimise tarkvara ainult Microsoft Windowsi platvormil, mis on tasuline. Seega lisandub palju varjatud kulusid, millega koos on kogu süsteemi hind palju kallim ja võib ulatuda tuhandetesse eurodesse. Projekti tegijad küll väidavad, et paljudel inimestel on juba Windowsiga arvuti olemas. See võib olla küll tõsi tervete inimeste puhul, aga kui sihtgrupiks on halvatud inimesed, siis võib selles väites kahelda. ICrafti komponentide ühendamise skeem on näidatud joonisel 1.

[14] http://www.robotshop.com/en/robotshop-m100rak-v2-modular-robotic-arm-kit-no-electronics.html (30.04.2015)
[15] http://www.robotshop.com/en/lynxmotion-ssc-32-servo-controller.html (30.04.2015)
[16] http://www.robotshop.com/en/lynxmotion-ssc-32u-usb-servo-controller.html (30.04.2015)
[17] http://www.amazon.com/Ableware-745340000-Scooper-Bowl-Suction/dp/B00LSOM8WU (30.04.2015)
[18] http://www.maddak.com/the-hydrant-p-27999.html (30.04.2015)
[19] http://www.ece.neu.edu/personal/meleis/icraft.pdf (30.04.2015)
[20] http://www.radioshack.com/12-6v-ct-3-0a-chassis-mount-transformer-with-leads/2731511.html (30.04.2015)
[21] http://www.ece.neu.edu/personal/meleis/icraft.pdf (30.04.2015)



Joonis 1 - ICraft arhitektuur

Nagu jooniselt näha, siis keskne komponent kogu süsteemil on arvuti. Selle külge on ühendatud infrapuna filtrita USB veebikaamera, monitor ja servokontroller. Viimane on omakorda ühendatud robotkäega, mille ülesanne on kausist toitu patsiendi suhu toimetada.

Arvutisse on installeeritud spetsiaalselt silma asukoha jälgimiseks välja töötatud vabavaraline tarkvara ITU Gaze Tracker.[22]

Silma jälgimise lihtsustamiseks on neil kasutusel ilma infrapuna filtrita veebikaamera. Lauale on monteeritud infrapunases diapasoonis kiirgav valgusallikas. Järgnevalt kirjeldatavat

[22] http://sourceforge.net/projects/gazetrackinglib/ (01.05.2015)

tehnikat nimetatakse "tumeda pupilli tehnikaks."[23] Silmalt kaamerasse tagasi peegeldunud aktiivne valgus moodustab pildile heleda täpi, mida saab pildituvastusalgoritmide abil tuvastada. Peegeldunud valguse ja pupilli keskpunkti vahelise vektori järgi saab hinnata, kuhu patsient parasjagu vaatab. Enne kasutamist tuleb läbi viia individuaalne kalibreerimistsükkel. Teine sarnane tehnika on "heleda pupilli tehnika." Nende kahe vahe seisneb aktiivse valguse allika paigutamises. Kui tumeda pupilli korral paigutati infrapuna valgusallikas kaamera optilises teljest eemale, siis heleda pupilli tehnikat kasutades on see vastupidi. Heleda pupilli puhul näib pupill pildi peal võrreldes iirisega hele, kuna valgus peegeldub silma võrkkestalt otse tagasi kaamerasse. Sama fenomen esineb ka välguga pildistatud fotodel,[24] mis on tehtud nõrgas valguses. Kompakt-kaamerate välk ja objektiiv on lähestikku, seega fotoaparaat jäädvustab silma võrkkestalt tagasi peegeldunud valguse ning silma pupill näib punane.

Mõlemad tehnikad töötavad hästi ning tulemused olenevad silma eripärast, valguses ja ka patsiendi päritolust. Silmatuvastus on väga lai ja keeruline ala, sellest on avaldatud palju uurimusöid ja seda on uurinud sajad ja tuhanded teadlased aastakümneid. Võimalusi pupilli jälgimiseks on palju ning ei ole olemas 100% õiget lahendust. Sageli seab piirangud ka kasutatav riistvara või peab arvestama ekstreemsete valgustingimustega. Parim lahendus leitakse tavaliselt projekti arenduse käigus.

ICrafti graafilise liidesena ekraanile on ekraanil kuvatud neli kastikest. Kolm nendest monitori ülaosas, üks suur all. Seadme kontrollimine käib kastikeste vaatamise abil. Kasutaja valib vastavat kastikest silmitsedes endale kausi, millest robotkäpp teda toitma hakkab. Et funktsiooni valida, tuleb kasutajal vaadata valitud kasti kaks sekundit, mille jooksul toimub animatsioon selle kohta, et funktsiooni valitakse parasjagu. Selle ajaraami sees on kasutajal võimalik veel valikut muuta. Samal ajal kui robotkäsi liigub, kuvatakse ekraanile vastav teade ja valikut muuta ei ole võimalik.

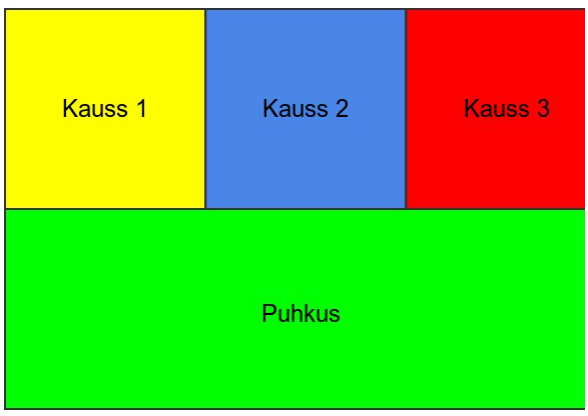
Silmatuvastustarkvara kui ka programmikood töötavad mõlemad arvutis. Kui kasutaja on silmadega valinud soovitava kausi, saadab arvutis töötav programm vastavad käsud servosid

[23] http://answers.eyetechds.com/questions/52/dark-pupil-v-bright-pupil (01.05.2015)
[24] http://www.allaboutvision.com/resources/red-eye-photo.htm (01.05.2015)

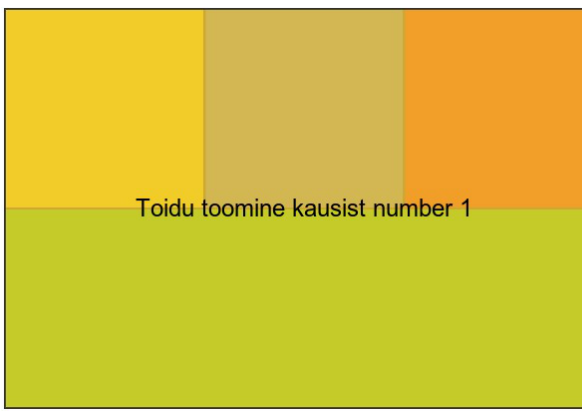
kontrollivale mikrokontrollerile, mis genereerib PWM signaalid servode juhtimiseks.

Esimesel kasutamiskorral tuleb graafilist liidest kasutades seade kalibreerida. Selleks tuleb patsiendil lihtsalt vaadata ekraanile kuvatud punkte. Pärast kalibreerimist saab kasutaja defineerida oma suu asukoha. Seda eelistust saab järgmisel kasutuskorral muuta või ka samaks jätta. Järgnevalt on sammudena toodud seadme kasutamise etapid ning nende kohta käivad kasutajaliidese joonised:

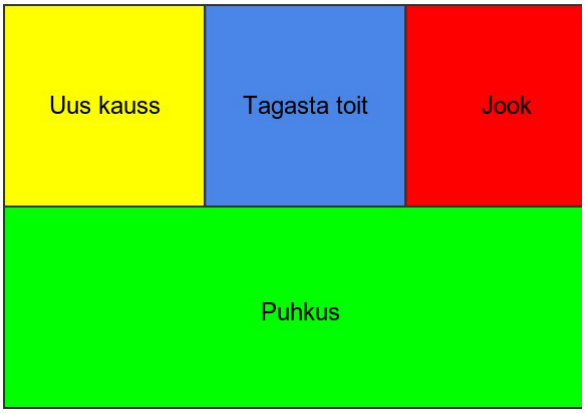
1. assistent valmistab ette toidu(d) ja paigutab kausid selleks ettenähtud kohtadele;
2. kasutaja defineerib oma suu asukoha (kui tahab kasutada uut suu asukohta);
3. kasutaja valib ühe kolmest kausist (kasutades oma silmi ja kasutajaliidest) (joonis 2);
4. robotkäpp kasutab eelprogrammeeritud algoritmi, et võtta kausist lusikatäis ja tuua see patsiendi suuni (joonis 3);
5. kasutaja saab vahetada toitu, viia üleliigne toit tagasi kaussi, valida joomise funktsiooni või puhata (joonis 4);
    a. joomise funktsiooni lõpetamiseks kuvatakse kasutajale erinevat kasutajaliidest (joonis 5);
6. robotkäpp liigub algpositsiooni;
7. korduvad sammud 3-5;
8. assistent eemaldab robotilt kausi ja lusikad ning peseb need puhtaks;



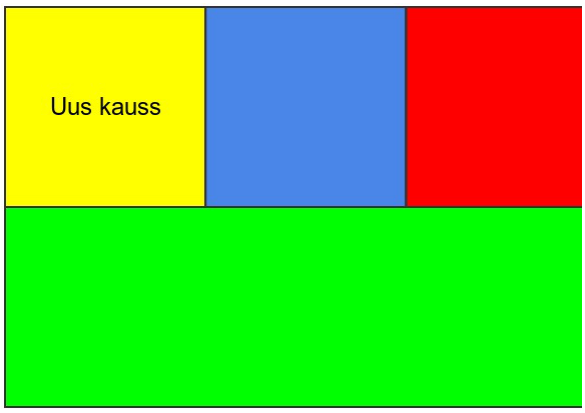
Joonis 2 - ICraft kausi valik



Joonis 3 - ICraft toidu toomine



Joonis 4 - ICraft söömine



Joonis 5 - ICraft joomine

Robotkäe juhtimiseks on kasutusel eelprogrammeeritud algoritmid. Kuna robotil ei ole tagasiside süsteemi, siis on kasutusele võetud meetmed, kuidas siiski toitu kausist ühtlaselt kätte saada. Kui kasutaja valib kausi, siis robot kontrollib esmalt kas kausist on eelnevalt juba söödud. Kui jah, siis lükatakse lusika abil toitu rohkem kausi keskmisse ossa ja seejärel võetakse uuesti toitu lusikale. Veel üks nipp, kuidas toitu ühtlaselt tarbida on, seisneb selles, et robot läheneb kausile iga kord erineva nurga alt. Kauss ise on spetsiaalselt kumerate äärtega, et robotkäsi saaks lusikalt üleliigset toitu maha lükata. Joomise jaoks on samuti spetsiaalne tuub.

ICrafti kasutamise kitsenduseks on patsiendi mõningane pea liigutamise võime ja esialgse kalibreerimise jaoks on vaja juurde ka assistenti (tervet inimest). See ei ole aga probleem, sest hooldaja peab nagunii toidu ette valmistama ja pärast nõud pesema, seega on teine inimene niikuinii kohal.

3.3. Kokkuvõte ja järeldused sarnastest projektidest

Kõikide analüüsitud toodete eesmärk on sama - manustada kausist patsiendile sööki. Kõik süsteemid võimaldavad toitu võtta mitmest erinevast anumast. Meal Buddy roboti konfiguratsioon sisaldab ka toidu koguse seadistamist ning robot lükkab ise üleliigse koguse maha. Viimast teeb ka ICraft, My Spoon roboti kohta sellised andmed puuduvad. Teistega võrreldes on My Spoon robotit võimalik suuremal määral kontrollida. Patsiendil on võimalik

juhtkangi abil valida kauss ja isegi selle abil toitu võtta. Samas on sellel ka võimalus kõike seda automaatselt sooritada. Teised robotid opereerivad ainult eelprogrammeeritud liigutuste abil.

Suurimad erinevused robotite puhul on nende kontrollimise mehhanismid. My Spoon robotit kontrollitakse olenevalt konfiguratsioonist kas nupu või juhtkangi abil, Meal Buddy seade töötab nupule vajutusega ning ICrafti juhitakse silmadega. Seega on viimane aparaat kasutajaliidese poolelt teistest tehnoloogiliselt palju keerukam, kuid samas tagab see ka patsiendile rohkem võimalusi robotit kontrollida.

Maksumuselt on kõik lahendused üsna kallid, ainult ICraft on teistega võrreldes natukene odavam, kuid kui sisse arvutada ka varjatud kulud, siis võib hind ulatuda siiski kallite kommertstoodete tasemele.

Käesolevas töös loodav robot Rebot sarnaneb kõige rohkem ICraftile, kuna mõlemat robotkätt kontrollitakse silmadega. Edasises töös toetutakse suuresti ICraft lahendusele.

4. Analüüs

4.1. Lähtetingimused

Projekti jaoks on eraldatud rahalisi vahendeid mahus 300€. Riistvara valik jääb täielikult inseneride otsustada. Tarkvara kasutamise osas piiranguid ei ole, nii kaua kui projekti eelarve valikute tõttu ei kannata.

4.2. Sihtgrupp

Rebot on mõeldud inimestele, kellel on raskusi iseseisvalt söömisega, olgu siis see seisund tulenevat paralüüsist või amputatsioonist või millestki muust. Kuna toidu peab siiski keegi ette valmistama ja selle ettenähtud kohta paigutama, siis hooldaja juuresolek on seadme kasutamisel osaliselt vajalik. Seade võtab hooldajatelt osa töökoormust ära. Näiteks ema, kes hooldab kodus oma ajukahjustusega last, saab lapse toitmise asemel temaga koos õhtust süüa.

4.3. Eetilised aspektid

Osad inimesed, kellel reaalselt on söömisega raskusi (käed värisevad või on koordinatsiooni häired), võivad seda ise mitte tunnistada ja neile võib robotjäse piinlikkust valmistada. Seega enne Reboti kasutusele võtmist tuleb kindlasti patsiendiga suhelda ja kindlaks teha, kas tal on üldse soovi sellist aparaati kasutada. Inimesele tuleb läheneda individuaalselt ning selgitada kuidas seade töötab ja näidata talle videoid, kuidas teised seda kasutavad. Täiesti elujõus inimestele ei ole seade mõeldud.

4.4. Nõuded patsiendile

Patsient peab:

- saama suud avada, ja huultega lusikalt toitu kätte saama;
- saama olla istukil, 80-90 kraadise nurga all (tahapoole kallutatult);
- saama neelata, närimine on oluline ainult siis, kui kausis on tahkem toit;
- nägema lusikat (kusjuures silmal ei tohi olla olulist defekti, mis takistaks silmatuvastust);
- nägema sihtmärki (punast LED-i)
- aru saama kuidas Rebot töötab;

4.5. Nõuded robotile

Kuna inimeste seisundid on erinevad, siis on väga oluline, et patsient saaks söömise kiirust ise kontrollida.

Peale selle peab robot:

- oskama võtta kausist toitu;
- kasutaja signaali peale toidu tema suuni transportima;
- kasutaja signaali peale tegevust kordama;

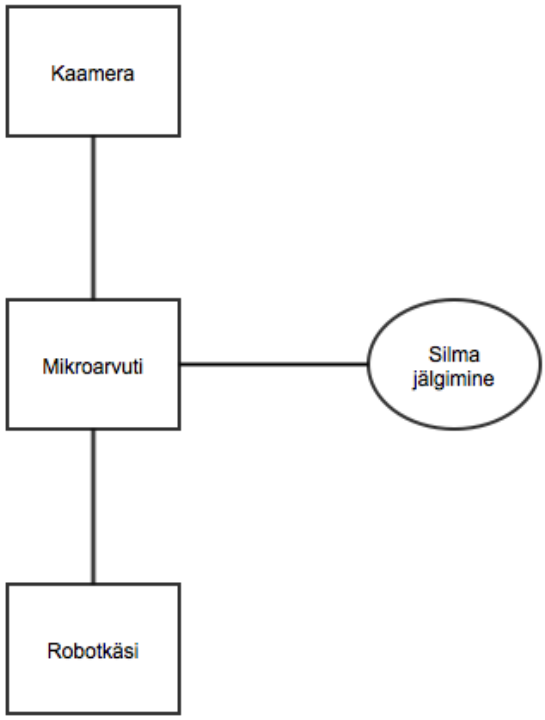
Toitude valikusse jäävad esialgu ainult püreed, supid ja püreesupid. Tahke toidu jaoks Reboti esimene versioon ei sobi.

4.6. Komponendid

Kuna lähteülesanne on sisuliselt sama, mida ICrafti näol on juba tehtud, siis komponendid roboti disainis kattuvad. On vaja platvormi, kus töötaks programmi kood, kaamerat, valgusallikat silma valgustamiseks, robotkäppa, kaussi, lusikat. Käesoleva töö eelarve on aga vaid kolmandik sellest, mis oli kasutada ICrafti arendajatel (ja see ei hõlma isegi varjatud

kulusid). Seega minimeeritakse kulutusi ja ei kasutata kalleid komponente. Välja jäetakse eraldiseisev monitor. See tähendab, et kasutajaliides on vaja muud moodi disainida. Välditakse Microsoft Windows tasulist operatsioonisüsteemi. Tarkvara platvormina valitakse välja midagi väga väikest ja odavat. Selge on see, et sellise hinnaga tavalise PC kasutamine ei ole võimalik, kasutatakse mikroarvutit. Samuti proovitakse vältida eraldiseisvat servokontrollerit ning kauss ei pea olema spetsiaalse kujuga. Lisaks ei kasutata juba valmis robotkäppa.

Kui eelnevalt loetletud komponendid välja jätta, siis roboti üldine skeem näeb välja järgmine:



Joonis 6 - Roboti skeem

Komponentide valik toimub kahes osas. Mikroarvuti ja kaamera valik mõjutavad äärmiselt tugevalt tarkvara arendust, seega otsustab nende komponentide valiku tarkvara arendaja. Ülejäänud komponentide valiku eest vastutab elektroonikainsener ning nende komponentide valimise analüüs on kirjeldatud töö teises osas.

4.6.1. Mikroarvuti

Komponentide valik algab mikroarvuti valikust. Suurimaks kitsenduseks on hind. Samas peab see suutma hea kaadrisagedusega pilditöötlust sooritada. Samuti peab saama seadmega PWM signaali genereerida, kuna robotkäe jaoks kasutatakse servosid, mis kasutavad sisendina just PWM signaali.

Töö kirjutamise ajal on kindlasti kõige kuulsam platvorm Raspberry PI. PI 2 Model B suudab 640x480 resolutsiooni juures töödelda 24-32 kaadrit sekundis,[25] seega on see piisav pilditöötluse jaoks.

Järgnevalt on toodud Raspberry PI 2 Model B riistvara spetsifikatsioon:[26]

- A 900MHz neljatuumaline ARM Cortex-A7 CPU;
- 1GB RAM;
- 4 USB pesa;
- 40 GPIO väljaviiku (sh. 2 riistvaralist PWM kanalit);
- HDMI pesa;
- võrgukaabli pesa;
- 3,5mm heli pesa;
- kaamera liides (CSI);
- ekraani liides (DSI);
- mikro SD kaardi pesa;
- VideoCore IV 3D graafikaprotsessor;

Tähelepanu väärib fakt, et plaadil on olemas kaameraliides ning sobiva kaamera + kaabli saab osta eraldi juurde. Kuna platvorm on kuulus, siis lahendused võimalikele probleemidele on tõenäoliselt internetis olemas.[27] Tarkvara arendamise puhul on suur kasutajate kommuun oluline. Hind Elfas koos käibemaksuga on 43.80€.[28]

[25] http://www.pyimagesearch.com/2015/03/30/accessing-the-raspberry-pi-camera-with-opencv-and-python/ (04.05.2015)
[26] https://www.raspberrypi.org/products/raspberry-pi-2-model-b/ (04.05.2015)
[27] https://www.raspberrypi.org/forums/ (04.05.2015)
[28] https://www.elfa.se/elfa3~ee_et/elfa/init.do?item=30-011-591&toc=0&q=raspberry+pi+2 (04.05.2015)

PI üheks suurimaks konkurendiks on BeagleBone Black:[29]

- AM335x 1GHz ARM® Cortex-A8;
- 512MB DDR3 RAM;
- 4GB 8-bit eMMC flash mälu;
- 3D graafikakiirendi;
- NEON ujuvkoma kiirendi;
- 2x PRU 32-bit mikrokontroller;
- USB (toide + kommunikatsioon);
- USB host;
- võrgukaabli pesa;
- HDMI pesa;
- 2x 46 GPIO väljaviiku (sh. 8 riistvaralist PWM kanalit);

Kommuun on aktsepteeritav, kuid mitte kindlasti nii suur kui RPI-l.[30] Beagle üheks suureks miinuseks on see, et kaamera lisamiseks peab ostma laiendusplaadi ja kaamera sensorplaadi. [31] Samuti ei ole kindel kui pika kaabli otsas kaamera veel head pilti näitab. Kaasas olev kaabel on ainult 2,5 tolli pikk, mis on selgelt liiga vähe. RPI kaamera seevastu annab isegi 4m pika kaabliga hea tulemuse,[32] mis on märkimisväärne. Ametlikult müüakse RPI kaamerat 15cm pikkuse kaabliga. Muidugi saaks kasutada USB kaamerat, aga ei ole täpselt teada kui palju see jõudlust mõjutab ning selle kohta infot ei leitud. BeagleBone Black maskab YEInternationalis koos käibemaksuga 53.76€.[33] Plaat on kallim kui Rasbperry PI 2 B.

Teine konkurent Raspberryle on Banana PI:[34]

- ARM Cortex-A7 dual-core, 1GHz, Mali400MP2 GPU;
- 1GB DDR3 DRAM;
- SD kaardi pesa;
- SATA konnektor (2.5" SATA HDD with 5V);
- HDMI pesa, LVDS konnektor, komposiit video;

[29] http://beagleboard.org/BLACK (04.05.2014)
[30] http://beagleboard.org/Community/Forums (04.05.2014)
[31] http://www.ti.com/devnet/docs/catalog/endequipmentproductfolder.tsp?actionPerformed=productFolder&productId=19580 (04.05.2014)
[32] https://www.raspberrypi.org/forums/viewtopic.php?p=362714 (04.05.2014)
[33] http://yeint.ee/elektroonika-1/arenduskomplektid/beaglebone-black-revision-c (04.05.2014)
[34] http://www.bananapi.org/p/product.html (04.05.2014)

- 3.5mm heli pesa;
- sisseehitatud mikrofon;
- võrgukaabli pesa;
- 2 USB 2.0 pesa;
- OTG mikro USB pesa (saab kasutada ka toite jaoks);
- 26 GPIO väljaviiku;
- kaamera liides (CSI);
- kolm füüsilist nuppu;
- infrapuna andur;

Protsessori taktisagedus veidi kõrgem kui RPI-l, aga samas on see ainult 2-tuumaline, Raspberry 4 vastu. Banana PI koos rohkemate laiendusvõimalustega (olemas on ka CSI kaamera pesa). Kõige suurem probleem platvormi puhul on kommuuni väiksus, seega on see ohtlik valik. Hind samuti veidi kõrgem, Amazonis 53.88€.[35]

Kokkuvõttes võetakse kasutusele Raspberry PI 2 B. See on hästi toetatud ja suure kommuuniga platvorm, vajalikud laiendusvõimalused (näiteks kaamera) on olemas, samuti on hind teiste konkurentidega võrreldes väiksem.

RPI-le on vaja ka eraldi mälukaarti. Selleks võetakse kasutusele 8GB mahuga klass 10 Samsungi mikro SD kaart.

4.6.2. Kaamera

Raspberry PI-ga ühildub kaks ametlikku kaamerat. Üks neist tavaline ja teine ilma infrapuna filtrita kaamera. Kuna patsient võib viibida ka hämaras toas, kasutatakse RPI NOIR kaamerat ja infrapuna valgusallikaid. Nii ei häiri üleliigne nähtav valgus patsiendi heaolu. RPI kaamera plaati on integreeritud fikseeritud fookusega Omnivision 5647 sensor, millel on viis miljonit pikslit. Sensor on tundlik 800-1000nm infrapuna lainepikkusele. Kaamera võimaldab filmida full HD režiimis (1080p) maksimaalselt 30 kaadrit sekundis. Kaamera plaat on mõõtmetega

[35] http://www.amazon.com/Raspberry-Pi-like-devepment-Gigabit-ethernet/dp/B00LGXINGS (04.05.2015)

25 x 20 x 10 mm. 15 cm pikkune 15-sooneline kaabel ühendub kaamera liidesega.[36]

4.6.3. Muud elektroonika komponendid

Muude elektroonika komponentide valiku kohta leiab detailse info projekti elektroonika osast.

Tarkvara arenduse seisukohast on oluline teada, et robotjäseme sisenditeks on universaalsed servode konnektorid. Silma valgustamiseks kasutatakse infrapuna LED-e ning kasutajaliidese lihtsustamise jaoks on üks punane LED.

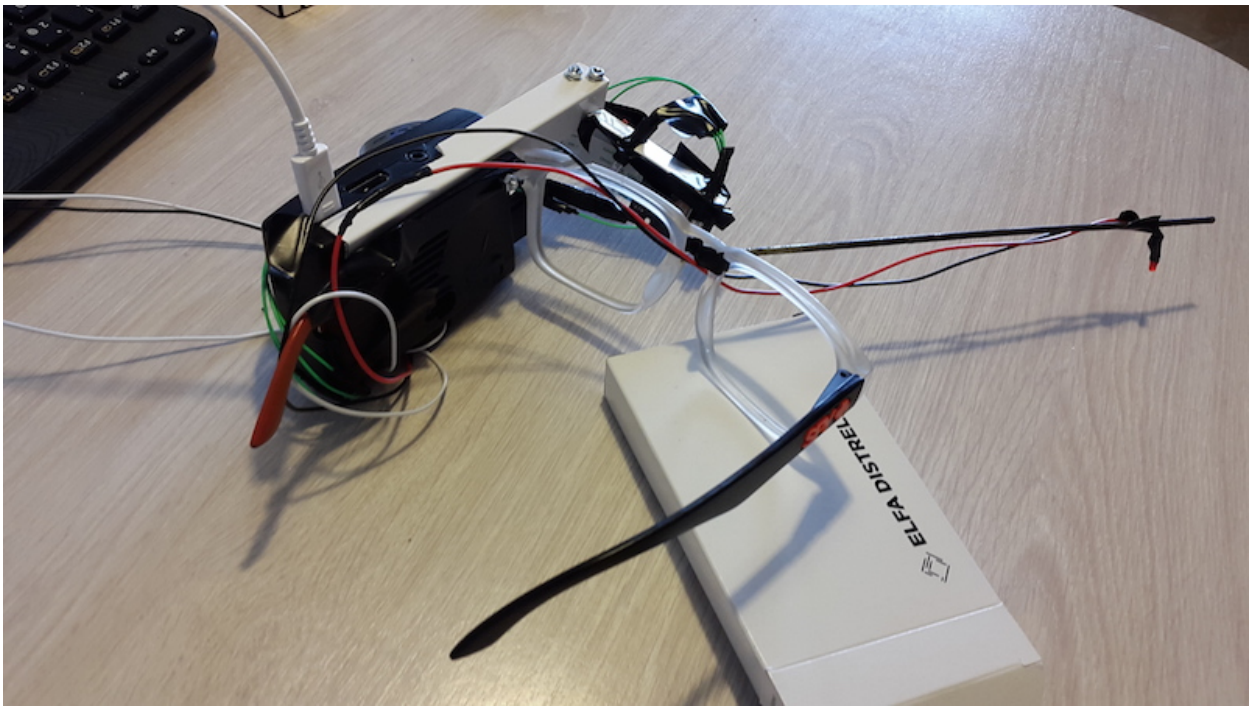
Järgnevalt on toodud projekti kogukulu:

Raspberry PI 2 B	43.80€
Raspberry PI NOIR Kaamera	40.92€[37]
Mikro SD kaart, Samsung	8.00€
Muu (servod, takistid, materjalid jne.)	89.47€
Kokku	182.19€

4.7. Komponentide ühendamine

Silma jälgimise lihtsustamiseks paigutatakse kaamera silmale võimalikult lähedale. Konstrueeritakse spetsiaalsed prillid, mille külge paigaldatakse kaamera ja esimeses versioonis ka Raspberry PI ise, kuna kaameraga kaasa tulnud kaabel on lühike. Kaamera kõrvale (mitte kaamera optilisele teljele) paigutatakse kaks infrapuna LED-i. Veidi eemale, umbes 25 sentimeetri kaugusele prillidest, kinnitatakse sihtmärk (punane LED) (joonis 7). Selle eesmärk on lihtne - kui patsient oma silmadega seda vaatab, siis alustab robot toitmise tsüklit. Sihtmärk paigutatakse prillide külge, kuna siis on patsiendil võimalus pead vabalt liigutada ja sihtmärk jääb silma asukohaga võrreldes ikka samale positsioonile. Raspberry PI-ga ühendatakse ka robotkäpp. Mehaanika ja täpsed ühenduste skeemid leiab projekti elektroonika osast.

[36] https://www.raspberrypi.org/products/camera-module/ (04.05.2015)
[37] https://www.elfa.se/elfa3~ee_et/elfa/init.do?item=10-709-62&toc=0&q=noir (04.05.2015)



Joonis 7 - Prillid

4.8. Tööpõhimõte

Järgnevalt on kirjeldatud sammud, kuidas Rebotit üldjoontes kasutada:

1. patsient istub seljaga põranda suhtes 80-90 kraadise nurga all (tahapoole kaldu);
2. hooldaja paneb patsiendile ette spetsiaalsed prillid;
3. hooldaja liigutab robotkäpa õigesse kohta ning asetab kausi koos toiduga ettenähtud kohale;
4. hooldaja lülitab Reboti sisse;
5. järgneb Reboti kalibreerimine;
6. Rebot võtab esimese lusikatäie toitu lusikale;
7. kui patsient vaatab sihtmärki, siis alustatakse toitmise tsüklit (toiduga lusikas liigub patsiendi suuni);
8. kui patsient vaatab uuesti sihtmärki, liigub robotkäsi tagasi nullasendisse ning võtab uue lusikatäie toitu (nii on inimesel võimalik dünaamiliselt ise valida aega kui kaua aega lusikas suu juures on);

9. korduvad sammud 7-8;
10. Rebot lülitatakse välja;
11. hooldaja eemaldab prillid, peseb nõud;

5. Praktilise osa realiseerimine

5.1. Süsteemi ettevalmistamine

5.1.1. Raspberry PI ülesseadmine

Siin peatükis ei kirjeldata arendaja arvuti töökeskkonna üles seadmist, vaid keskendutakse ainult RPI-le. Käsurea notatsioon on arendaja arvuti (klient) puhul ⇒ ja RPI (server) puhul →.

Esmalt installeeritakse RPI-le operatsioonisüsteem. Kasutades tavalist arvutit, laetakse SD kaardile NOOBS tarkvara. NOOBS on operatsioonisüsteemi installeerimise haldur, mis juba sisaldab Raspbian operatsioonisüsteemi. NOOBS tarkvara levitatakse ametlikult Raspberry PI tootja poolt.[38]

Raspbian on mitteametlik Debian 7 Wheezy ARM hard-float (armhf) port Raspberry PI-le. Töötab ARMv7 ja v6 käsustikega ning on kompileeritud seadetega, mis on optimeeritud Raspberry PI-le. Operatsioonisüsteemi jaoks on olemas üles 35000 eelkompileeritud paki, mida saab kiiresti installeerida.[39] Raspbian ei ole seotud Raspberry PI sihtasutusega, kuid see on RPI tootja poolt soovitatud tarkvara.

RPI külge ühendatakse monitoriga HDMI kaabli abil, külge pannakse USB klaviatuur, hiir,

[38] https://www.raspberrypi.org/downloads/ (05.05.2015)
[39] http://raspbian.org/ (05.05.2015)

WIFI pulk ning mälukaardi pessa sisestatakse SD kaart. Lõpuks ühendatakse RPI vooluvõrku.

Raspbiani installeerimise ajal konfigureeritakse Eesti ajatsoon, võimaldatakse kaamera kasutamine ja SSH ligipääs süsteemile. Süsteem taaskäivitatakse käsuga `sudo reboot`. Pärast sisselogimist luuakse *X Window System* sessioon käsuga `startx`, konfigureeritakse WIFI võrk ning uuendatakse läbi terminali emulaatori süsteemi pakid viimastele versioonidele:

```
→ sudo apt-get update
→ sudo apt-get upgrade
```

Järgnevalt installeeritakse arendaja avalik SSH võti RPI süsteemi. See võimaldab ilma paroolita serverisse sisse logida. Kitsenduseks on kliendi privaatvõtme salasõna puudumine. Antud juhul on kriteerium täidetud. Ühtlasi lisatakse sellega automaatselt RPI host kliendi known_hosts faili.

```
⇒ cat ~/.ssh/id_rsa.pub | ssh pi@192.168.0.19 "mkdir -p
~/.ssh && cat >> ~/.ssh/authorized_keys"
```

Programmi koodi halduse lihtsustamiseks kasutatakse Git[40] versioonihaldustarkvara. Koodi kirjutamise ajal on koodi majutusteenuseks BitBucket,[41] kuna see võimaldab tasuta privaatseid repositooriume luua. Projekti valmides tõstetakse kood Githubi[42] ja tehakse avalikuks MIT litsentsi alusel.

Luuakse Rebot projekt:

```
⇒ mkdir ~/projects/rebot -p
```

[40] http://git-scm.com/ (06.05.2015)
[41] https://bitbucket.org/ (06.05.2015)
[42] https://github.com/ (06.05.2015)

```
⇒ cd ~/projects/rebot
⇒ git init
⇒ git remote add origin
git@bitbucket.org:mlensment/rebot.git
⇒ touch README.md
⇒ git add .
⇒ git commit -m "Initial commit"
⇒ git push -u origin master
```

Et RPI süsteemist saaks koodi alla laadida, peab sinna genereerima SSH võtmed:

```
→ ssh-keygen -t rsa -C "pi@raspberry"
→ eval "$(ssh-agent -s)"
→ ssh-add ~/.ssh/id_rsa
```

Avalik võti kopeeritakse BitBucketisse ja repositoorium kloonitakse:

```
→ mkdir ~/projects
→ cd projects
→ git clone git@bitbucket.org:mlensment/rebot.git
```

Koodi uuendamine käib käsuga:

```
→ git pull origin master
```

Iga kord kui kood uueneb, on tülikas käsitsi koodi alla laadida. Selle protsessi lihtsustamiseks luuakse lühike *shell* skript, mida käivitab kliendi arvutis antud juhul Bash.

```
⇒ nano commit.sh
```

```
#!/bin/bash

git add --all
if [ -z "$1" ]; then
  git commit -m "Update"
else
  git commit -m "$1"
fi

git push origin master && ssh pi@192.168.0.19 'cd
~/projects/rebot && git pull origin master'
```

Iga Giti toimingu juurde saab lisada kommentaari, vaikeväärtuseks on kommentaariks sõna "Update."

Välise monitori ühendamine on tülikas ja järgmisena installeeritakse ja konfigureeritakse VNC server, et kliendi arvutist saaks näha RPI graafilist liidest.

```
→ sudo apt-get install tightvncserver
→ tightvncserver
```

Luuakse skriptid, mille abil saab serverit mugavalt käivitada ja peatada:

```
→ nano ~/startvnc.sh
```

```
#!/bin/bash
vncserver :0 -geometry 1366x768 -depth 24 -dpi 96
echo "vnc://$(ifconfig wlan0 | awk '/t
addr:/{gsub(/.*:/,"",$2);print$2}')"
```

Viimane rida skriptist ei ole tegelikult serveri käivitamiseks vajalik, tegu on mugavuslahendusega, mis trükib välja protokolli ja RPI juhtmevaba võrgu IP. Selle saab

kiiresti kopeerida VNC klienti.

```
→ nano ~/stopvnc.sh
```

```
#!/bin/bash
/usr/bin/vncserver -kill :0
```

```
→ chmod +x ~/stopvnc.sh ~/startvnc.sh
```

5.1.2. Kaamera testimine

```
→ sudo apt-get install python-picamera
```

Python-picamera on vabavaraline tarkvara, mis pakub puhast Pythoni liidest Raspberry PI kaamerale. Toetatud on Pythoni versioonid >= 2.7 ja >= 3.2.[43]

Programmeerimiskeelena kasutatakse Pythonit, kuna see on Raspbianis eelinstalleeritud ja platvormil hästi toetatud. Pythoni pakkide repositooriumis on neid kokku üle 50 000.[44] Kommuun on suur ja üsna küps (keel on loodud aastal 1991)[45] ja on kiiresti populaarsust kogunud.

```
→ mkdir ~/projects/camera-test -p
→ cd ~/projects/camera-test
→ nano camera.py
```

[43] https://picamera.readthedocs.org/en/release-1.10/ (06.05.2015)
[44] https://pypi.python.org/pypi (06.05.2015)
[45] http://groups.engin.umd.umich.edu/CIS/course.des/cis400/python/python.html (03.06.2015)

Järgneb programmikood:

```
import picamera
camera = picamera.PiCamera()
camera.capture('image.jpg')
```

Programmi käivitamine:

```
→ python camera.py
```

Failihalduriga vaadates on projekti kausta tekkinud fail image.jpg, seega tähendab, et kaamera töötab.

Järgmine ülesanne on konfigureerida kaamera töötama video seadmena. Selleks on vaja installeerida UV4L draiver.[46] Selleks lisatakse uus pakihalduri allikas ja installeeritakse draiver:

```
→ curl
http://www.linux-projects.org/listing/uv4l_repo/lrkey.asc |
sudo apt-key add -
→ sudo bash -c "echo 'deb
http://www.linux-projects.org/listing/uv4l_repo/raspbian/
wheezy main' >> /etc/apt/sources.list"
→ sudo apt-get update
→ sudo apt-get install uv4l uv4l-raspicam
uv4l-raspicam-extras
→ sudo nano /etc/uv4l/uv4l-raspicam.conf
```

```
nopreview = yes
```

[46] https://www.raspberrypi.org/forums/viewtopic.php?t=57788 (06.05.2015)

```
→ sudo service uv4l_raspicam restart
→ echo 'export
LD_PRELOAD=/usr/lib/uv4l/uv4lext/armv6l/libuv4lext.so' >>
~/.bashrc
sudo bash -c "echo 'export
LD_PRELOAD=/usr/lib/uv4l/uv4lext/armv6l/libuv4lext.so' >>
/root/.bashrc"
→ source ~/.bashrc
```

Masinnägemise platvormina kasutatakse OpenCV-d. See on populaarne raamistik, millel on suur kommuun (peaaegu 50 000 inimest) ja allalaadimisi üle 9 miljoni. Masinõppimise ja -nägemise jaoks on platvormil üle 2500 optimeeritud algoritmi. Tarkvara kasutavad paljud suured firmad ja kasutusalad ulatuvad tipptasemel robootikast kuni lennukite maandumisradade puhtuse tuvastamiseni. Sellest on avaldatud ka kümneid raamatuid. Programmeeritud C/C++ keeltes, kuid toetab liideseid erinevatesse keeltesse, seal hulgas ka Pythonisse.[47] Paraku ei ole töö kirjutamise ajal OpenCV 3.0 veel väljas, seega ei saa projektis kasutada Pythoni uusimat versiooni, kuna OpenCV stabiilses redaktsioonis vastavat liidest veel pole.[48] Hetkel kasutatakse aja kokkuhoiu mõttes OpenCV eelkompileeritud versiooni 2.4.1 ja Pythoni versiooni 2.7.3.

Kuna OpenCV õppimiskõver on järsk, katsetati korraks ka SimpleCV-d,[49] mis on sisuliselt lihtsam ümbris OpenCV keerulistele funktsioonidele. See osutus aga liiga aeglaseks ja selle installeerimisprotseduuri siin dokumendis pikemalt ei kirjeldata.

```
→ sudo apt-get install python-opencv
→ mkdir ~/projects/video-test -p
→ cd ~/projects/video-test
→ nano video.py
```

```
import cv2
```

[47] http://opencv.org/ (07.05.2015)
[48] http://opencv.org/opencv-3-0-alpha.html (07.05.2015)
[49] http://simplecv.org/ (07.05.2015)

```
import cv2.cv as cv

cap = cv2.VideoCapture(0)
width, height = 320, 320
cap.set(cv.CV_CAP_PROP_FRAME_WIDTH, width)
cap.set(cv.CV_CAP_PROP_FRAME_HEIGHT, height)

while(True):
     ret, frame = cap.read()
     cv2.imshow('frame', frame)
     if cv2.waitKey(1) & 0xFF == ord('q'):
     break

cap.release()
cv2.destroyAllWindows()
```

Programm impordib OpenCV teegi, seab kaamera resolutsiooni, käivitab programmi põhitsükli, mis loeb ükshaaval kaadreid ja näitab neid aknas. Kui kasutaja vajutab nupule “q”, põhitsükkel katkestatakse, kaamera suletakse ja aken pannakse kinni.

Kaamera töötamise ajal põleb plaadil punane tuli, mis on silmale väga ebameeldiv. Järgnevalt muudetakse operatsioonisüsteemi konfiguratsiooni, et punane LED kustutada:

```
→ sudo bash -c "echo 'disable_camera_led=1' >>
/boot/config.txt"
→ sudo reboot
```

Pärast sisselogimist käivitatakse video test uuesti ja punane tuli enam ei põle.

5.1.3. Servodraiveri installeerimine

Raspberry PI 2 B sisaldab ainult kahte riistvaralist PWM väljaviiku. Projekti algusjärgus ei ole kindel mitut servot robotkäsi sisaldama hakkab. Seega peab tarkvara pool olema valmis ka rohkemate servode kasutamiseks. Otse programmi koodis on PWM signaali genereerimine

RPI-l võimalik, aga kuna Raspbian ei ole reaalaja operatsioonisüsteem, siis servode liikumine on hüplik. Põhjus on selles, et CPU aeg on kerneli poolt protsesside vahel jaotatud.[50] Kui roboti programm protsessori aega ootab, siis tekivad PWM signaali vead ning servode liikumine ei ole enam ühtlane. Sellises olukorras kasutatakse sageli eraldiseisvat servokontrollerit nagu seda tehti ka ICrafti projektis. Käesolevas töös aga selle kasutamist välditakse ja installeeritakse hoopis RPI-le mõeldud servodraiver Servoblaster.[51] See tarkvara võimaldab vaikimisi seadistuses kontrollida kuni kaheksat servot. Tööpõhimõte seisneb omavahel seotud DMA kontrollblokkide loomises, kus ka esimene ja viimane blokk on omavahel seotud. Nii saab DMA kontroller lõpmatuseni ringiratast käia ja PWM signaali hoida ilma, et CPU peaks sekkuma. Taolisi draivereid on teisigi, kuid esimesena katsetatakse Servoblasterit.

Arhiiv laetakse RPI-sse,[52] järgneb kompileerimine:

```
→ tar -xzf ServoBlaster-20150219.tgz
→ cd ServoBlaster
→ make
→ mkdir ~/drivers
→ mv servod ~/drivers
→ cd ..
→ rm -rf ServoBlaster-20150219.tgz ServoBlaster
```

Et servo draiver automaatselt süsteemi käivitamise ajal laadida, luuakse uus skript, mis käivitatakse läbi rc.local faili. See omakorda käivitatakse, kui kõik tavalised süsteemi teenused on laetud.

```
→ nano ~/init.sh
```

```
#!/bin/bash
# This file is executed on boot as root via /etc/rc.local
echo "Loading servo driver..."
/home/pi/drivers/servod --step-size 2 --min 280
```

[50] http://www.linuxjournal.com/magazine/completely-fair-scheduler (07.05.2105)
[51] https://github.com/richardghirst/PiBits/tree/master/ServoBlaster (07.05.2015)
[52] https://www.raspberrypi.org/forums/viewtopic.php?f=28&t=99115&start=47 (07.05.2015)

Servodraiver konfigureeritakse nii, et impulsi laiuse samm oleks võimalikult väike. Seda selleks, et tarkvara kaudu saaks servosid maksimaalselt sujuvalt juhtida. Vaikimisi oleks samm 10 mikrosekundit, Rebot kasutab 2 mikrosekundilist sammu. Servode kaitsmise jaoks konfigureeritakse minimaalne lubatud impulsi laius 280 sammuks. RPI süsteemi kiibi PWM välisseade on jagatud RPI helisüsteemiga.[53] Kuna aga heliväljundit robotis ei kasutata, siis seda vaikimisi seadet ei muudeta.

```
→ chmod +x ~/init.sh && sudo nano /etc/rc.local
```

Enne viimast rida:

```
/home/pi/init.sh &
```

```
→ sudo poweroff
```

Servo signaali juhe ühendatakse RPI GPIO-23 külge, mis vastab servodraiveri kaardi järgi viiendale servole. Täpne servode ühenduste skeem on leitav töö teisest osast. Süsteem lülitatakse sisse ja pärast sisselogimist testitakse servot:

```
→ echo 5=1100 > /dev/servoblaster
→ echo 5=280 > /dev/servoblaster
```

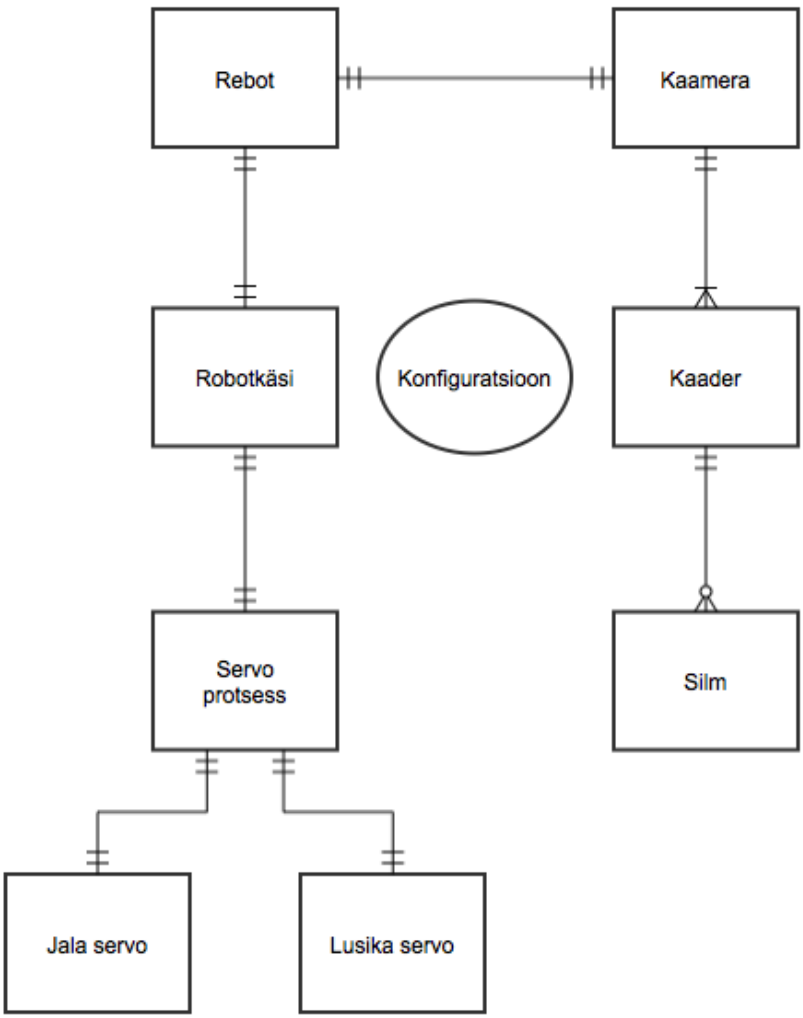
Servo töötab hästi ja muid draivereid ei katsetata.

[53] http://hertaville.com/2014/07/07/rpipwm/ (07.05.2015)

5.2. Tarkvara arhitektuur

Programm realiseeritakse täielikult objekt orienteeritult.

Järgnevalt on toodud klassid ja nende seosed:



Joonis 8 - Tarkvara arhitektuur

Joonis on tehtud kasutades ERD notatsiooni. Kõik klassid kasutavad ühte globaalset konfiguratsiooni. Programmi klasside tööpõhimõte on lahti seletatud järgnevates peatükkides.

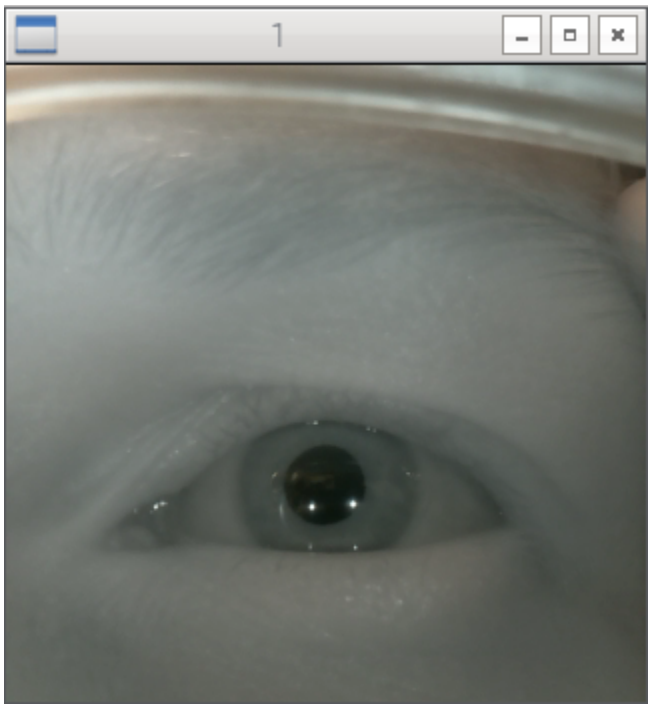
Servo protsessi klassi on vaja selleks, et servosid sujuvalt kontrollida. Klass kaader sisaldab silma tuvastuse jaoks vajalikku algoritmi ning selle käitamine on protsessori jaoks programmi

kõige kulukam operatsioon. Et servode liikumist pildituvastuse ajal mitte blokeerida, viiakse servode liigutamine Reboti alamprotsessile. Esialgne idee oli kasutada servode jaoks lihtsalt eraldi lõimesid, aga nagu Ruby MRI-s (tarkvarainsener on Ruby taustaga), on ka CPythonis GIL, mis ei luba lõimedel paralleelselt protsessoril töötada ja mitut tuuma utiliseerida. CPythoni implementatsioonis on lõimed vaid kasulikud siis, kui programm ootab I/O taga.[54] I/O pole aga antud juhul probleem, kuna kaadri lugemine kaamerast on kiire ja aeganõudev protsess on pilditöötlus ise. Selleks, et lõimedest kasu saada, peaks vahetama Pythoni implementatsiooni IronPythoni[55] või Jythoni[56] vastu. Need on vastavalt .NET ja Java implementatsioonid. See on aga palju keerulisem lahendus kui lihtsalt luua Rebotile servode kontrollimiseks alamprotsess ja jagada protsesside vahel mälu. Pythonis on vastavad teegid selleks olemas.[57] Paralleelprogrammeerimise paradigma on võrreldes järjestikuse programmeerimisega kordades keerukam, kuna kõik protsessid peavad arvestama, et nad jagatud mälus midagi valel ajal üle ei kirjutaks. Servo protsessi on sisse ehitatud kaitsemehhanismid, mis tagavad vigadeta programmi töö jätkumise isegi juhul, kui midagi sellist peaks juhtuma. Veaohtlikke olukordi on mitu, kuid põhiline neist tekib siis, kui peaprotsess puhastab servode sisemisi käsupuhvreid ning servode positsioonide kalkuleerimine on sellel hetkel alles poolepeal.

5.3. Silmatuvastus

Silmatuvastus käib läbi nelja objekti. Läbi Reboti peatsükli loeb kaamera objekt olenevalt Reboti konfiguratsioonist füüsilisest kaamerast või failisüsteemist kaadri (pildi) ja pöörab seda 90 kraadi, kuna prillide küljes on kaamera 90-kraadise nurga all. Saadud info alusel initsialiseeritakse kaadri objekt. Nimetatud objekt oskab enda pildi infost üles leida pupilli keskpunkti koordinaadid, pupilli raadiuse ja pindala. Veaotsimise režiimis oskab kaadri objekt töödeldud pilti kuvada.

[54] http://www.dabeaz.com/python/GIL.pdf (07.05.2015)
[55] http://ironpython.net/ (07.05.2015)
[56] http://www.jython.org/ (07.05.2015)
[57] https://docs.python.org/2/library/multiprocessing.html (07.05.2015)



Joonis 9 - Töötlemata kaader

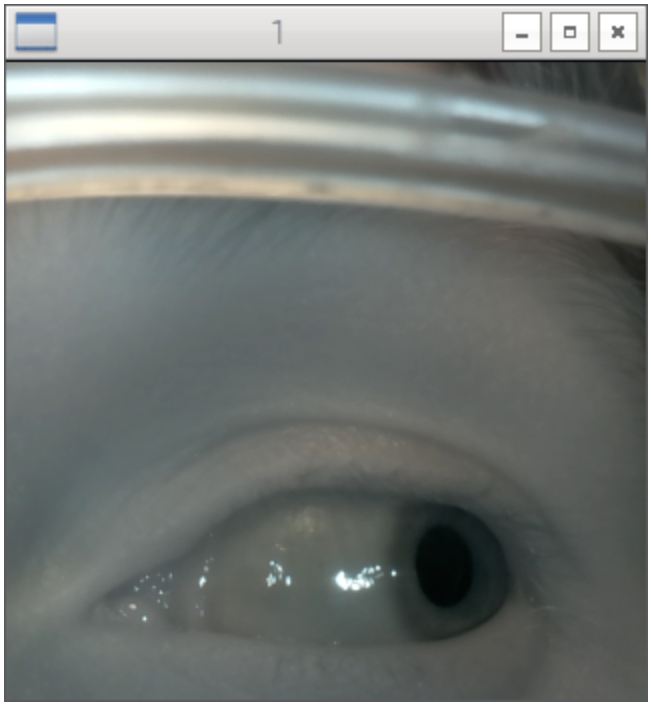
ICrafti projektis rakendati tumeda pupilli tehnikat selleks, et hinnata kuhu patsient parasjagu ekraanil vaatab. See sisaldas infrapuna valgusallika peegelduse ja pupilli keskpunkti vahelise vektori välja arvutamist ja selle abil kasutaja pilgu asukoha hindamist ekraanil.

Kuna aega Reboti projektis on sihtmärk kinnitatud prillide külge, siis piisab sellest, kui teada on ainult pupilli keskpunkt, sest kaamera, silm ja sihtmärk on alati üksteise suhtes samas asendis.

Kuigi käesolevas projektis peegelduste järgi pilgu suunda otseselt ei hinnata, tundub, et peegeldusi saab siiski pupilli otsimise lihtsustamiseks ära kasutada, kuna need liiguvad silma pööramisel pupilliga mingil määral kaasa.

Kõige lihtsamalt saab peegeldusi otsida kui pilt konverteerida mustvalgeks ja rakendada tehnikat, mille nimetus inglise keeles on *thresholding*. Sisuliselt see tähendab seda, et kõikidest pildi pikslitest liigutakse üle ning kui piksli heledus ületab mingit eeldefineeritud väärtust, muudetakse piksel valgeks. Kui piksel ei ületa väärtust, muudetakse see mustaks. Tulemuseks on kaks heledat täppi mustal taustal, mille koordinaadid saab OpenCV abil kätte.

Järgnevalt saab peegelduste ümbrusest otsida tumedat ala - pupilli ennast. Paraku, kui patsient vaatab kõrvale, ilmub silmale palju heledaid peegeldusi, mille hulgast on raske kindlaks teha õigeid infrapuna LED-ide peegeldusi (joonis 9). Seega, seda ideed pupilli tuvastamiseks kasutada ei saa.

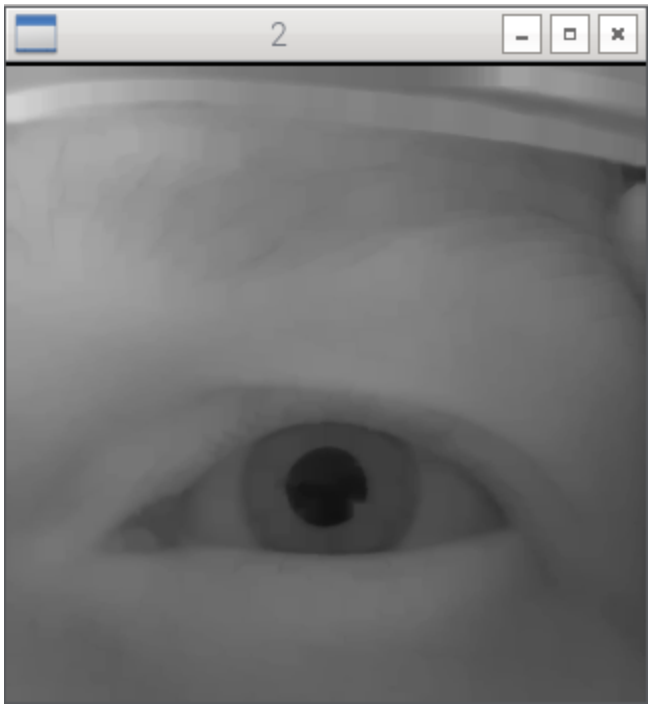


Joonis 10 - Peegeldused silmal

Kuna heledaid peegeldusi kasutada ei saa, otsustatakse need üldse pildilt eemaldada kasutades matemaatilise morfoloogia põhifunktsioone, mis on OpenCV-s implementeeritud. Pilt konverteeritakse esmalt mustvalgesse värviruumi, järgmisena rakendatakse morfoloogilise avamise meetodit,[58] mis koosneb tegelikult kahest osast: uhtumiset[59] ja paisutamisest.[60] Mõlema operatsiooni jaoks valitakse struktuurelement (kernel). Praegusel juhul kasutatakse selleks 9x9 piksli suurust ruudukujulist kernelit. Katsetati ka ovaalset kernelit, kuna peegeldused silmal on valdavalt ümmargused, aga see ei andnud märkimisväärselt paremaid tulemusi. Binaarse uhtumise põhimõte seisneb selles, et kui kõik pikslid kerneli all on väärtusega 1, arvestatakse kerneli keskmise piksli all olevat pikslit valgena ja määratakse talle väljundpildi maatriksis väärtus 1. Vastasel juhul väärtus 0.

[58] http://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm (08.05.2015)
[59] http://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm (08.05.2015)
[60] http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm (08.05.2015)

Mittebinaarse mustvalge uhtumise korral määratakse piksli väärtuseks minimaalne naabruses oleva piksli väärtus.[61] Tulemusena heledad alad kahanevad ning tumedad alad suurenevad. Pärast uhtumist peab tumedaid alasid uuesti vähendama, et pildi proportsioonid paigast ei läheks. Kasutatakse pärast uhtumist selle vastandfunktsiooni - paisutamist, mis kasvatab kahandab uuesti tumedaid alasid.

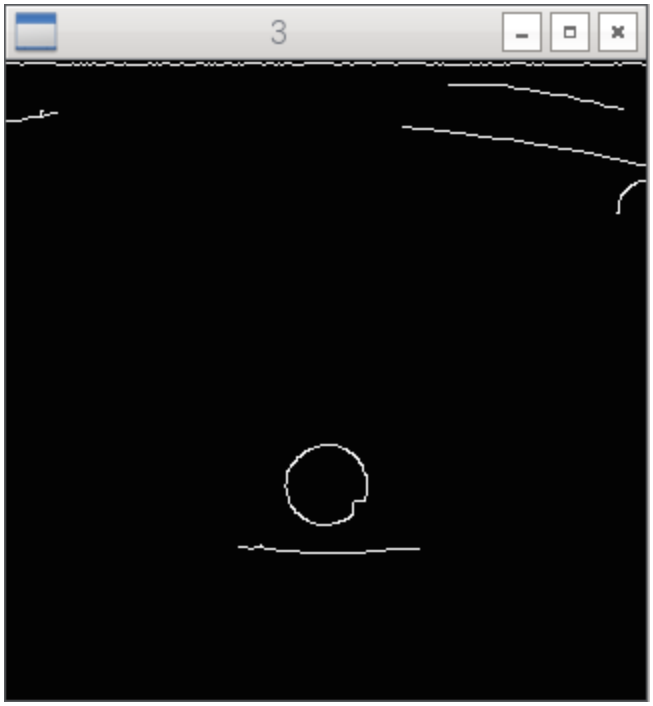


Joonis 11 - Morfoloogiliselt avatud pilt

Pärast esialgset töötlust on silmalt kadunud heledad peegeldused. Edasi tuleb kaardistada kõik servad. Selleks kasutatakse teadlase John F. Canny väljaarendatud ja ka tema nime kandvat Canny servade tuvastuse algoritmi.[62] Matemaatika selle protsessi taga on üsna keeruline ja sellesse siin töös pikalt ei süveneta, kuna see ei ole programmeerimise kontekstis oluline. Serv on sisuliselt punkt pildil, kus ühel pool punkti on väga madal väärtus ja teisel pool kõrge. Need kohad võimendatakse spetsiaalse operaatori abi, mida kutsutakse Sobeli filtriks.[63] Järgmiseks tekitatakse gradientide analüüsi käigus binaarne pilt, mis sisaldab peeneid jooni, mis on servad. Viimases etapis filtreeritakse joontest välja reaalsed servad. Selle jaoks on vaja funktsiooni sisendina anda minimaalne ja maksimaalne piirväärtus. Kõik

[61] http://www-rohan.sdsu.edu/doc/matlab/toolbox/images/morph4.html (08.05.2015)
[62] http://homepages.inf.ed.ac.uk/rbf/HIPR2/canny.htm (08.05.2015)
[63] http://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm (08.05.2015)

servad, mille intensiivsuse gradient ületab seatud maksimaalset piirväärtust, on kindlasti servad ja kõik, mis on allapoole minimaalset piirväärtust, ei ole kindlasti servad ja pildil need peidetakse. Piirväärtuste vahele jäävad alad on servad ainult siis, kui nad on ühendatud mõne joonega, mis on ülalpool piirväärtust.[64] Piirväärtused seadistatakse igale pildile individuaalselt.



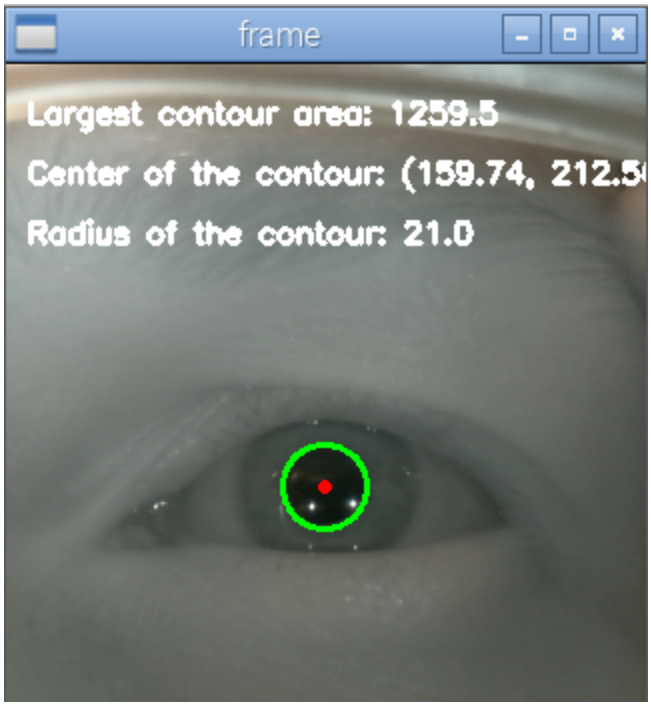
Joonis 12 - Pilt pärast Canny servaotsimist

Nagu joonis 12 näitab, on pupill selgelt välja joonistunud. Järgneb pildi kontuuride tuvastamine. Kontuur on sisuliselt hulk punkte pildil, mis on omavahel seotud. OpenCV-s on kontuuride tuvastamise vahendid olemas.[65]

Kuna leitud kontuure on mitu tükki, siis tuleb nende hulgast välja filtreerida pupilli oma. Pupill saab olla ainult pildi keskosas, seega pildi ülasosas ja alaosas paiknevad kontuurid filtreeritakse välja. Järele jäänud kontuuride hulgast leitakse kõige suurema pindalaga element ning kontuuri ümber projekteeritakse ring. Ringi raadius peab vastama pupilli pindalale kehtestatud vahemikule. Tulemuseks saadud kontuur on pupill. Silmaotsimise

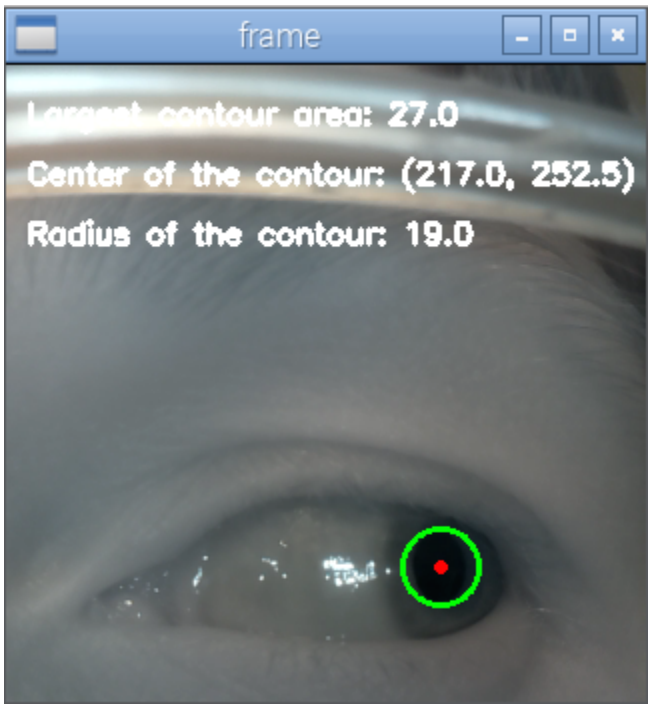
[64] http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html (08.05.2015)
[65] http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_contours/py_contours_begin/py_contours_begin.html (08.05.2015)

funktsioon konstrueerib silma objekti, mis sisaldab pupilli keskpunkti x- ja y-koordinaate. Kui kaadrilt pupilli ei leitud, konstrueeritakse silma objekt, mis ei sisalda pupilli koordinaate. Veaotsimise režiimis joonistatakse originaalpildile pupilli andmeid ning kuvatakse see (joonis 13).



Joonis 13 - Tuvastatud pupill

Arendamise ajal testis silmatuvastuse algoritmi kokku neli inimest: neist üks pruunide ning ülejäänud sinakas-hallide silmadega. Kõigi inimeste puhul andis tarkvara häid tulemusi. Algoritm suudab tuvastada pupilli ka siis, kui silmal on palju peegeldusi (joonis 14).



Joonis 14 - Tuvastatud pupill peegeldustega

5.4. Kaamera kalibreerimine

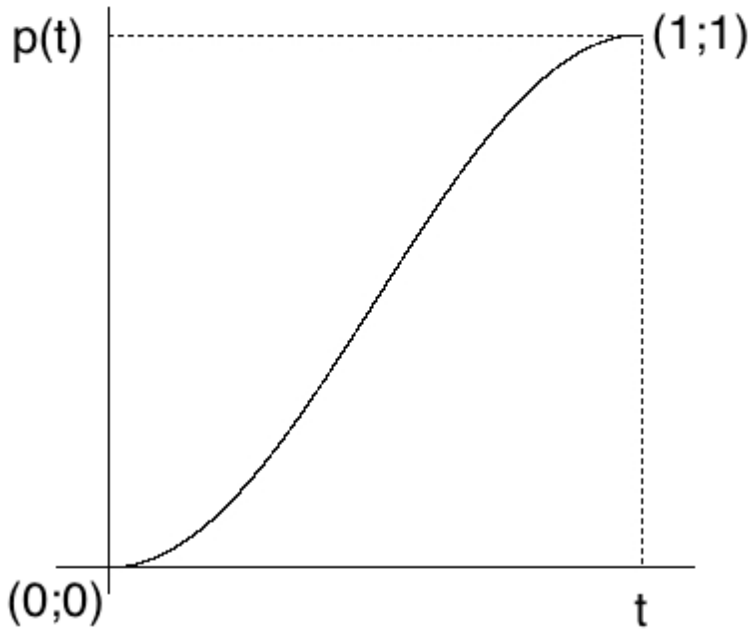
Reboti käivitumise ajal sooritatakse kaamera kalibreerimine. Selle protsessi toimumise ajal põleb punane LED, mida patsient sel ajal vaatama peab. Kaamerast loetakse järjest kaadreid ja tuvastatakse igal kaadril pupilli asukoht. Kui protsessi on korratud 40 (väärtus konfigureeritav) korda, arvutatakse kõikide saadud x ja y koordinaatide mediaanväärtused. Tulemus salvestatakse silma klassimuutujasse. Kalibreerimise lõppemisel kustub LED ja patsient peab mujale vaatama. Pärast programmi peatsükli käivitumist alustab robotkäsi tööd siis, kui patsient uuesti LED-i vaatab. Sihtmärgi vaatamisel süttib LED põlema, et anda patsiendile tagasisidet. Sihtmärgi vaatamisel on lubatud väike viga, sest inimene ei pruugi iga kord 100% samasse kohta vaadata. Samuti võivad prillid peas natukene liikuda.

Silmapilgutuste filtreerimiseks kasutatakse silma klassis puhvrit, mis ignoreerib olukordi kui kaadris silma ei tuvastatud. Sel puhul hoiab Rebot eelmise tsükli staatust. Tegevuse kinnitamise jaoks on silma objektis sisemine taimer, mille aega saab konfiguratsioonis lihtsasti muuta.

5.5. Servode juhtimine

Servosid kontrollib robotkäpa objekt. See hoiab mõlema servo staatusi meeles ning vastavalt hetke staatustele ja silma asendile otsustab, milliseid käske objektidele järgmisena saata. Servod on kättesaadavad Reboti põhiprotsessi ja servoprotsessi vahel jagatud mälu kaudu. Servod ise hoiavad samuti oma sisemist staatust. Mälus on käskluste järjekord, hetkel käimas olev operatsioon, hetke nurk ja kas servo on kõik käsklused lõpetanud. Käsklusi on kahte tüüpi - ootamine ja pöörlemine. Servo objektile saab pöörlemiseks ette anda vajaliku nurga. Objekt oskab ise selle ümber arvutada õigeks PWM impulsi vahemikuks ja suhtleb otse servodraiveriga. Oluline aspekt sealjuures on see, et servodraiveri impulsi samm oleks programmiga vastavuses.

Servode maksimaalselt sujuvaks kontrollimiseks arvutatakse servode positsioone läbi funktsiooni, mis on illustreeritud joonisel 15.



Joonis 15 - Servode kiiruse piiramiseks kasutatav funktsioon

Servo pöörlemise käsk algab punktis (0;0) ja lõppeb punktis (1;1). X teljel on kujutatud aeg, Y teljel servo positsioon (nurk). Seega on joonisel kujutatud servo liikumist ühe sekundi

jooksul positsiooni 1. Jooniselt on näha, et servo liikumise alguses muutub servo positsioon aeglaselt. Kui servo liikumiseks kulunud aeg jõuab umbes kolmandikuni ettenähtud ajaraamist, muutub servo positsioon maksimaalse kiirusega ning liikumise lõpus servo taas aeglustab. Funktsiooni analüütiline kuju on järgmine:

$$p = -\frac{|e-b|}{2} * \left(cos\left(\frac{\pi * t_1}{t_2}\right) - 1\right)$$

Kus e on servo lõpp-positsioon, b on servo algpositsioon, t_1 on kulunud aeg, t_2 on liikumiseks ettenähtud aeg.

5.6. Servode kalibreerimine

Servodraiveri eripära tõttu ei saa selle kaudu lugeda millises positsioonis parasjagu servod on. Et robotkäsi ei hüppaks programmi käivitumisel liiga kiiresti nullasendisse, kalibreeritakse Reboti käivitumise ajal mõlemad servod. Sisuliselt tähendab see seda, et servod viiakse aeglaselt nullasendisse. Selleks võetakse konfiguratsioonist servode maksimaalne impulsi ulatus ning vähendatakse tasapisi väärtust kuni jõutakse nulli. Pärast seda seatakse muutuja selle kohta, et robotkäsi on ennast initsialiseerinud.

5.7. Reboti alglaadimine

Pythoni RPi.GPIO teek ei võimalda selle kasutamist tavakasutaja õigustes, seega tuleb Rebot käivitada juurkasutaja õigustes. Kuna `sudo` käsk ei säilita UV4L draiveri jaoks vajalikku keskkonnamuutujat, on vaja see enne programmi käivitamist eraldi eksportida. Protsessi lihtsustamiseks luuakse käivitatav skript, mis lisatakse ka projekti repositooriumisse:

```
#!/bin/bash

export LD_PRELOAD=/usr/lib/uv4l/uv4lext/armv6l/libuv4lext.so
```

```
python rebot.py $@
```

Skript võtab sisendina argumente, mis edastatakse otse Reboti programmile. Ühe argumendina on võimalik ette anda kaadri asukoht failisüsteemis. Kui atribuut on seatud, loeb kaamera objekt selle enda vahemällu ning kasutab iga kaadri lugemise ajal ühte ja sama puhverdatud kaadrit. Kui failisüsteemist etteantud kaadrit ei leita, salvestab kaamera ühe kaadri kettale ja puhverdab selle endale mällu. Teine argument on veaotsimise režiimi kasutamine. Alternatiivina saab käivitada programmi ka otse juurkasutajana, sel juhul on vajalik keskkonna muutuja juba eksporditud, kuna UV4L draiver konfigureeriti vastavalt.

Programmi alglaadimise ajal sooritatakse hulk operatsioone. Järgnevalt on toodud Reboti initsialiseerimise logi:

```
---------- BOOT LOG -----------
--> Starting Rebot...
----> Checking servo driver...
----> Servo driver loaded
--> Initiating camera calibration sequence...
----> Initiating servo calibration sequence...
----> Servo calibration complete
--> Camera calibration complete
--> Initiating main loop...
--> Rebot is running
```

Lühike nool tähendab programmi peaprotsessi, pikk nool alamprotsessi. Reboti laadimise alguses süttib punane LED pooleks sekundiks, et kasutajale märku anda, et Rebot käivitub. Logist on näha, et servoprotsess initsialiseeritakse kohe pärast Reboti alglaadimise algust. Esmalt kontrollib alamprotsess kas servodraiver on olemas. Kui seda pole, programmi laadimine seiskub ning välja antakse vastav viga. Servode ja kaamera kalibreerimise operatsioonid käivituvad samal ajal. Tavaliselt lõppeb servode kalibreerimine varem, aga see oleneb konfiguratsioonist. Kui kaamera kalibreerimine peaks kiiremini lõppema, oskab Rebot

siiski oodata kuni ka servode kalibreerimine lõppeb. Kui kõik alglaadimise operatsioonid on lõppenud, süttib uuesti punane LED pooleks sekundiks, mis annab kasutajale märku, et kui ta järgmisel hetkel sihtmärki vaatab, hakkab Rebot teda toitma.

5.8. Reboti peatsükkel

1. Kaamerast loetakse kaader
2. Kaadrist leitakse silm
3. Kontrollitakse kas silmal on x- ja y-koordinaadid
    a. Kui jah, uuendatakse robotkäe staatust
    b. Kui ei, jätkatakse eelmise tsükli staatusega
4. Kontrollitakse kas patsient vaatab sihtmärki
    a. Kui jah, süttib punane LED
    b. Kui ei, kustub punane LED
5. Kontrollitakse kas kasutaja on vajutanud Reboti sulgemiseks vastavat nuppu
    a. Kui jah, programmi peatsükkel katkestatakse
    b. Kui ei, korratakse peatsüklit

Näide Reboti peatsükli logist:

```
---------- ACTION LOG ---------
--> Scooping
--> Finished scooping
--> Extending
--> Finished extending
--> Retracting
--> Finished retracting
--> Scooping
--> Finished scooping
```

5.9. Reboti sulgemine

```
-------- SHUT DOWN LOG --------
--> Received shut down signal
--> Finishing servo processes...
--> Servo processes finished
--> Shutting down camera...
--> Camera shut down finished
--> Rebot will shut down NOW!
```

Reboti sulgemine toimub veaotsimise režiimis klahvivajutusega “c” ja tavalises režiimis SIGINT signaali abil. Mälu haldurites ja servoprotsessis SIGINT signaali ignoreeritakse, aga peaprotsessis püütakse see kinni, et vältida programmi sulgumist koos vealogiga. Tehnilistel põhjustel on erandiks alglaadimise ajal saadud SIGINT. Pärast sulgemise signaali saamist puhastavad servod oma sisemise käskude järjekorra ja pöörlevad nullasendisse. Kaamera suletakse ning kõik veaotsimise režiimis avatud aknad pannakse kinni. Pärast seda lõpetab programm töö. Tootmises olevas robotis peab süsteemi käivitamine ja sulgemine olema lahendatud ühe füüsilise nupuvajutusega, kuid praeguses prototüübis seda funktsionaalsust välja ei arendata.

6. Kokkuvõte

Töö eesmärk oli ehitada söömiseks mõeldud silmaga juhitav robotkäsi, mida kasutatakse puudega inimeste ja nende hooldajate abistamiseks. Töö koosnes kahest osast: riistvara ja tarkvara. Käesolev magistritöö keskendus tarkvarale. Valiti välja sobilik mikroarvuti koos kaameraga. Arvutile installeeriti operatsioonisüsteem, vajalikud draiverid ning spetsiaalselt projekti jaoks arendatud programmikood. Valminud robot nimega R(eye)bot, lühidalt Rebot, täidab oma ülesannet ja seega oli projekt edukas. Programmikood tagab hea platvormi edasisteks arendusteks. Servode arvu saab lihtsasti tõsta kuni kaheksani ning ka kasutajaliidest on üsna lihtne laiendada tänu töökindlale silmatuvastusele. See võimaldab ehitada vajadusel palju keerukama robotkäpa, millega saab näiteks mitmest erinevast kausist süüa. Kood koos installeerimisjuhendiga on avaldatud MIT litsentsi all aadressil https://github.com/mlensment/rebot. Video valminud prototüübi tööst asub aadressil https://youtu.be/gn2T4rIBAUM.

7. Summary

Eye controllable feeding robot for disabled people (Software)

The object of the project was to create an eye controllable feeding robot for the disabled. Code name of the project is R(eye)bot which is Rebot, in short. The work consisted of two parts: software and hardware. This paper focused on software. Processing platform was chosen along with the infra red camera. After installing the operating system and necessary drivers, complex software was developed to control the robotic arm via the headset. The code was written in Python and Bash and was released under MIT licence in Github: https://github.com/mlensment/rebot. Video from the prototype was published in Youtube: https://youtu.be/gn2T4rIBAUM. For further development, the code provides a good platform to easily add more servos and extend the user interface on the headset.

Kasutatud kirjandus

1. (30.04.2015) http://science.howstuffworks.com/prosthetic-limb1.htm
2. (30.04.2015) http://www.amputee-coalition.org/inmotion/nov_dec_07/ history_prosthetics.html
3. (30.04.2015) http://www.healthguideinfo.com/prosthetics-bionics/p9070/
4. (30.04.2015) http://www.engadget.com/2014/12/17/darpa-mind-control-robot-arm/
5. (30.04.2015) http://iopscience.iop.org/1741-2552/12/1/016011/article
6. (30.04.2015) http://spectrum.ieee.org/automaton/biomedical/bionics/dean-kamen-luke-arm-prosthesis- receives-fda-approval
7. (30.04.2015) http://www.christopherreeve.org/site/c.ddJFKRNoFiG/b.5900347/k.3B2/ PARALYSIS_SURVEY_ALMOST_6_Million_affected.htm
8. (2015) Katrin Kibbal https://github.com/mlensment/rebot/raw/master/katrin_msc.pdf
9. (30.04.2015) http://9to5mac.com/2014/09/17/iphone-6-production-by-the-numbers-100-production-lines-200k-workers-540k-phones-a-day/
10. (30.04.2015) http://www.pattersonmedical.com/app.aspx?cmd=getProduct& key=IF_46883
11. (30.04.2015) http://uudised.err.ee/v/majandus/1a12afc7-590c-4345-a986-f3ef6a84b854
12. (30.04.2015) http://www.secom.co.jp/english/myspoon/index.html
13. (30.04.2015) http://www.ece.neu.edu/personal/meleis/icraft.pdf
14. (30.04.2015) http://www.robotshop.com/en/robotshop-m100rak-v2-modular-robotic-

arm-kit-no-electronics.html
15. (30.04.2015) http://www.robotshop.com/en/lynxmotion-ssc-32-servo-controller.html
16. (30.04.2015) http://www.robotshop.com/en/lynxmotion-ssc-32u-usb-servo-controller.html
17. (30.04.2015) http://www.amazon.com/Ableware-745340000-Scooper-Bowl-Suction/dp/B00LSOM8WU
18. (30.04.2015) http://www.maddak.com/the-hydrant-p-27999.html
19. (30.04.2015) http://www.ece.neu.edu/personal/meleis/icraft.pdf
20. (30.04.2015) http://www.radioshack.com/12-6v-ct-3-0a-chassis-mount-transformer-with-leads/ 2731511.html
21. (30.04.2015) http://www.ece.neu.edu/personal/meleis/icraft.pdf
22. (01.05.2015) http://sourceforge.net/projects/gazetrackinglib/
23. (01.05.2015) http://answers.eyetechds.com/questions/52/dark-pupil-v-bright-pupil
24. (01.05.2015) http://www.allaboutvision.com/resources/red-eye-photo.htm
25. (04.05.2015) http://www.pyimagesearch.com/2015/03/30/accessing-the-raspberry-pi-camera-with-opencv-and-python/
26. (04.05.2015) https://www.raspberrypi.org/products/raspberry-pi-2-model-b/
27. (04.05.2015) https://www.raspberrypi.org/forums/
28. (04.05.2015) https://www.elfa.se/elfa3~ee_et/elfa/init.do?item=30-011-591&toc=0&q=raspberry+pi+2
29. (04.05.2014) http://beagleboard.org/BLACK
30. (04.05.2014) http://beagleboard.org/Community/Forums
31. (04.05.2014) http://www.ti.com/devnet/docs/catalog/endequipmentproductfolder.tsp?actionPerformed=productFolder&productId=19580
32. (04.05.2014) https://www.raspberrypi.org/forums/viewtopic.php?p=362714
33. (04.05.2014) http://yeint.ee/elektroonika-1/arenduskomplektid/beaglebone-black-revision-c
34. (04.05.2014) http://www.bananapi.org/p/product.html
35. (04.05.2015) http://www.amazon.com/Raspberry-Pi-like-devepment-Gigabit-ethernet/dp/B00LGXINGS

36. (04.05.2015) https://www.raspberrypi.org/products/camera-module/
37. (04.05.2015) https://www.elfa.se/elfa3~ee_et/elfa/init.do?item=10-709-62&toc=0&q=noir
38. (06.05.2015) https://picamera.readthedocs.org/en/release-1.10/
39. (06.05.2015) https://pypi.python.org/pypi
40. (06.05.2015) http://git-scm.com/
41. (06.05.2015) https://bitbucket.org/
42. (06.05.2015) https://github.com/
43. (06.05.2015) https://picamera.readthedocs.org/en/release-1.10/
44. (06.05.2015) https://pypi.python.org/pypi
45. (03.06.2015) http://groups.engin.umd.umich.edu/CIS/course.des/cis400/python/python.html
46. (06.05.2015) https://www.raspberrypi.org/forums/viewtopic.php?t=57788
47. (07.05.2015) http://opencv.org/
48. (07.05.2015) http://opencv.org/opencv-3-0-alpha.html
49. (07.05.2015) http://simplecv.org/
50. (07.05.2105) http://www.linuxjournal.com/magazine/completely-fair-scheduler
51. (07.05.2015) https://github.com/richardghirst/PiBits/tree/master/ServoBlaster
52. (07.05.2015) https://www.raspberrypi.org/forums/viewtopic.php?f=28&t=99115&start=47
53. (07.05.2015) http://hertaville.com/2014/07/07/rpipwm/
54. (07.05.2015) http://www.dabeaz.com/python/GIL.pdf
55. (07.05.2015) http://ironpython.net/
56. (07.05.2015) http://www.jython.org/
57. (07.05.2015) https://docs.python.org/2/library/multiprocessing.html
58. (08.05.2015) http://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm
59. (08.05.2015) http://homepages.inf.ed.ac.uk/rbf/HIPR2/erode.htm
60. (08.05.2015) http://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm
61. (08.05.2015) http://www-rohan.sdsu.edu/doc/matlab/toolbox/images/morph4.html
62. (08.05.2015) http://homepages.inf.ed.ac.uk/rbf/HIPR2/canny.htm
63. (08.05.2015) http://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm
64. (08.05.2015) http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/

py_imgproc/py_canny/py_canny.html

65. (08.05.2015) http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_contours/py_contours_begin/py_contours_begin.html