TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Reimo Roopärg 163576 IAPM

NÄOTUVASTUSEGA TURVASÜSTEEM RASPBERRY PI ARVUTIL

Magistritöö

Juhendaja: Enn Õunapuu
PhD

Tallinn 2018

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Reimo Roopärg

07.05.2018

Annotatsioon

Käesoleva magistritöö eesmärk on luua turvasüsteemi prototüüp Raspberry Pi arvutile koos näotuvastusega. Lisaks turvasüsteemi põhilisele osale on vajalik implementeerida veebirakendus, mis võimaldaks kasutajal vaadata süsteemi salvestatud videoid.

Töö käigus analüüsitakse erinevaid võimalikke variante püstitatud nõuete realiseerimiseks. Samuti tuleb tähelepanu pöörata süsteemi üleüldisele jõudlusele ning kiirusele. Tuvastamiseks valitud raamistiku tulemused sõltuvad paljuski erinevatest parameetritest ning koefitsientidest, mida töö käigus uuritakse ning mille tulemusena valitakse välja sobivaimad väärtused tagamaks piisav jõudlus ning täpsus.

Töö tulemusena valmis Raspberry Pi arvutil prototüüp, mis suudab fikseerida liikumist, tuvastada nägusid, sisestada süsteemi uusi inimesi, salvestada videona märgatud liikumist ning vaadata videoid läbi veebirakenduse.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 60 leheküljel, 8 peatükki, 19 joonist, 6 tabelit.

Abstract

Security system with face recognition on Raspberry Pi computer

The aim of this thesis is to develop security system prototype on Raspberry Pi computer with face recognition. Additionally, to main part it is needed to implement web interface where user could watch saved security footage.

Different options are analysed for realizations to meet requirements. Also need to pay attention for systems overall performance and speed. Selected library for detection depends more or less on different parameter and coefficients which are analysed during thesis and select most suitable values to ensure sufficient performance and accuracy.

The result of this work is prototype on Raspberry Pi computer which can detect movement, recognize faces, add need persons to system, save spotted movement as videos and look these videos through web interface.

The thesis is in Estonian and contains 60 pages of text, 8 chapters, 19 figures, 6 tables.

Lühendite ja mõistete sõnastik

LBP	*Local Binary Patterns*, Lokaalsed Binaarmustrid
API	*Application programming interface*, rakendusliides
LED	*Light-emitting diode*, valgust kiirgav diood
2FA	*Two Factor Authentication*, Kahetasemeline autentimine
SOC	*System-On-Chip*, Süsteemikiip
URL	*Uniform Resource Locator*, universaalne ressursilokaator
ORM	*Object-relational mapping*, Objekt-relatsiooniline vastavus
HOG	*Histogram of Oriented Gradients*, Orienteeritud gradientide histogramm
SVM	*Support vector machine*, Tugivektormasin
SSH	*Secure Shell*, Turvakest
QR	*Quick Response (code)*, Ruutkood

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Turvasüsteemid ohutuse ja vara kaitsmisel on olnud kasutusel juba aastakümneid. Selgesti näha olev informatsioon turvatud kodu või hoone kohta vähendab juba iseenesest sissemurdmiste arvu, sõltumata konkreetset lahendusest. Artiklis [1] on välja toodud, et kodudesse, kus puudub turvasüsteem, murtakse sisse 300% suurema tõenäosusega kui nendesse, kuhu on paigaldatud turvasüsteem.

Tänapäeva turvasüsteemid on jõudsalt arenemas nii täpsuselt, kiiruselt kui ka tehnoloogialt. Antud valdkonna lahendused liiguvad aina enam intelligentsete ning iseõppivate süsteemide poole. Jätkuvalt on laialdaselt kasutuses erinevaid lahendusi, kus paigaldatakse andurid ustele või akendele ning alarmi alla pandud süsteemi puhul nende avamine annab häire. Samuti on kasutuses mahu- ning liikumisandurite süsteeme, mis samuti annavad teatud olukordades märku liikumisest, mida ei tohiks teatud hetkedel olla. Sellised süsteemid on olnud pikalt kasutuses ning töötavad suhteliselt hästi, kuid neid süsteeme on võimalik lihtsasti välja lülitada või mõjutada. Uuemate lahenduste puhul kasutatakse erinevaid kaameraid, mis suudavad video või pildi pealt tuvastada liikumist, inimesi, autosid ja paljusid muid erinevaid objekte. Taolised süsteemid on muutunud viimasel ajal populaarsemaks ning leidnud laiemat kasutust ka inimeste kodus. Lisaks on antud süsteemid enamasti ühendatud internetti, mille puhul on süsteemi omanikule võimalik saata koheselt informatsiooni koos pildiga ning sellest võib hilisemalt kasu olla kurjategija tuvastamisel. Salvestavad kaamerad turvalisuse eesmärgil on kasutusel ka juba varasemast, kuid kuna nende informatsioon salvestati lokaalsele kettale, võis see kaduda koos kurjategijaga ning tõestusmaterjal varga kohta oli kadunud.

Näotuvastusel põhinevad süsteemid võivad küll mõnes olukorras toimida efektiivsemalt kui sensoritel põhinevad turvasüsteemid, kuid nende üheks miinuseks võib olla hind. Suuremate elamute puhul võib selliseid seadmeid olla vaja mitmeid, mis omakorda tähendab suuremat väljaminekut. Samas Raspberry Pi 3 mudel B maksab alla 40€ [2] ning odavamad veebikaamerad sõltuvalt kvaliteedist 10-20€ [3] .

Lisaks leidub ka turvasüsteeme, kus tuvastamiseks kasutatakse pöördumist mõne välise API poole, kus tuvastatakse ära pildil olevad objektid ning saadetakse tagasi informatsioon. Selline lahendus ei pruugi sobida paljudele inimestele seetõttu, et nende suhteliselt privaatset informatsiooni kodusest olukorrast jagatakse mõnele kolmandale osapoolele.

1.1 Taust

Turvalisuse alla saab ühest küljest liigitada erinevaid võimalikke õnnetust põhjustavaid tegureid nagu näiteks tuli ja gaas. Teisest küljest võib tõlgendada turvalisust kui ennetavaid meetmeid sissemurdmiste vastu. Käesoleva töö raames pakuvad huvi erinevad süsteemid varaste ning muude sissepääsu õigust mitte omavate inimeste tuvastamine ning sellest teavitamist. Elu- ja tööruumides hoiustatakse palju erineva väärtusega esemeid, mida tuleb valvata turvasüsteemidega. Lisaks esemetele on inimestele tähtis nende enda turvalisus ning turvatunne. Paljuski sellest tagatakse erinevate turvasüsteemidega. Praegusel hetkel on turult võimalik leida väga erinevaid viise oma elukoha või mõne muu eramu julgeoleku tagamiseks. Süsteeme on võimalik leida erinevatest hinnaklassidest, turvalisuse tagamise viisidest kui ka lahendustest. Suures plaanis jagunevad turvasüsteemid kaheks – sensoritel põhinevad turvasüsteemid või objektide tuvastusel põhinevad turvasüsteemid.

1.1.1 Sensoritel põhinevad turvasüsteemid

Erinevaid sensoreid kasutades on koduseid turvasüsteeme loodud üle 30 aasta. Esimesi liikumissensoreid tutvustati 1970ndatel ning infrapuna kaameraid 1980ndatel aastatel [4]. Infrapuna kaamerad on siiani tähtsal kohal turvalisuse tagamiseks nii kodudes kui firmades. Antud andurid ei tuvasta otseselt liikumist, vaid reageerivad soojuse muutumisele, mis registreeritakse kui elusolendi ilmumist kaamera vaatevälja. Lisaks infrapuna anduritele on veel olemas heli muutusi tuvastavad andurid, uste ja akende avamist tuvastavad sensorid, klaasi purunemist registreerivad andurid jne. Taolised lahendused pakuvad küll kõrget turvalisuse taset, kuid kuna need on kasutusel olnud juba pikka aega, on leitud ka nende juures erinevaid puudusi ning viise, kuidas vältida alarmi käivitumist. Veebiartiklis [5] on välja toodud, et kõik juhtmevabalt töötavad süsteemid kasutavad omavaheliseks suhtlemiseks raadiosageduse signaale. Antud artiklis avastasid teadlased, et süsteemidel ebaõnnestub krüpteerimine ning autentimine, mille tõttu on

võimalik infovahetusele vahele segada. Halvemal juhul on võimalik raadiosignaali segada ning takistada info jõudmist süsteemi teistesse komponentidesse.

Samuti on taolised süsteemid tavaliselt ühendatud ühe keskse kontrolleriga. See tähendab aga seda, et kui antud kontroller lakkab töötamast kas rikke või osavate sissemurdjate tõttu, ei ole kasu ka ühestki eraldiseisvast sensorist, mis ukse või akna avamisel peaks tööle hakkama. Häire käivitumisel võivad kiiremad murdvargad olla läinud ennem, kui korrakaitsjad kohale jõuavad ning informatsioon varaste välimuse kohta ei ole kuskile talletatud.

1.1.2 Näotuvastusel põhinevad turvasüsteemid

Erinevaid tooteid kodu turvalisuse tagamiseks kasutades näotuvastust on tehtud mitmeid. Paljud neist on arendatud selliselt, et kasutajal on neid väga lihtne kasutada. Vaja on seade ühendada vooluvõrku, seada üles interneti ühendus ning sisestada pildid sissepääsu õigust omavatest inimestest. Näotuvastust kasutatavate seadmete üheks eeliseks võib pidada seda, et lõppkasutajal on võimalik saada erinevat informatsiooni oma kodus toimuvast (näiteks otseülekanne, vaadata järgi salvestusi) sõltuvalt seadme realisatsioonist.

Netatmol [6] on turvasüsteeme nii sise- kui välistingimuste jaoks. Välise kaamera puhul on võimalik valida välja tsoonid, kus liikumist märgates saadetakse kohe teavitus (näiteks auto ümber) ning vaadata ka otseülekannet läbi nutiseadme. Antud seade kasutab 4 megapikslist kaamerat, millel on 100 kraadine vaateväli ning 1920x1080 resolutsioon. Selline kaamera peaks edastama piisavalt selget pilti, millelt tuvastada ära hiljem inimese nägu. Sisetingimustesse mõeldud kaamera põhieesmärgiks on tuvastada liikuvaid inimesi ning sissetungijate korral saata informatsioon koos pildi ja videoga lõppkasutaja nutiseadmesse. Lisaks, kuna suudetakse tuvastada erinevaid elusolendeid, siis on võimalik välja lülitada loomade registreerimine ning seeläbi välistada nendega seotud liikumisest tulenevad teated.

BuddyGuard [7] on turvasüsteem, kus kasutatakse turvalisuse tagamiseks tehisintellekti. Riistvara poolelt on kasutusel:

- 1080p kvaliteediga lainurgaga kaamera koos infrapuna LED öövaatlusega
- Liikumissensorid

- Kahepoolne heli koos 80dB sireeniga
- 2 tuumaline 1.2GHz ARM Cortex 9 protsessor
- WiFi ühendus
- AES256 krüpteeringuga video ning audioedastus

Tarkvara poolelt täpsemat informatsiooni tuvastamise kohta ei leidnud, kuna seda võib pidada antud toote puhul ärisaladuseks. Lisaks on arendatud mobiilirakendus, mille kaudu lõppkasutaja saab vaadata oma kodus toimuvat. Rakendusest saab turvasüsteemi sisse ning välja lülitada, näha erinevaid seisundeid - kõik on turvaline, lubatud külaline viibib majas (näiteks koristaja) või ohtlik situatsioon (keegi võõras on sisenenud). Võimalik on ka määrata erineva tasemega luba omavaid inimesi nagu näiteks sõbrad, residendid, külalised. Antud lahendus on väga hea, kuna olenevalt olukorrast võib hoones viibida inimesi, kes pole otseselt pereliikmed, kuid kelle puhul häiret ei ole vaja käivitada.

Floodlight Cam [8] on firma Ring poolt arendatud turvakaamera. Mainitud kaamera suudab tuvastada nägusid ning objekte, reageerida liikumisele, edastada mobiilirakendusse reaalajas videot ning salvestada 1080p (täis HD) resolutsiooniga videot. Riistvara poolelt on ära märgitud toode kirjelduses:

- WiFi ühendus
- Kahepoolne heli
- 110dB sireen
- Kaks 3000K võimsust valgustust
- Veekindel ümbris

Floodlight Cam üks seade maksab nende kodulehelt ostes 299€. Kodulehel välja toodud privaatsuse seadetest võib välja lugeda, et seadme töö tagamiseks salvestatakse kogu heli ning video. Videote salvestamiseks ning hilisemaks vaatamiseks tuleb maksta kuu või aasta põhist tasu.

Tabel 1 Turvasüsteemide võrdlus

	Netatmo	BuddyGuard	Floodlight Cam
Näotuvastus	✓	✓	✓
Informatsiooni jagamine seadmest välja	×	×	✓
Lisatasud	×	×	×/ ✓
Hind	199.99€	367€	299€

1.1.3 Puudused

Näotuvastusele põhinevatel turvasüsteemidel on vaatamata eelistele ka erinevaid puudusid. Artiklis [9] on välja toodud 5 järgnevat põhilist probleemi targa kodu turvalahendustes:

- Rünnakud füüsilisele kihile
    - Segajad – näiteks raadiosignaalide ja interneti ühenduse häirimine
    - Võltsimine – viiruse sokutamine süsteemi, salavõtmete informatsiooni kätte saamine
- Rünnakud andmekihtidele
    - Üle võrgu liigutatavate andmeblokkide lugemine ja maha kirjutamine
- Rünnakud võrgustikule (*Networking*)
- Rünnakud transpordikihile
- Rünnakud rakenduse kihile

Lisaks eelnevalt väljatoodud probleemidele on võimalusi ka pääseda süsteemist läbi tänu valepositiivsete tulemuste loomise. Näiteks kasutades süsteemile ligipääsu omava isiku pilti või maski ning seeläbi ära petta süsteemil olev näotuvastus programm.

Turvasüsteemid on jätkuvalt väga teemakohased ning laialdaselt kasutuses nii korterites, eramajades, tööstushoonetes, firmades jne. Erinevatel turvasüsteemidel on omad plussid ja miinused. Ideaalset turvasüsteemi ei ole olemas, kuna enamusest neile on leitud viise möödapääsemiseks ja/või süsteemide normaalse töörütmi segamist.

1.2 Ülesande püstitus

Magistritöö eesmärgiks arendada välja prototüüp, mis oleks kasutajale odavam ning turvasüsteem oleks paigutatud lokaalselt ning võimalikult palju tuvastamisest toimuks seadme peal. Tegevused, mille teostamine ei ole võimalik seadmes endas, võib kaaluda pöördumist mõne välise teenuse poole. Süsteemi arendamisel tuleks tähelepanu pöörata järgnevatele nõuetele:

- Süsteemile kaugligipääs ei tohiks olla triviaalne, kuna seadmes sisaldub väga isiklikku informatsiooni kasutajale ning see ei tohiks sattuda valedesse kätesse.
- Süsteem peab olema kindel tuvastatud näo õigsuses vältimaks liiga vähese tõenäosusega tuvastatud nägude aktsepteerimist.
- Süsteemi salvestatud turvavideosid peaks lõppkasutajal olema võimalik näha (peale sisselogimist) veebilehel.
- Süsteemi riist- ja tarkvara valimisel tuleb lõplik hind hoida võimalikult madalana.

1.3 Metoodika

Prototüübi arendamisel tuleks esialgu tutvuda sarnaste toodetega ning analüüsida erinevaid võimalusi nii riistvara kui tarkvara tasemel. Süsteemile tuleks püstitada esialgsed minimaalsed nõuded, mis võiksid olla sarnased teiste turul olevate toodetega. Vastavalt nõuetest ning analüüsitud võimalustest saab valida vajalikud komponendid süsteemi arendamiseks. Komponentide mittesobimise korral teostada uuesti analüüs ning leida alternatiivne võimalus antud probleemi lahendamiseks. Keerukamate otsuste

tegemiseks kasutan SAATY AHP mudelit, mille abil leida sobivaim (alternatiivne) lahendus omaduste omavahelisel võrdlemisel.

Planeeritavate ideede ning töö eri osade valmimiseks kasutada mõnda visuaalset vahendit saamaks paremat ülevaadet hetke olukorrast. Agiilse tarkvaraarenduse üheks populaaremaks vahendiks tööde jälgimiseks on Atlassiani toode Jira [10]. Jira pakub väga laialdaselt erinevaid võimalusi planeerimiseks ning jälgimiseks tööde hetke olukorrast, töölaua kujundamiseks, raportite genereerimiseks, töö voo teostamiseks jne. Jirat on võimalik kasutada nii pilveteenusena kui ka enda serveris. Mõlemal juhul on võimalik kasutada vastavalt 7 või 30 päeva tasuta versiooni, kuid peale seda tuleb maksta kasutusõiguse eest. Üheks alternatiiviks võib kasutada Trello [11] tasuta versiooni. Trello ei pruugi pakkuda nii laialdaselt erinevaid võimalusi kui Jira, kuid sisaldab siiski piisavalt vahendeid nagu näiteks tööde jälgimist, oma töölaua kujundamist, sobivate tulpade loomist vastavalt vajadusele jne.

Masinõppe programmide tööd juhivad sageli erinevad sisendparameetrid ning koefitsiendid. Erinevate parameetrite mõju võib kujuneda oluliseks osaks kogu süsteemi arendamisel. Väärtuste valikul tuleb tähelepanu pöörata süsteemi täpsusele, kuid samal ajal ei tohi ära unustada mõju jõudlusele ning kiirusele. Parameetri väärtuse muutmisel võib süsteemi täpsus paraneda minimaalselt, kuid samal ajal vähendada kiirust kordades.

Prototüübi tulemusi hinnatakse vastavalt püstitatud nõuetele ning eesmärkidele. Tulemuste valideerimiseks oleks hea valminud prototüüpi võrrelda mõne turul oleva sarnase tootega erinevates aspektides (näiteks kiirus, täpsus, hind).

1.3.1 SAATY

Järgnev kirjeldus SAATY mudelist põhineb artikli [12] põhjal. Analüütiline hierarhiline protsess ehk AHP (*Analytic Hierarchy Process*) on üldine mõõtmisteooria, mis arendati T.L.Saaty poolt 1971-1975 Whartoni koolis. AHP mudelit saab kasutada suhete skaala tuletamiseks nii diskreetsete kui pidevalt seotud võrdluste jaoks. Neid võrdlusi võib võtta reaalsetest mõõtetulemustest kui ka fundamentaalsest skaalast, mis peegeldab eelistuste ja tunnete relatiivset tugevust. Üldiselt jaguneb mõne ühiskonna probleemi hierarhiline mudel järgmiselt:

- Eesmärk (*focus*) – millele otsitakse lahendus

- Kriteeriumid (*criteria*) – eesmärgi olulised omadused
- Alamkriteeriumid (*subcriteria*) – lisakriteeriumid eelnevatele kriteeriumitele
- Alternatiivid (*alternatives*) – võimalikud lahendused, mille hulgast teha valik

Paari kaupa võrdlemine on üks põhilisi aluseid AHP mudeli kasutamiseks. Paaride omavahelises võrdluses tähtsuse määramiseks kasutatakse tabelis 2 välja toodud.

Tabel 2 Saaty skaala relatiivseks tähistuseks

Tähtsuste tugevus absoluutskaalal	**Definitsioon**	**Selgitus**
1	Võrdselt tähtsad	Kaks tegevust annavad võrdse panuse objektile
3	Ühe mõõdukas tähtsus üle teise	Kogemused ja otsused toetavad ühte tegevust teisest rohkem
5	Tugev tähtsus	Kogemused ja otsused toetavad ühte tegevust teisest rohkem
7	Väga tugev tähtsus	Üks tegevus on tugevalt soositud ning praktikas tõestatud
9	Ekstreemne tähtsus	Kõige tugevamad tõendid kinnitavad, et üks tegevus on teisest parem
2,4,6,8	Vaheväärtused kahe kõrvuti asetseva otsuse vahel	Kompromissi tegemiseks
Pöördväärtused	Kui tegevusele i on omastatud eelnevalt mingi	

	väärtus x võrreldes tegevusega j, siis j saab i pöördväärtuse ehk 1/x	
Ratsionaalarvud (1.1,1.2,...)	Omadused on väga võrdse kaaluga	Täpsemaks määramiseks, kui omadusi ei ole võimalik määrata tavalisel skaalal

Antud klassifikatsiooni abil tuleb ära hinnata kõik kriteeriumid paarikaupa omavahel, mille tulemusena on võimalik leida kõige sobivaim alternatiiv antud probleemi lahenduseks.

1.4 Ülevaade tööst

Töö järgnevas peatükis antakse lühike ülevaade seotud töödest ning vaadeldakse Raspberry Pi realisatsioone nii turvasüsteemidel kui veebilehtede majutuses. Samuti kirjeldatakse kahetasemelist autentimist ning üle võrgu saadetava videopildi edastuse ohte ning võimalusi nende vältimiseks.

Töö kolmandas peatükis püstitatakse süsteemile esialgsed nõuded, millele arenduse käigus tähelepanu pöörata ning mille vastu lõpus valminud prototüüpi on võimalik hinnata.

Neljandas peatükis võrreldakse erinevaid tuvastustarkvarasid erinevate parameetrite alusel ning valitakse välja antud olukorra lahenduseks sobivaim lahendus. Lisaks analüüsitakse erinevaid veebiraamistikke (*web frameworks*), mida kasutada kasutajaliidese arendamiseks.

Töö viiendas peatükis kirjeldatakse täpsemalt objektide tuvastamist, sellega seotud põhilisi funktsioone ning nende tööd mõjutavate parameetrite tähendust ning mõju tulemustele.

Töö lõpus vaadeldakse süsteemi tulemusi realiseeritud prototüübi peal, mis on seotud isikute tuvastamisega ning jõudlusega. Viimases osas antakse erinevaid võimalusi antud magistritöö edasiarenduseks.

2 Seotud tööd

Käesolevas töös planeeritud turvasüsteemi lahendusi on ka varasemalt teostatud erinevate lahendustega. Samuti on realiseeritud näotuvastusi ning veebilehtede majutust Raspberry Pi arvutil. Järgnev peatükk vaatleb erinevaid teostatud lahendusi ning nendega seotud tulemusi ja järeldusi.

2.1 Turvasüsteemide realisatsioonid Raspberry PI peal

Raspberry Pi peale on loodud erinevaid näotuvastusel põhinevaid süsteeme. Sellist lähenemist on proovitud ka kasutada turvasüsteemi kontekstis. Artiklis [13] on toodud välja üks lahendusviis Raspberry Pi peale kasutades Open CV ning *Haar Cascade* klassifikaatorit. Antud projektis tuvastatakse liikumist, proovitakse pildilt tuvastada nägusid kasutades „haarcascade_frontalface_default.xml“ klassifikaatorit ning saadetakse SMS ja e-mail kasutajale. Kokkuvõttes on autorid välja toonud, et Raspberry Pi on kuluefektiivne järelevalve mehhanism.

Artiklis [14] on kasutatud Raspberry Pi ning Open CV teeki tagamaks riietusruumi turvalisust. Salvestusmahu vähendamiseks filmib kaamera ainult siis, kui eelmine kaader erineb praegusest kaadrist, ehk on märgatud mingit liikumist. Video töötluseks kasutatakse teeki Open CV ning programmeerimiskeeleks Pythonit. Liikumist märgates salvestatakse maha video kaader ning saadetakse teavitus e-mailile. Objektide tuvastamiseks kasutatakse antud töös Lokaalseid Binaarmustreid ehk LBP. Kokkuvõttes on autorid välja toonud, et sellised intelligentsed süsteemid võimaldavad vähendada kulusid, kuna salvestatud videote maht on väiksem ning kulub vähem käsitööd, et töötada läbi videoid. Samuti on välja toodud, et vabavaralise Open CV kasutamine Raspberry Pi peal on efektiivne meetod ning kuna arvutil ei ole ühendust internetiga, on see ka palju turvalisem.

2.2 Raspberry Pi arvutil veebilehtede majutamine

Raspberry Pi on jõudluselt ning võimalustelt sobilik erinevate veebilehtede majutuseks. Artiklis [15] on Raspberry Pi peale arendatud kodu automatiseerimise süsteem koos veebilehega haldamaks erinevaid komponente kodus (soojus, niiskus, valgus jne). Veebilehe tarkvara arenduseks kasutasid autorid vabavaralist Django raamistikku. Lisaks tasuta kasutamisele on Django eeliseks artiklis toodud stabiilsus ning võimalus vältida mitmeid levinumaid turvalisuse riske. Raamistikku on sisse ehitatud kasutajate autentimise süsteem, mis tagab turvalise kasutajanimede ja paroolide hoidmise.

2.3 Kahetasemeline autentimine

Kahetasemeline autentimine ehk 2FA on täiendav turvalisuse tase lisaks kasutajatunnuse ja parooli kombinatsioonile [4]. Taoline autentimine tagab parema turvalisuse, kuna sobivaid koode sisenemiseks muudetakse mingi kindla intervalli tagant. Samuti kasutavad inimesed tihti liiga lihtsasti ära arvatavaid ning erinevates kohtades samu paroole [16].

Turvasüsteemides korrektne ning modifitseerimata video edastamine on üks tähtsamaid komponente kogu süsteemis. Artiklis [17] on autorid välja toonud kaks põhilist viisi, kuidas on võimalik turvasüsteemide kaugjuhitavaid videoülekandeid kurjalt ära kasutada. Esiteks on võimalik algset videot modifitseerida või täielikult üle kirjutada. Teiseks võib mitte autoriseeritud inimene näha videot, mida ta ei tohiks näha. Esimene oht võib tõsiselt seada ohtu seadme turvalisuse. Kirjeldatud probleemide lahendamiseks on autorid käinud välja samuti kaks lahenduskäiku mõlema ohu kohta. Tuleb arendada multimeedia autentimine, mis verifitseerib, kas saatja poolt teele läinud video on sama, mis jõudis vastuvõtjale. Teiseks tuleb arendada seadme autentimine, mis kontrollib saaja õigust antud videot mängida.

2.4 Järeldused

Raspberry Pi koos Open CV'ga võiks võimaldada realiseerida reaalset kodust turvasüsteemi. Salvestusmahu vähendamiseks, kuid samas olulise informatsiooni talletamiseks, võiks kasutada kaadrite omavahelist võrdlemist liikumise tuvastamiseks ning video salvestamiseks. Lisaks ruumi kokkuhoiule oleks võimalik optimeerida

süsteemi tööd ning rakendada rohkem ressurssi nõudvaid protsesse ainult siis, kui on märgatud liikumist ehk potentsiaalne ohuolukord. Jõudluse poole pealt peaks Raspberry Pi arvutil olema piisavalt võimas, et tegeleda näotuvastusega ning samuti lõppkasutajale kuvatava veebirakenduse majutusega. Juhtmevaba kaamera info edastuse puhul tuleb suurt tähelepanu pöörata turvalisele andmeedastusele või selle vältimiseks edastada infot juhtmega.

3 Nõuded

Antud tööd realiseeritava prototüübi üheks osaks on esialgsete nõuete olemasolu. Süsteemi kaheks kõige olulisemaks nõudeks on hind ning võimalikult suure osa tuvastuse teostamine lokaalselt. Nõudeid, mille täitmist ei ole võimalik teostada lokaalselt tuleb vaadelda antud nõude tähtsust ning võimalusel vältida väliste teenuste kasutamist.

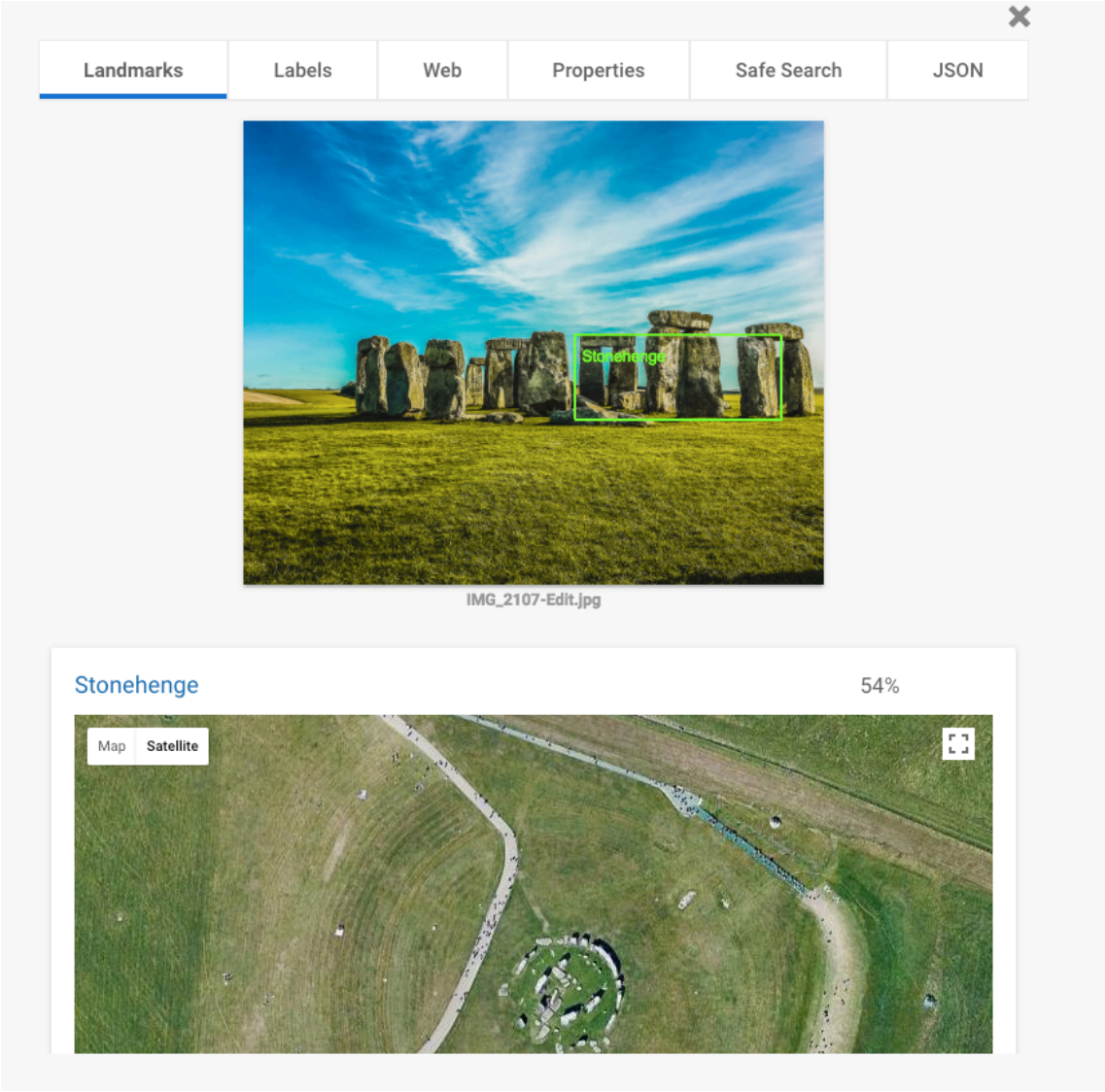
Järgnevalt toon välja mõned esialgsed nõuded süsteemi arendamisel:

- Süsteemi komponendid (näiteks treeningandmete koostamine, treenimine, tuvastamine) peavad olema üksteisest sõltumatud võimaldamaks erinevate osade kasutamist hilisemas arenduses.
- Süsteemi failidele ligi pääsemiseks (SSH) peab autentimiseks kasutama kahetasemelist autentimist.
- Süsteemi taaskäivitamisel peab automaatselt tööle minema veebirakendus ning turvasüsteem.
- Töö optimeerimiseks peab süsteem suutma tuvastada liikumist kaamera vaateväljas.
- Liikumise avastamisel peab süsteem töötlema pilti täpsemalt.
- Video pealt näo leidmisel peab süsteem tuvastama iga nägu eraldi ning leidma, kas antud inimene on süsteemi sisestatud või mitte.
- Hilisemaks tõendamiseks ning analüüsiks peab süsteem salvestama MP4 formaadis ning H.264 kodeeringuga video.
- Video mahu vähendamiseks peab süsteem alustama video salvestus siis, kui on märgatud liikumist
- Peale 10 minuti möödumist viimasest liikumisest peab süsteem sulgema videofaili tekitamise, kui toimub video salvestamine.
- Süsteemi salvestatud videoid peab kasutajal olema võimalik näha läbi mõne veebirakenduse.

Lähtudes süsteemile püstitatud algsetest nõuetest on võimalik riistvara ja tarkvara valikul langetada mõningaid otsuseid.

4 Tarkvara valik

Magistritöös realiseeritava prototüübi üks eesmärkidest on hoida lõppkasutajale kulud võimalikult väiksena. Nõuetele sobiliku tarkvara leidmisel on lai valik erinevaid süsteeme, teeke, rakendusliideseid jne. Erinevate objektide tuvastamiseks on tehtud mitmeid API'si, nendest tuntumad on Google Cloud Vision API [18] ning Amazon Rekognition API (edaspidi AWS) [19]. Väljatoodud rakendusliidesed toimivad väga hästi. Need kuuluvad suurtele firmadele, kellel on võimalusi arendada kvaliteetset tarkvara, kasutada võimast riistvara ning tänu laialdasele kasutusele võimalusi pidevalt areneda ning tuvastada objekte täpsemalt. Google Cloud Visionit on võimalik katsetada läbi veebibrauseri. Lisaks pildil olevate objektide tuvastamisele suudab programm tuvastada näiteks hoone asukohta kaardil ja nimetust [Joonis 1] ning muud lisa informatsiooni, mille jaoks on vaja hulgaliselt andmeid.



Joonis 1 Google Cloud vision - Stonehenge

Vaatamata sellele, et taolised rakendusliidesed on ülimalt võimsad, on nende suure mahuliseks kasutamiseks vaja maksta kuupõhist tasu. Nende teenuste kasutamise hind sõltub erinevatest teguritest. Hinna määrab näiteks kuus tuvastatavate piltide arv, tuvastamise võimalused jne.

Amazon Rekognition kodulehelt võetud näite puhul Ida USA regiooni hindade järgi tuleks 110 000 minuti video analüüsimise eest maksta 11 000 dollarit. Informatsiooni edastamist töötlemiseks on võimalik optimeerida, näiteks pöörduda rakendusliidese poole ainult juhul, kui on toimunud kaamera vaateväljas mingeid muutusi. See toimiks juhul, kui lõppkasutaja kasutaks süsteemi suhteliselt vähekäidavas kohas ning ei toimuks palju muutusi kaamerapildis. See piiraks süsteemi võimalusi ning võib põhjustada ootamatuid kulutusi süsteemi kasutusel.

4.1 Tuvastustarkvarade analüüs

Erinevaid võimalusi nii avatud lähtekoodiga (*open-source*) kui ka (kuu)tasuliste süsteemide seast objektide tuvastamiseks on väga palju erinevaid. Kõigil neil on oma eelised ja puudused. Sõltuvalt eelnevalt püstitatud nõuetest tuleb valida teek või teegid, mille abil oleks võimalik täita püstitatud eesmärgid.

Järgnevalt võrdlen artiklis [20] välja toodud 3 erinevat võimalust nägude tuvastamise teostamiseks ja OpenFace [21] teeki ning nende sobivust süsteemile püstitatud nõuetele.

Tabel 3 Tuvastustarkvarade võrdlus

	Nägude leidmine	**Nägude tuvastamine pildilt**	**Näo tuvastamine videolt**	**Lokaalne tuvastamine**	**Tasuline**	**Järjepidev edasiarendus**
OpenCV	✓	✓	✓	✓	×	✓ (viimane uuendus 2018)
Amazon rekognition	✓	✓	×	×	✓	✓
Google Cloud vision	✓	×	×	×	✓	✓
OpenFace	✓	✓	✓	✓	×	× (viimane uuendus 2016)

Saaty meetodi abil leidsin sobivaima teegi lähtudes tabelis 3 kirjeldatud alternatiividest ning kriteeriumitest. Omaduste omavahelisel võrdlusel osutus kõige prioriteetsemaks lokaalne tuvastamine, millele järgnes nägude leidmine ning tasuta kasutamise võimalus. Parimaks võimaluseks osutus OpenCV teek. Lisaks kriteeriumite sobimisele on OpenCV hea dokumentatsiooniga, paljude näidetega, aktiivse kommuuniga ning laialdaselt kasutusel olev teek.

4.2 Veebiliidese valiku analüüs

Lisaks tuvastamisele peab lõppkasutajal olema võimalik läbi veebiliidese vaadata süsteemi salvestatud videoid ning sisestada uusi inimesi süsteemi. Veebirakenduse arenduseks on mõistlik valida raamistik, mida saab kirjutada programmeerimiskeeles Python. Esiteks on Raspberry Pi peal olev operatsioonisüsteem Raspbian Linuxi põhine operatsioonisüsteem, kus on eelnevalt installeeritud Python. Teiseks on turvasüsteem arendatud kasutades OpenCV teeki programmeerimiskeeles Python. Erinevaid teeke antud ülesande lahendamiseks on mitmeid. Järgnevalt toon välja 3 populaarsemat Pythoni rakendusraamistikku veebiartiklist [22] ning nende positiivsed ja negatiivsed küljed.

4.2.1 Django

Django on vaba-varaline täispakett raamistik, sisaldades erinevaid funktsioone (*feature*) nagu näiteks autentimine, URL suunamine, ORM ja andmebaasiskeemide migreerimine. Django kogukond on võrreldes teiste raamistikega kõige suurem. Raamistikku on sisse ehitatud funktsionaalsus genereerimaks uusi projekte koos korrektse struktuuri ja algseadetega. Projekt struktureeritakse erinevateks rakendusteks, mis tõttu on võimalik kasutada mitmeid rakendusi ühes projektis. Flask ja Pyramid eeldavad, et üks projekt sisaldab ühte rakendust erinevate vaadete ja mudelitega. Võrreldes teise kahe raamistikuga peab Django olema täpsemalt struktureeritud korrektseks töötamiseks.

4.2.2 Flask

Flask on „mikroraamistik“ mis on peamiselt mõeldud väikeste ja lihtsate nõuetega rakenduse arenduseks. Antud raamistiku kõige minimaalsema töötava rakenduse loomiseks on vaja alla 10 rea koodi. Seetõttu ei ole loodud projekti genereerimiseks eraldi käsku, kuna projekti üles seadmine on primitiivne.

4.2.3 Pyramid

Pyramid on arhitektuurilt ja kasutuse eesmärgilt Django ning Flaski vahel paiknev raamistik. Pyramid sisaldab vähem eelnevalt installeeritud funktsioone (*feature*) võrreldes Djangoga. Väliste teekide abiga on võimalik laiendada raamistiku võimalusi. Raamistikku on sisse ehitatud sarnaselt Djangole uue projekti automaatne genereerimine, kuid loodud struktuur ei ole oma kasutuselt nii jäik.

4.3 Järeldused

Võimalusi erinevate lahenduste kasuks on mitmeid. Lähtudes probleemist, püstitatud eesmärgist ning erinevate alternatiivide võrdlemisel kriteeriumite alusel osutus tuvastamiseks kõige sobilikumaks avatud lähtekoodiga teek OpenCV. Veebirakenduse loomisel kõige sobilikumaks valikuks osutus rakendusraamistik Django, kuna võimaldab suuremaid ning keerukamaid projekte lihtsamalt arendada ning sisaldab vajalike sissehitatud teeke (autentimine, ümbersuunamine). Flask ning Pyramid võivad tulevikus hakata takistama süsteemi edasi arendamist või võimaldavad seda teha keerukama arhitektuuriga.

5 Objektide tuvastamine

Turvasüsteemi puhul tuleb paljugi pöörata tähelepanu erinevatele tegevustele, mis jäävad kaamera vaatevälja. Sõltuvalt kaamera paigutusest võib pildis toimuda väga palju erinevaid tegevusi ning leida erinevaid liikuvaid objekte. Seetõttu tuleb esialgu määrata ära, mis objekte ning tegevusi üldse tuvastada ning mis järjekorras tegeleda objektide tuvastamisega. See tagab parema tulemuse objektide tuvastamisel ning vähendab liigsete tegevuste tegemise. Probleemi lahendamise struktuurist võib lähtuda inimese käitumisest objektide tuvastamisel, kus tuvastamine jagatakse väiksemateks osadeks ning liigutakse samm sammult keerukamate ning täpsemate objektide identifitseerimise poole [23]. Arvuti võiks oma käitumiselt lähtuda sarnaselt:

- Tuvastada liikumine.
- Vaadata, kas pildil leidub inimene.
- Leida üles inimese nägu.
- Proovida nägu tuvastada.

5.1 Liikumise märkamine

Süsteemi töö optimeerimiseks tuleks fikseerida kaamera vaateväljas erinevaid liikumisi. Liikumise leidmiseks tuleb fikseerida üks kaader, mis oleks süsteemile algolekuks ning võrrelda seda edaspidi kaamerast tuleva hetke pildiga.

5.1.1 Algoleku määramine

Algoleku määramisel võiks valida esimese kaadri, mis süsteem fikseerib ning hakata sellega võrdlema järgnevaid kaadreid. Selline lahendus toimiks ainult sellisel juhul, kui:

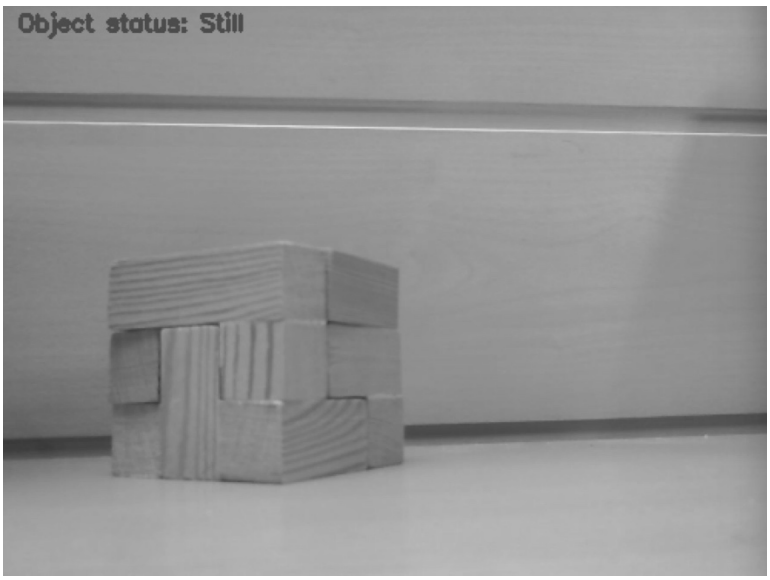
1. Esimese kaadri tegemisel ei oleks ühtegi liikuvat objekti vaateväljas, kuna objekti ära liikumiselt oma kohalt oleks see süsteemile kui erinev pilt ehk märk pidevast liikumisest.
2. Kaamera vaateväljas ei toimuks valguse muutust (näiteks varjude liikumist)

3. Kaamerat algasukohta ei liigutata kunagi, kuna sellisel juhul toimuks süsteemi vaatest kogu aeg liikumine (algne kaader ei vasta hetke sisule).

Eelpool nimetatud stsenaariumid on kõik võimalikud juhtuma igapäevaselt ning seetõttu ei ole sellist lähenemist mõistlik kasutada. Selliste probleemide lahendamiseks peab süsteem algolekut muutma jooksvalt. Süsteemile määratakse uus algolek iga kord, kui 20 järjestikuse kaadri tagajärjel ei ole tuvastatud liikumist. Olukorras, kus siseneb inimene ning seisab 20 kaadrit paigal ning see seatakse uueks algolekuks, siis on eelnev liikumine siiski salvestatud ning samuti iga liigutuse peale registreeritakse see uuesti.

5.1.2 Kaadrite võrdlemine

Liikumise tuvastamiseks piisab, kui võrrelda kahte kaadrit omavahel. Selleks on vaja võtta iga kaader, mis kaamerast tuleb, vähendada tema suurust ja konverteerida hallidesse toonidesse (*grayscale*). Kaamerast tulevad 2 kaadrit ei ole kunagi 100% ühesugused, seepärast tuleb pilti hägustada (*blurring*).

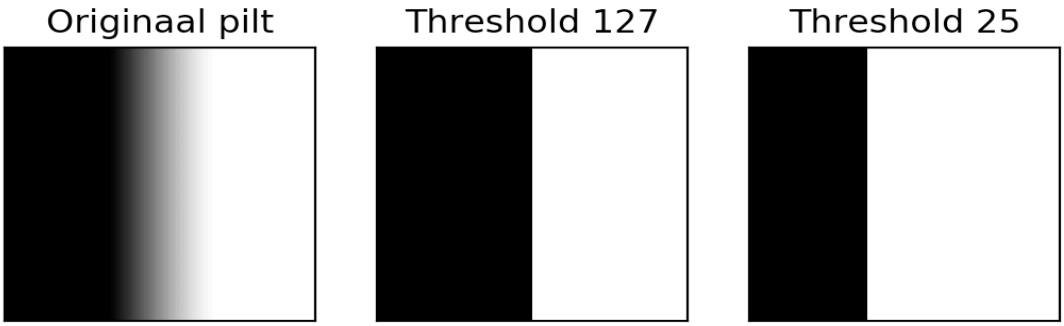


Joonis 2 Liikumise algolek

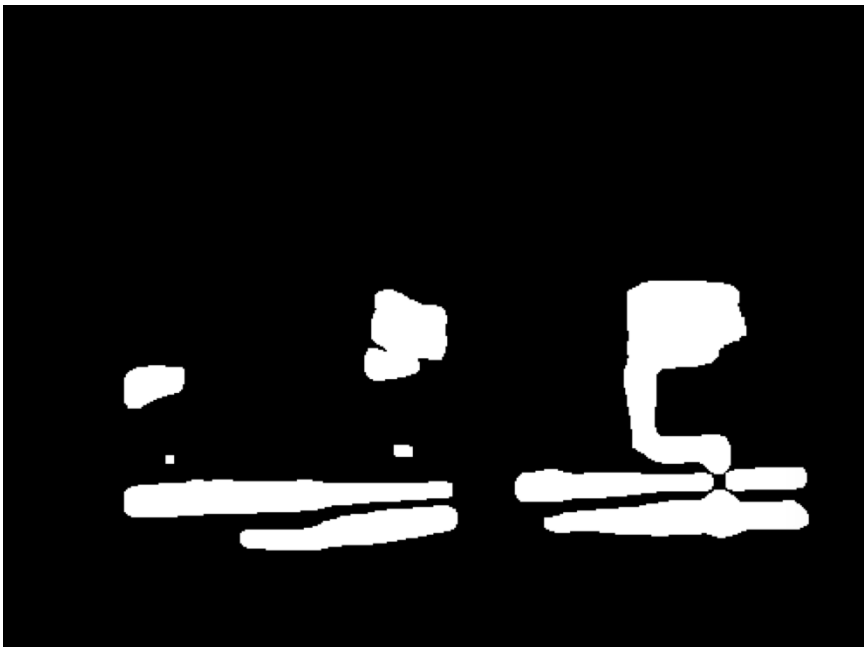
Algolekut ning hetke kaadrit (töödeldud samamoodi nagu algkaader) on võimalik võrrelda omavahel kasutades Open CV funktsiooni `absdiff`. Funktsiooni rakendamise tulemust on võimalik vaadelda järgneval joonisel, kus vasakul on objekt esialgses asukohas ning paremal kaadril on objekti liigutatud kaadri vasakust servast paremasse serva. Muutunud on ainult objekti asukoht, mistõttu on ülejäänud pilt (taust) must.

Joonis 3 OpenCV absdiff

Kontuuride leidmiseks on vaja leitud erinevuste kaadrile rakendada funktsiooni `threshold` [24], mis võrdleb halltoonides pildil piksli väärtusi läve väärtusega. Piksli väärtused, mis on üle läve värvitakse valgeks ning ülejäänud mustaks. Tulemuseks on konkreetsete joontega ära piiritletud pilt.

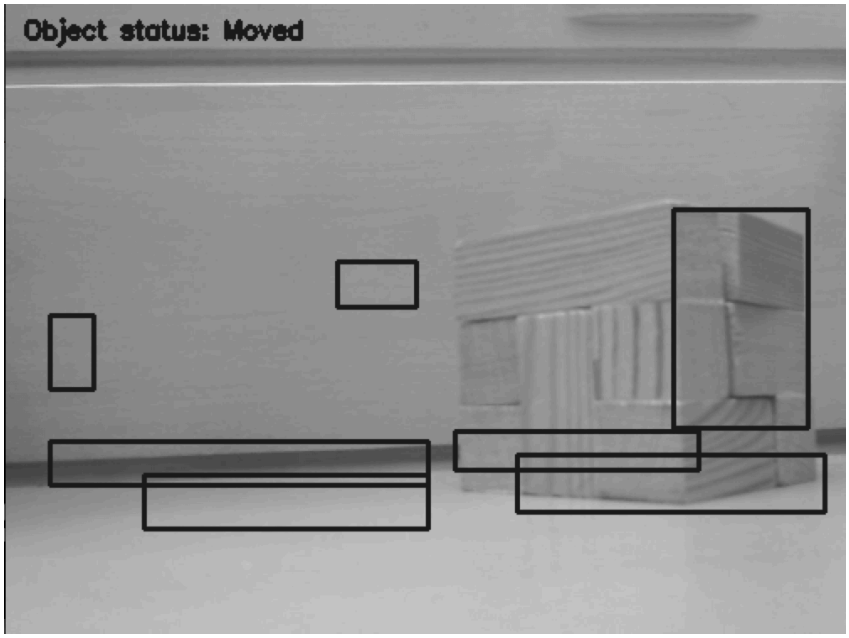


Joonis 4 OpenCV threshold



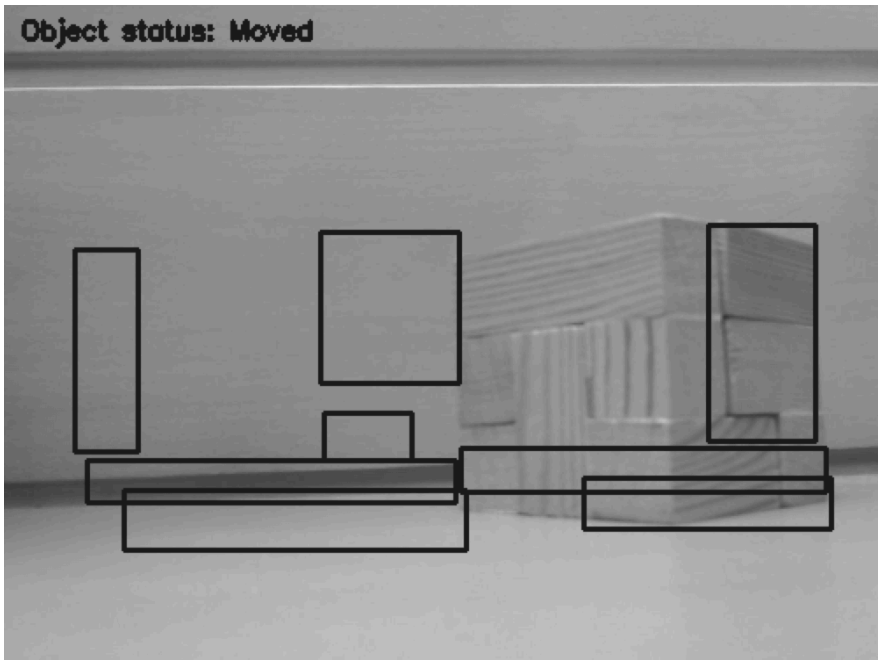
Joonis 5 OpenCv absdiff tulemusele rakendatud thershold

Kontuuride objektide massiivi leidmiseks saab binaarpiltide puhul kasutada OpenCv funktsiooni `findContours` [25]. Tagastatud massiiv sisaldab kontuuride alguskoordinaate koos kõrguse ja laiusega. Igale kontuurile on võimalik leida selle suurus ning liiga väikeste kontuuride puhul ignoreerida seda kui liikumist. Joonis 6 puhul ignoreeritakse kontuure, mille suurus on alla 500 piksli.



Joonis 6 Kontuuride minimaalne suurus 500 pikslit

Joonis 7 on miinimum kontuuri suuruse arvuks määratud 100 pikslit. Jooniselt on näha, et tekkinud kontuure on rohkem ning on mõõtmetelt suuremad. Vähendades minimaalset pikslite arvu suurust veelgi, ehk tekib veelgi rohkem kontuure, võib süsteem muutuda aeglasemaks, kuna töötlemist vajavaid objekte on rohkem.



Joonis 7 Kontuuride minimaalne suurus 100 pikslit

5.2 Inimeste märkamine

Inimese leidmiseks on mõistlik kasutada tunnuseid, mis ei sõltu rassist, soost ega etnilisest päritolust. Üldisel struktuursel tasemel kõigil inimestel pea, kaks kätt, torso ning kaks jalga. OpenCV'l on sisse ehitatud funktsionaalsus, mis suudab jalakäijaid märgata nii pildilt kui videolt. OpenCV installimisel tuleb kaasa eeltreenitud HOG koos Lineaarse SVM klassifikatsiooniga, mida saab eelpool nimetatud otsinguks kasutada. Joonis 8 puhul kasutasin inimeste tuvastamiseks pilti veebilehelt [26].

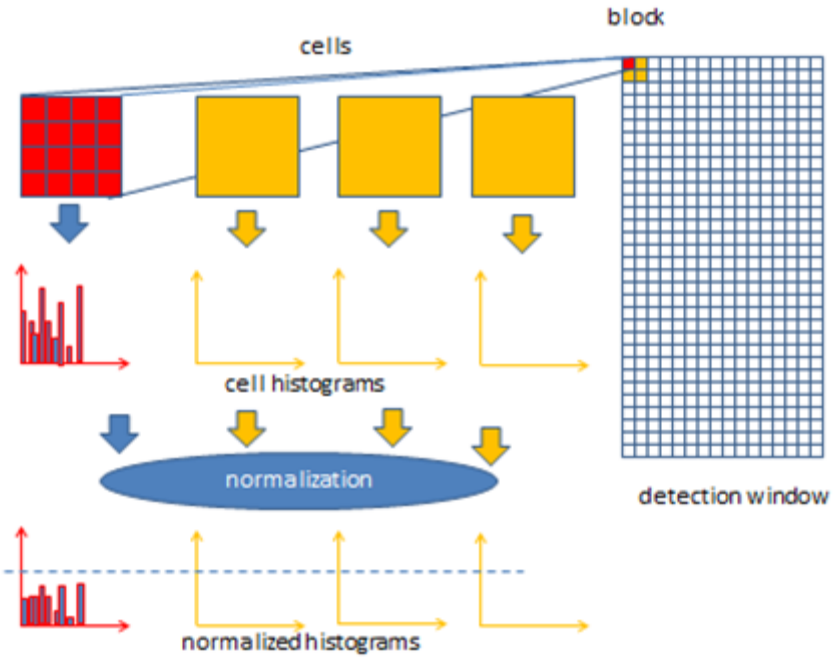
Joonis 8 Inimeste tuvastamine

5.2.1 Histogram of Oriented Gradients

Järgnev seletatav kirjeldus orienteeritud gradientide histogrammi kohta on võetud artiklis [27]. Orienteeritud gradientide histogrammi ehk HOG põhiliseks ideeks on objekti kujude karakteriseerimine kasutades lokaalsete gradientide jaotumise tugevust. Selle saavutamiseks jagatakse pilt plokkideks, mis omakorda jaotatakse väiksemateks ruumilisteks osadeks, mida kutsutakse rakkudeks. HOG algoritmi sammud on järgnevad:

- Iga raku kohta arvutatakse gradientide suunad.
- Koostatakse histogramm kõikidest rakkudest kirjeldamaks tunnuseid.
- Normaliseeritakse plokid ja kalkuleeritakse lõplik HOG tunnus.

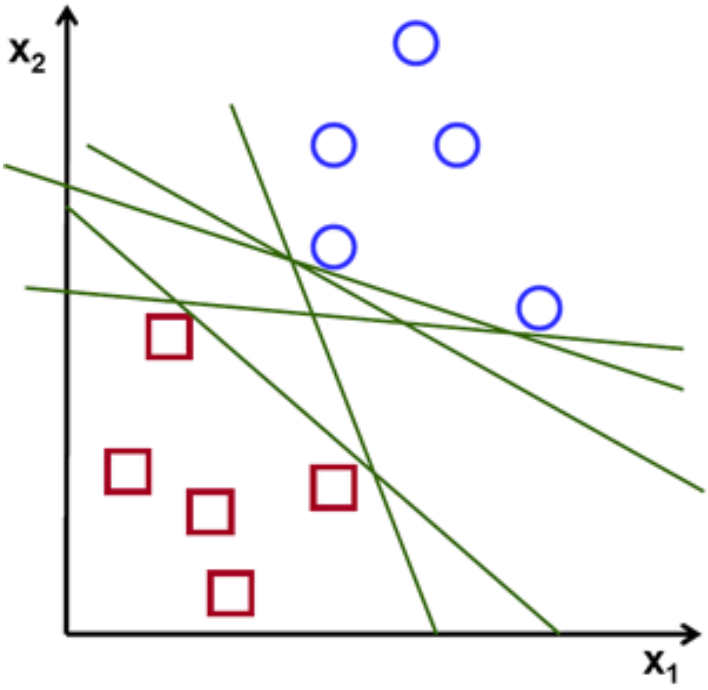
Järgnev HOG algoritmi tööd illustreeriv pilt on võetud veebilehelt [28].



Joonis 9 HOG algoritm

5.2.2 Linear SVM meetod

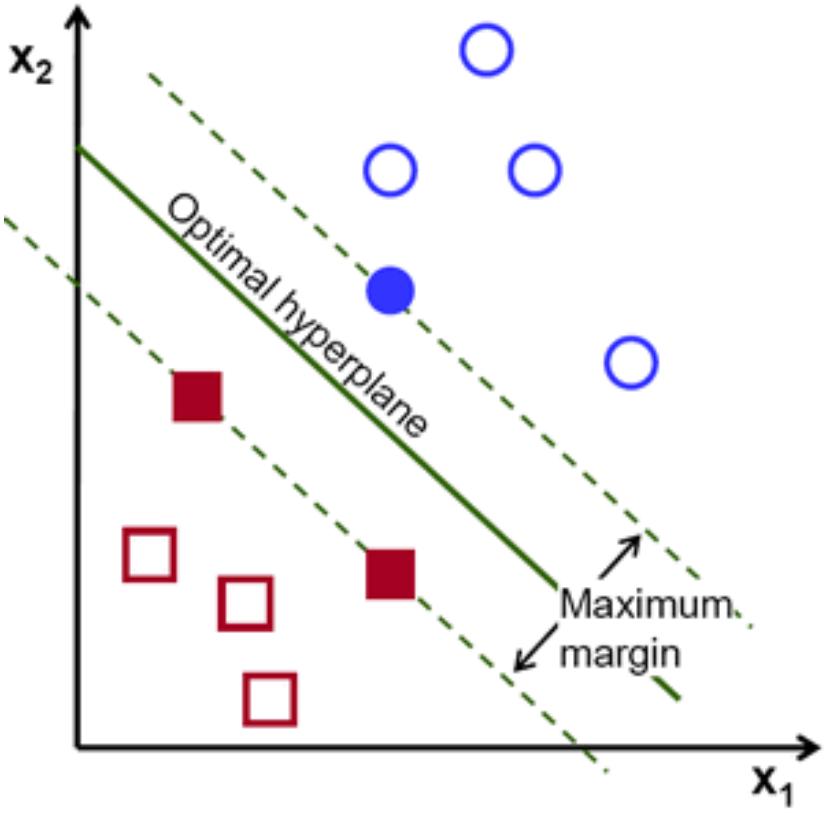
Käesoleva peatüki kirjeldamiseks olen lähtunud artiklist [29] ning näidetena kasutanud seal esitatud jooniseid. Lineaarne SVM on diskreetne klassifikaator formaalselt määratletud eraldavate puutujatega. Teisisõnu, antud märgendatud treeningandmetele (järelevalvega õppega) annab algoritm välja optimaalse puutuja, mis kategoriseerib uued andmed. Olgu näide 2D punktidest, mis on eraldatud sirgete puutuvate joontega. Antud juhul on tegemist kahemõõtmelise koordinaatsüsteemiga näite illustreerimiseks ning lihtsustamiseks.



Joonis 10 Puutujad

Joonis 10 koosneb mitmetest eraldatavatest puutujatest, kuid ükski nendest ei ole sobilik masinõppe probleemi lahendamiseks. Joonte liiga lähedasel möödumisel punktist on müra tundlikkus suur ning ei teki korrektset üldistust. Seetõttu tuleb leida eraldaja, mis on kõigist punktidest võimalikult kaugel.

SVM algoritm põhineb puutujate leidmisel, millega leitakse suurim minimaalne distants treeningandmete näidetele ehk marginaal. Teisisõnu optimaalne eraldaja maksimeerib marginaali treeningandmete vahel [Joonis 11].



Joonis 11 SVM optimaalne eraldaja

5.3 Nägude märkamine

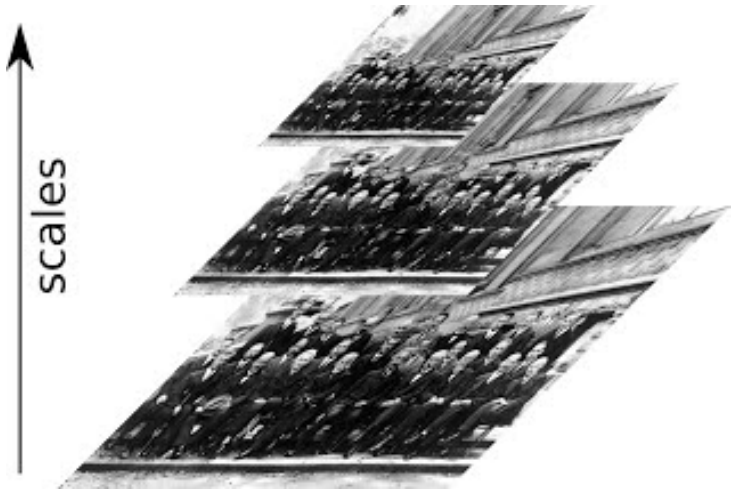
Videolt või pildilt nägude märkamiseks tuleb kas ise õpetada, milline on inimese nägu ja selle iseloomulikud tunnused või kasutada mõnda olemasolevat klassifikaatorit. OpenCV puhul on tulemuseks XML fail, kus on positiivsete ja negatiivsete piltide abil talletatud informatsioon objekti tuvastamiseks. Näo tuvastamiseks on olemas erinevaid klassifikaatoreid, kuid kui on vaja tuvastada mõnda objekti, mille jaoks ei leidu vajalikku klassifikaatorit, on võimalik neid ise luua. Antud töös on näo avastamiseks kasutatud „haarcascade_frontalface_default.xml“ [30] klassifikaatorit, mille kasutamisel tuleb arvestada faili alguses kirjeldatud autorideklaratsiooniga.

OpenCV’s objektide avastamine erinevate kaskaadide (*cascade*) abil on triviaalne. Tuleb luua unikaalne objekt antud kaskaadi jaoks kasutades OpenCV funktsiooni CascadeClassifier. Järgnevalt on kas pildile või video kaadrile (halltoonides) võimalik kasutada OpenCV funktsiooni `detectMultiScale`, mida rakendada eelnevalt loodud kaskaadi objektile. Lisaks kaadrile tuleb ette anda ka kaks parameetrit – skaala koefitsient

ja miinimum naabrite arv. Tulemuseks on objektide (käesoleva töö raames nägude) hulk koos x ja y telje algkoordinaatidega ning antud objekti kõrgus ning laius.

5.3.1 Skaala koefitsient

Skaala koefitsient määrab, kui palju pildi suurust vähendatakse igal kujutise skaalal [31]. Skaala faktorit kasutatakse skaala püramiidi loomiseks, mida illustreerib järgnev joonis [32].

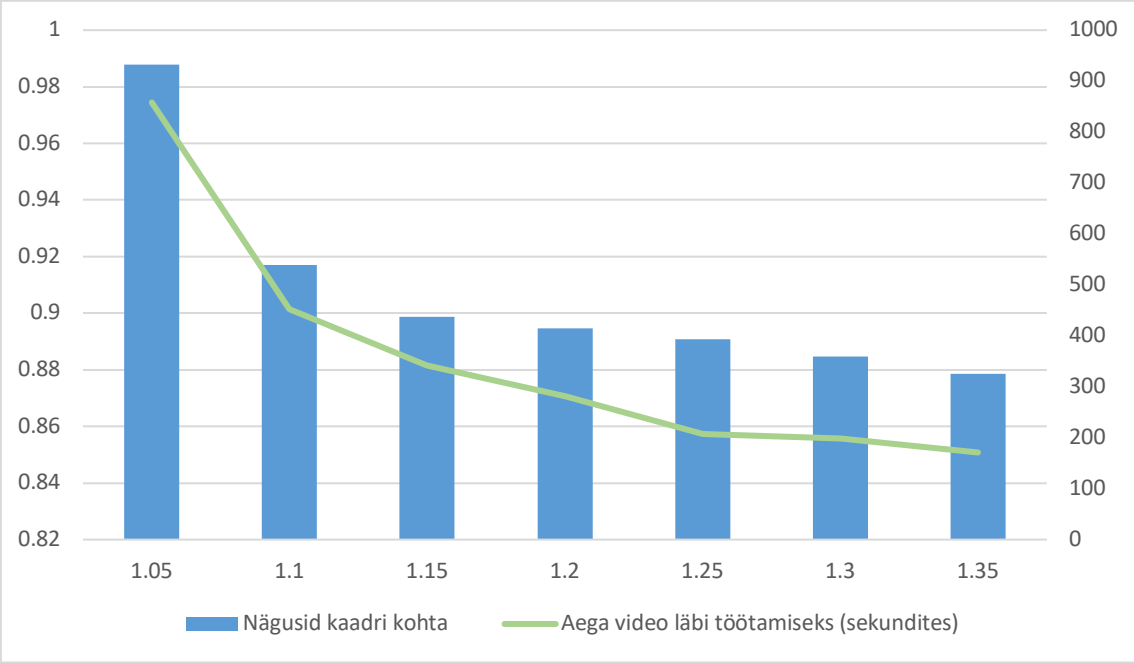


Joonis 12 Skaala püramiid

Videol või pildil ei ole enamasti näod sama suurusega, mis eelnevalt treenitud piltidel. Antud juhul tuleb sisendina antav nägu ümber skaleerida, mis võimaldab algoritmil tuvastada ära õige inimene. Näiteks skaala koefitsiendi 1.05 puhul vähendatakse tuvastamisel iga kord pilti 5%. Mida lähemal on skaala koefitsient ühele, seda suurema tõenäosusega on võimalus tuvastada pildilt nägu, kuid tänu sellele töötab ka programm aeglasemalt. Skaala koefitsiendi suurendamisel töötab programm kiiremini, kuid tuleb arvestada sellega, et osad näod võivad jääda üles leidmata.

Järgnevalt teostasin katse leidmaks optimaalset skaala koefitsienti nägude märkamiseks. Iga katse käigus läbiti 25 sekundiline video kaaderhaaval (kokku 494 kaadrit), kus iga kaadri pealt oli võimalik tuvastada nägu. Skaala alg koefitsient oli 1,05 ning väärtust suurendati iga katse korral 0,05 võrra kuni koefitsiendini 1,4. Iga koefitsiendiga vaadati läbi video kaadri kaupa ning prooviti tuvastada nägu. Miinimum naabrite arvuna kasutasin väärtus 5 iga katse korral.

Järgneval joonisel [Joonis 13] on tulpadena välja toodud keskmine leitud nägude arv kaadris. Jooniselt on näha, et skaala koefitsiendi suurenedes leiti vähem nägusid ehk täpsus vähenes. Samas vaadates rohelist joont on näha, et koefitsiendi suurendamisel kahanes video töötlemiseks kuluv aeg. Kõige väiksema koefitsiendi puhul oli video töötlemiseks kuluv aega ~850 sekundit, mis tegi ühe kaadri töötlemiseks üle 1,5 sekundi. Suurendades koefitsienti kõigest 0,05 punkti võrra, vähenes kaadri töötlemiseks kuluv aeg alla ühe sekundi. Edasisel suurendamisel vähenes nii töötlemiseks kuluv aeg kui ka videolt leitud nägude arv.

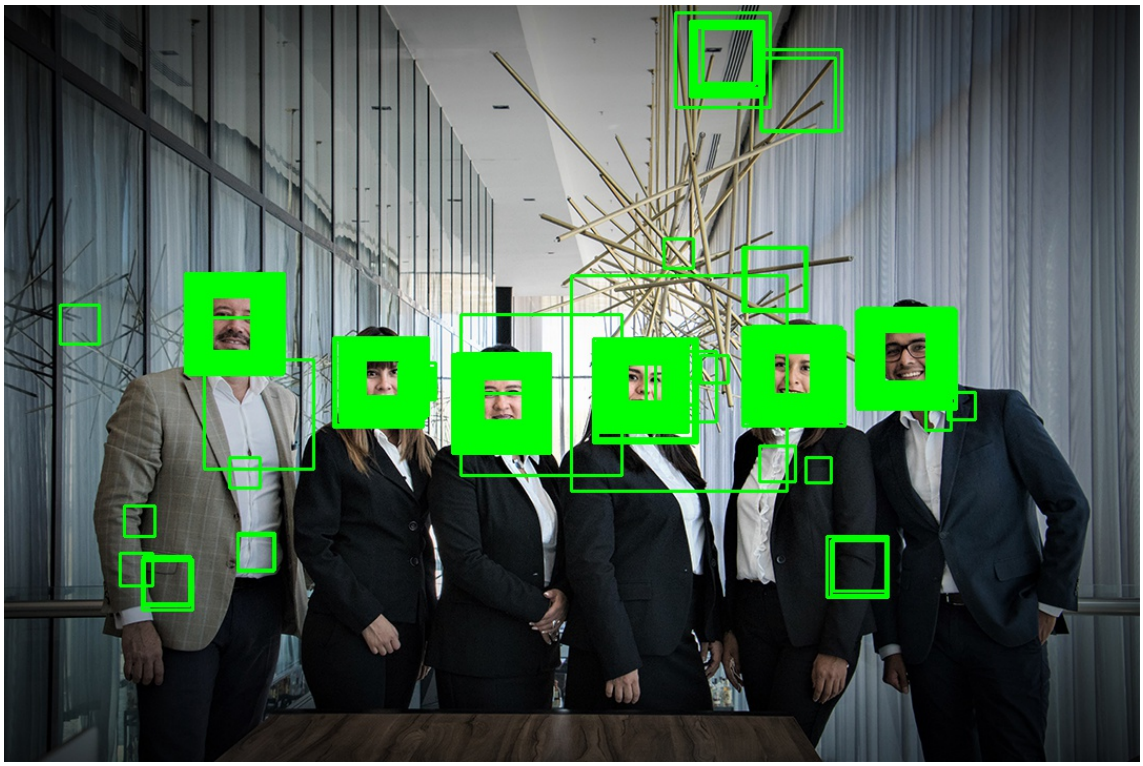


Joonis 13 Skaala koefitsient

5.3.2 Miinimum naabrite arv

Miinimum naabrite arv nägude leidmisel määrab, kui mitu naabrit igal kandidaat ristkülikul peab olema, et teda säilitada, ehk kas lähedusest on leitud veel sarnaseid objekte või mitte [33]. Üldisemas tähenduses käitub antud parameeter kui lävi, mille puhul võib väita, et tegemist on näoga. Mida väiksem number on parameetrina kaasa antud, seda vähem naaber ristkülikuid arvestatakse objekti tuvastamiseks ning vastupidi.

Järgneva nelja katse puhul kasutasin pilti [34], mille pealt on võimalik tuvastada kuue inimese näod. Parameetri 0 puhul märgitakse väga paljusid objekte nägudeks, mis tegelikkuses ei ole näod ehk tekib palju valepositiivseid tulemusi. Konkreetse pildi puhul leidis programm 937 ristkülikut, mida tuvastati näona [Joonis 14].



Joonis 14 Miinimum naabrite arv - 0

Väärtuse suurenedes väheneb valepositiivsete tulemuste esinemine ning üleliigsete objektide leidmine ühe näo juures.

Joonis 15 Miinimum naabrite arv – 4

Pildil, mida sai kasutatud miinimum naabrite arvu testimiseks, sai ideaalse tulemuse väärtuse 15 juures.

Liiga suure väärtuse korral jäetakse tuvastamata osad näod, mis oleks pidanud saama tuvastatuks. Näiteks väärtuse 90 puhul suutis programm tuvastada ainult 4 nägu [Joonis 16].

Joonis 16 Miinimum naabrite arv – 90

5.3.3 Järeldused

Näo leidmisel kaasa antavad parameetrid määravad väga palju lõpptulemust. Skaala koefitsiendi määramine mõjutab olulisel määral süsteemi kiirust. Miinimum naabrite arvu puhul tuleb jälgida, et ei oleks liiga palju valepositiivseid tulemusi, kuna näo tuvastamisel tuleb läbi käia kõik leitud objektid, mis näiteks arvu 0 puhul koormaks süsteemi muutmälu, suurendaks info salvestuse mahtu ning vähendaks töödeldavate kaadrite arvu sekundis.

5.4 Nägude tuvastamine

Nägude tuvastamise esimeseks etapiks on treeningandmete koostamine. Selleks sobivad erinevas suuruses ning värviskaalas pildid, kuna tuvastamise hetkel nende suurust

muudetakse ning konverteeritakse hallidesse toonidesse. Pildilt nägude tuvastamise loogika on sama, nagu punktis 5.2 kirjeldatud. Nägude treenimiseks saab kasutada OpenCV funktsiooni `train`, mille sisendparameetriteks on nägude ja märgendite massiivid. Antud tulemust on võimalik salvestada faili, mis kiirendab süsteemi käivitamist - ei ole vaja õppida uuesti, vaid piisab loodud faili sisse lugemisest.

Pildilt või videolt nägude tuvastamiseks tuleb esialgu leida pildilt nägu ning seda võrrelda treenitud andmetega. Selleks on OpenCV'l funktsioon `predict`, kus õpetatud objekti (eelnevalt loodud failist) võrreldakse pildilt leitud halltoonides näoga. Antud funktsioon tagastab kaks väärtust – mis inimese ta leidis ning mis koefitsiendiga on see tõene. Mida lähemal on koefitsient nullile, seda suurema tõenäosusega on antud inimene pildil või videol. Koefitsiendi kasvades väheneb tõenäosus, et inimene pildil või videol on see, kellena teda tuvastati.

5.4.1 Tuvastamise täpsuse sõltuvus treeningpiltide arvust

Treeningpildid on näo tuvastamise juures oluline faktor, kuna ilma treening andmeteta ei ole võimalik tuvastada objekte. Treeningpiltide arvu sõltuvuse määramiseks sai tehtud katse, kus treeniti võrdse arvu piltidega 6 inimest ning iga katse korral suurendati antud arvu. Tulemuse hindamiseks läbiti salvestatud video ühest isikust iga treeningandme kohta ning vaadeldi valesti tuvastatud kordi ning keskmist koefitsienti videol esineva isiku kindluse kohta. Video koosnes 494st kaadrist, millest tuvastati 455 nägu.

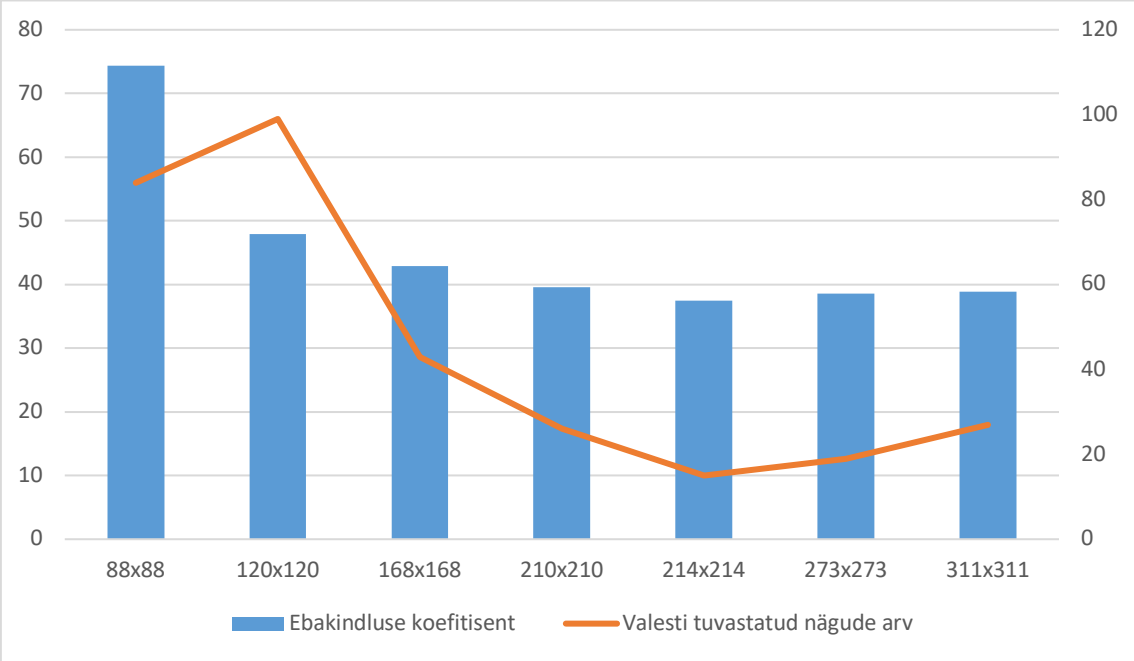
Tabel 4 Treeningandmete arvu mõju tuvastusele

	Keskmine kindluse koefitsient	Teise isikuna tuvastatud kordade arv
50 treening pilti inimese kohta	29.6	48
100 treening pilti inimese kohta	30.23	38
150 treening pilti inimese kohta	29.77	38
200 treening pilti inimese kohta	29.67	30
250 treening pilti inimese kohta	29.21	30
300 treening pilti inimese kohta	29.199	30

Tabelis 4 välja toodud andmete põhjal on näha, et treeningandmete suurendamisel kindluse koefitsient olulisel määral ei paranenud. Samas 200 ja rohkemate treeningandmetega tuvastati videol korrektsemalt õiget inimest ning ei leitud teisi süsteemi sisestatud inimesi.

5.4.2 Treeningpiltide suuruse mõju tuvastamise täpsusele

Treeningandmete pildi suurus võib mõjutada süsteemi täpsust ning kiirust. Katse läbiviimisel kasutasin nii treenimiseks kui testimiseks sama videot, millest esimesed 75% kaadritest kasutasin õppimiseks ning viimased 25% kaadritest testimiseks. Lisaks kasutasin treeningandmeid 5 inimese kohta, mille kõigi puhul võrdselt 200 pilti. Originaal nägude suurus ilma skaleerimata oli keskmisel 210x210 pikslit. Leitud näo suurendamisel treenimise käigus suurenes ebakindluse koefitsient tuvastamisel ning samuti suurenes valesti tuvastamiste arv (videol olev inimene aeti segi teise inimesega). 1,2 kordsel suurenemisel vähenes minimaalselt ebakindluse koefitsient ning valesti tuvastatud kordade arv, kuid pildi edasisel suurendamisel hakkasid mõlemad väärtused tõusma [Joonis 17].

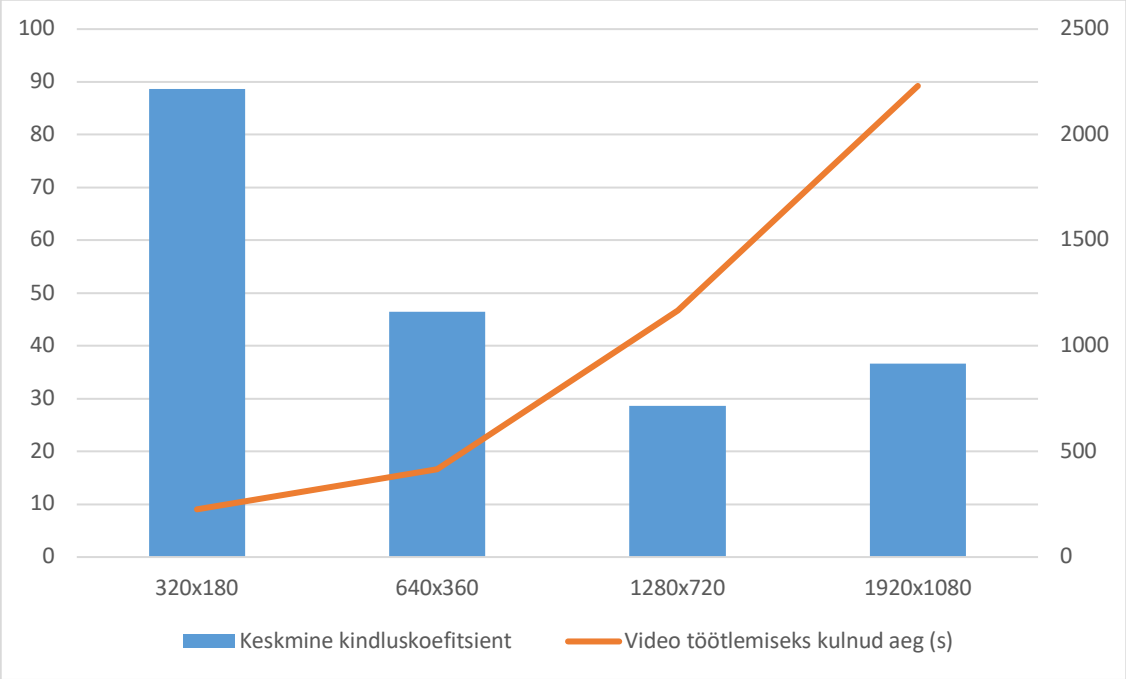


Joonis 17 Treeningpiltide skaleerimise mõju tuvastamisele

5.4.3 Tuvastatava kaadri suuruse mõju tulemusele

Turvasüsteemi videokvaliteet võib mõjutada süsteemi jõudlust ning täpsust objektide tuvastamisel. Erinevatel kaameratel võib olla erinev resolutsioon või kaamera lääts võib olla kriimustatud/määrdunud. Järgneva katse juures treenisin mudeli 200 pildiga ühest inimesest ning testimiseks kasutasin 25 sekundi pikkust videot samast isikust. Video originaal resolutsioon oli 1280x720 pikslit, mida katsete käigus sai suurendatud või

vähendatud. Nägude tuvastamise juures skaala koefitsiendi väärtus oli 1.1 ning miinimum naabrite arv 10.



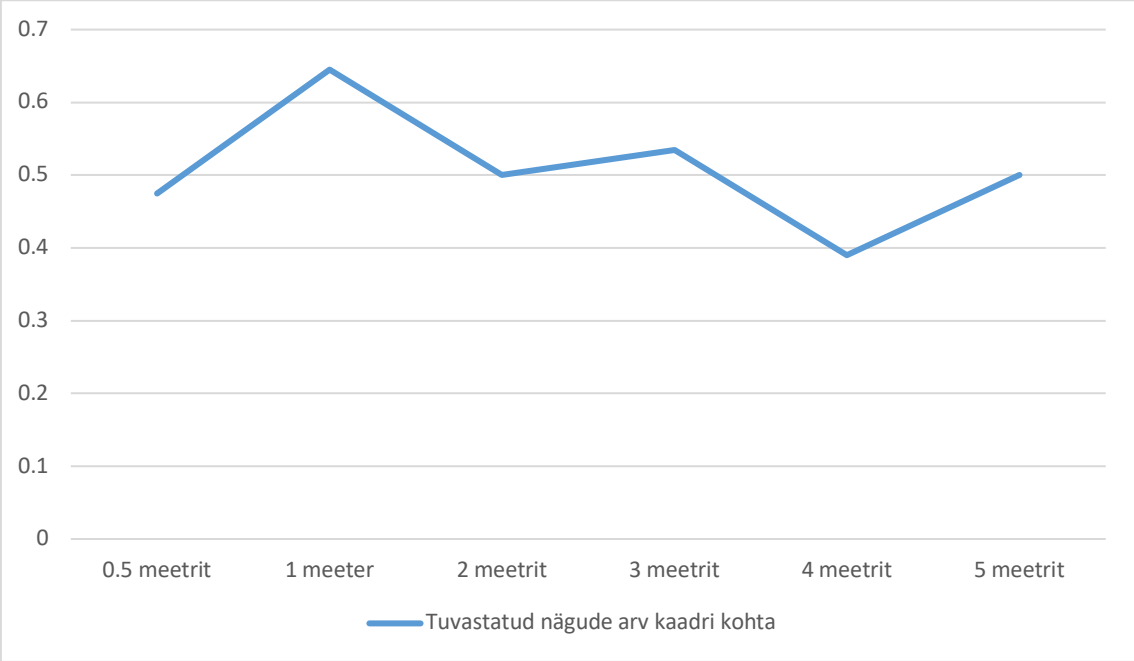
Joonis 18 Video resolutsiooni mõju tuvastusele ja ajale

Vaadeldes eelpool kujutatud joonist [Joonis 18] saab teha mitmeid järeldusi sisendkaadri suuruse mõjust programmi tööle. Nelja kordsel video suuruse vähendamisel lühenes töötlemiseks kuluv aeg oluliselt, kuid samuti vähenes kindlus koefitsient inimese kohta. Kahe kordsel resolutsiooni vähendamisel töötas programm kauem, kuid tuvastatud isikus oldi rohkem kindlamad. Originaal resolutsioon puhul oli koefitsient tuvastatud isiku kohta kõige parem võrreldes teiste katsetega, kuid liiga pikk programmi töö aeg ei sobiks reaalajas töötavale süsteemil – 1167 sekundit ehk üle kahe sekundi kaadri kohta. Neljanda katse puhul vähenes kindlus isiku tuvastamisel. Lisaks suurenes video resolutsiooni kahekordsel suurenemisel ka tööle kuluv aeg ligi kaks korda, mida reaalajas töötava süsteemi puhul ei oleks võimalik kasutada. 493 kaadri jaoks kulus 2231 sekundit, ehk ühe kaadri töötlemiseks kulus ~4,5 sekundit. Tehtud katseid vaadates on näha, et kõige optimaalsemaks resolutsiooniks oli 640x360 pikslit. See tulemus ei olnud küll kõige kindlam isiku tuvastamisel, kuid arvestades tulemus koos kulunud ajaga on antud valik kõige parem.

6 Tulemuste hindamine

Realiseeritud süsteemi tulemuste hindamisel kasutasin eelnevalt leitud kõige optimaalsemaid parameetreid. Igasugust liikumist kaamera vaateväljas, mis on suuremad minimaalsest objekti suurusest, fikseeritakse süsteemi poolt alla ühe sekundi. Antud operatsiooni teostamiseks ei ole vaja palju ressurssi ega keerukaid operatsioone.

Inimese näo leidmine pildilt (kaadrilt) sõltub paljuski kasutatud klassifikaatorist nägude märkamiseks. Järgnevalt teostasin 6 erinevat katset, kus isik seisis erineval kaugusel kaamerast ning liigutas igal videolt pead vasakult paremale 200 kaadri jooksul.



Valitud klassifikaatori abiga suutis süsteem tuvastada väga hästi nägusid ~1 meetri kaugusel kaamerast, kus suudeti nägu paremini märgata külgvaates. Visuaalse ülevaatuse käigus selgus, et näo märkamisel põhiliseks mitte tuvastamise põhjuseks on klassifikaatori suutmatus tuvastada nägu külgvaates. Keskmine tuvastatud nägude arv videos kõigub 0.4 – 0.65 vahel, mis on tingitud erinevast pea liikumise kiirusest.

6.1 Isiku tuvastuse täpsus

Isiku tuvastamisel sõltuvad tulemused paljudest erinevatest aspektidest nagu näiteks treenitud piltide arv, isiku kaugus kaamerast, vaateväljas olevate isikut arv, valgus, näo nurk kaamera suhte jne.

6.1.1 Üksikisiku tuvastus

Järgnev katse teostati 3 erineva isikuga ning isikud olid treenitud sarnaste treeningvideotega sarnastes tingimustes (kaugus, näo liikumine, treeningpiltide arv). Testimise videotes alustasid isikud ~0.5 meetri kaugusel kaamerast ning liikusid 5 sammu kaamerast eemale, igal sammul peatudes ning liigutades pead vasakule ning paremale.

Tabel 5 Üksikisiku tuvastamise tulemused

	Läbitud kaadreid	Korrektselt tuvastatud (keskmine koefitsient)	Valesti tuvastatud kordade arv
Isik 1	440	213 (80,8)	214
Isik 2	889	416 (91,29)	152
Isik 3	867	124(56,78)	624

Vaadates antud tulemusi on näha tulemuste kõikumist, kuigi nii treening- kui testvideod olid koostatud sarnaselt ning treenitud sama arvu piltidega. Visuaalsel vaatlusel programmi töö tulemusele oli näha, et valgus mõjutab tuvastamise tulemust olulisel määral. Sõltumata isiku kaugusest kaamerast suudeti tuvastada nägusid nii kaugemalt kui lähemalt, kuid nägude keeramisel tekitati palju valesid tulemusi. Tehtud test kattis laialdaselt erinevaid stsenaariume nagu näiteks inimese kaugus kaamerast, näo suhe kaameraga, valguse langemine inimese näole ning liikumine.

6.1.2 Mitmete inimeste paralleelne tuvastus

Sarnaselt eelnevalt teostatud katsele (6.1.1) teostasin katse, kasutati samu treeningandmeid mis eelnevalt, kuid testimiseks oli üks video (68 kaadriga) koos kolme isikuga. Videos teostasid isikud samu tegevusi ning on treenitud sarnaste treeningandmetega, mis tõttu eelduseks oli suhteliselt võrdne isikute tuvastamise arv ning koefitsient iga isiku kohta.

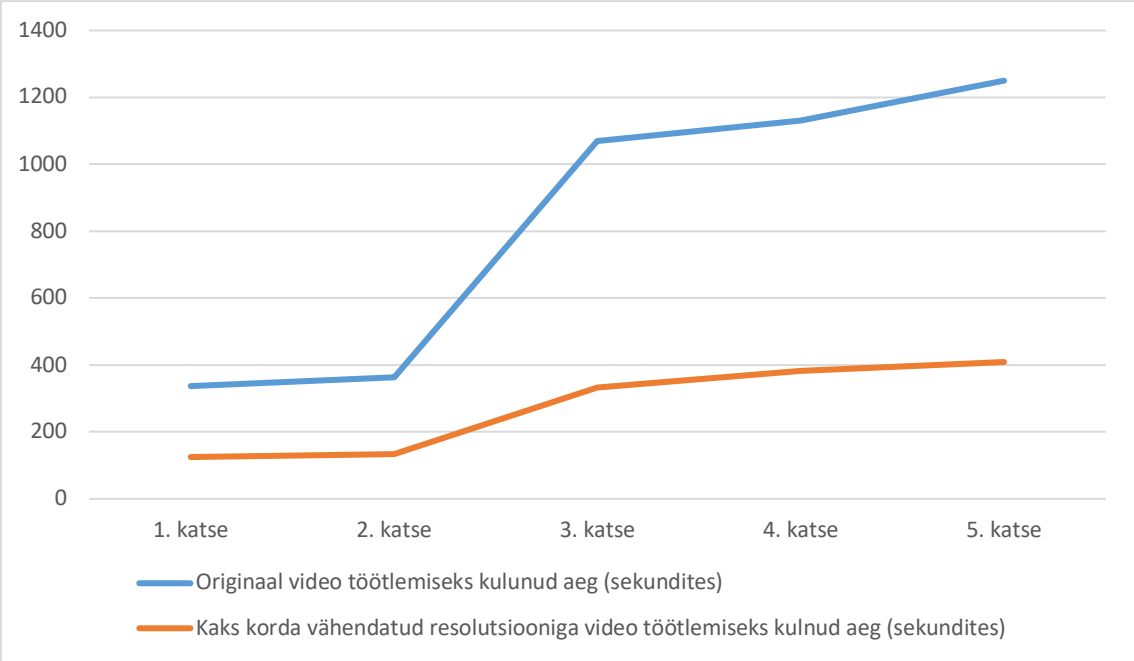
Tabel 6 Mitme isiku samaaegse tuvastamise tulemused

	Tuvastatud arv kokku	Keskmine tuvastamise koefitsient
Isik 1	47	38,07
Isik 2	68	51,57
Isik 3	8	37,90

Kõik isikud olid küll samas kaadris, kuid nendele langev valgus oli erinev. Isikule 2 langes kõige vähem valgust peale, isik 3 oli üle valgustatud ning isik 1 nende vahel. Katse tulemusi vaadates on näha, et esimese ja teise isiku puhul tuvastati nägusid ~70 kaadritest koefitsiendiga üle 38. Kolmanda isikut tuvastati kõigest kaheksal korral, kuid kõige parema koefitsiendiga. Kõige rohkem tuvastati isikut, kellele langes kõige vähem valgust, kuid samas oli ka tuvastamise koefitsient kõige väiksem. Isiku 3 puhul oli tuvastamise arv kõige madalam valguse mõju tõttu. Kokku tuvastati 17 muud isikut süsteemi poolt, ehk ~25% kordadest, keda tegelikult kaadris ei olnud. Võrreldes tabelis 5 välja toodud tulemustega on mitme inimese samaaegne tuvastamine parem, kuid käesoleva testi video oli lühem ning oma olemusel lihtsam (vähem liikumist, lähemal kaamerale), kui eelnev video.

6.2 Jõudlus

Süsteemi jõudluse mõõtmiseks võib ühe katsena vaadelda kaadrite arvu läbimise muutumist erinevate komponentide (liikumise tuvastamine, nägude leidmine, nägude tuvastamine, video salvestamine) lisamisel süsteemi töösse. Kiiruse testimiseks sai salvestatud 1000 kaadriga video, kus toimus liikumist, oli võimalik leida nägu ning tuvastada isikut. Video kiireima läbimisaja saamiseks ei teostatud esimesel katel ühtegi keerukat operatsiooni. Iga järgneva katsega lisati funktsionaalsust juurde [Joonis 19].



Joonis 19 1000 kaadri töötlemise aeg

Esimesel katsel, kus toimus ainult video kaadrite läbi käimine kulus video töötlemiseks originaal video resolutsiooni puhul 337 sekundit ning 2 korda vähendatud resolutsiooniga video puhul 125 sekundit. Teisel katsel lisati juurde liikumist tuvastav funktsioon, mis olulisel määral tulemust ei suurendanud. Kolmanda katse puhul lisati juurde nägude leidmine, mis tõstis video töötlemiseks kuluvat aega umbes kolm korda. Neljandal katsel lisati juurde nägude tuvastamine ning viimasel katsel video loomine ning salvestamine. Viimase kahe funktsionaalsuse lisamisega ei suurenenud programmi töö aeg oluliselt. Töötlemisel kasutatud video originaal resolutsioon on süsteemile liiga koormav ning kõikide funktsionaalsustega kulus ühe kaadri töötlemiseks ~2,5 sekundit. Resolutsiooni kahe kordsel vähendamisel paranes süsteemi kiirus oluliselt ning viienda katse puhul oli ühe kaadri töötlemiseks kuluv aeg ~0,8 sekundit.

Vähene muutmälu piirab Raspberry Pi arvutil ära treenitavate piltide arvu. Kuue inimese treenimisel oli võimalik kasutada iga inimese kohta ~50 treeningpilti. Rohkemate andmete puhul koormati süsteem üle ning mälu sai täis, mille tulemusena ei tekitatud treeningandmete faili.

6.3 Raspberry Pi kahetasemeline autentimine

Üle võrgu kaugsisselogimise turvalisuse tagamiseks sai Raspberry Pi peale installeeritud vabavaraline Google Authenticator [35]. Antud funktsionaalsuse lisamine süsteemi ning ühendamine nutiseadme rakendusega oli lihtne. Lisa sammu realiseerimiseks ennem sissepääsu lubamist tuli installeerida antud teek ning käivitada see käsureal. Esmakordsel käivitusel tehakse uued salajased võtmed, verifitseerimise koodid ja hädajuhtumite koodid, mis tasub üles kirjutada. Järgnevalt sai valida erinevaid võimalusi süsteemi tööks nagu näiteks ühe kasutaja sisse logimine ühe koodiga, läbipääsu lubavate koodide uuendamise kehtivus aja muutmine ja mitu vale koodi võib sisestada 30 sekundi jooksul. Kahetasemelise autentimise kasutamiseks tuleb nutiseadmesse installeerida Google Authenticator rakendus. Rakenduse avamisel on võimalik kas sisestada salajane võti või skaneerida QR kood, mille tulemusena mobiilirakendus ning üles seatud kahtetasemeline autentimine on omavahel ühendatud. Viimase sammuna tuleb lisada installeeritud autentimise viis süsteemi küsimaks antud koodi lisaks tavalisele paroolile. Kahetasemelise autentimise rakendumist süsteemi kaugsisselogimiseks on toodud välja lisas 1.

6.4 Veebirakendus

Django raamistik käesoleva turvasüsteemi kasutajaliidese realiseerimiseks osutus sobilikuks. Projekti algkonfiguratsiooni loomine oli lihtne, kaustade ülesehituse struktuur oli loogiline ning sisse ehitatud funktsioonid (näiteks autentimine) hõlbustasid kiiremat arendust. Kuna tegemist on lokaalse turvasüsteemiga ning kasutajate hulk on väike, siis veebirakenduse koormus süsteemile on minimaalne. Valminud veebirakenduses on sisseloginud kasutajal võimalik näha nimekirja turvasüsteemi poolt loodud videotest ning vaadata videoid [Lisa 2 – Veebirakenduse kasutaja vaade].

7 Edasine töö

Töös realiseeritud prototüüpi on võimalik palju edasi arendada ning uurida erinevaid võimalusi seotud näotuvastusega, turvalisusega, optimeerimisega kui ka kasutajaliidese mugavamaks tegemist.

7.1 Nägude tuvastamise täpsuse suurendamine

Antud töö käigus jäi mitme näo samaaegsel tuvastamisel täpsus üpriski madalaks ning tihti aeti nägusid ka segamini omavahel. Probleemi lahenduse leidmiseks võiks katsetada erinevaid olukordi erineva hulga treeningandmetega, muuta näotuvastusel kasutatavaid parameetreid ning video ja treeningandmete kvaliteeti.

7.2 Erinevate objektide tuvastamine

Praeguse lahenduse korral iga liikumise korral alustatakse (või jätkatakse) video salvestamist. Süsteemi töö optimeerimiseks ning salvestusmahu vähendamiseks võiks tuvastada koduloomi ning nende poolt tekitatud liikumist mitte arvestada.

7.3 Veebirakenduse edasiarendus

Realiseeritud prototüübi veebirakendus on suhteliselt algeline ning vajaks kindlasti edasist arendust. Kasutajale võiks anda rohkem kontrolli süsteemi töö üle. Lisada võimalusi mingeid objekte tuvastada, mingeid objekte ignoreerida, ajutiselt süsteemi töö peata jne. Samuti võiks kasutajal olla kontroll salvestatud videote üle. Võiks saada määrata, mitme päeva taguseid salvestatud videoid kustutatakse ning saada üksikuid videoid kustutada.

8 Kokkuvõte

Antud töö eesmärgiks oli luua näotuvastusel põhinev turvasüsteem kasutades Raspberry Pi arvutit koos tavalise veebikaameraga. Seotud töödest sai kinnitust sellele, et turvasüsteemid on laialdaselt kasutuses, tehisintellekte ja masinõppe programme kasutatakse aina rohkem antud valdkonnas ning Raspberry Pi, sõltumata oma hinnast ning vähesest jõudlusest, omab piisavalt jõudlust, et kasutada erinevaid teeke objektide tuvastamiseks ja veebilehtede majutuseks.

Lõputöö raames jõuti järeldusele, et:

- Süsteemi hinda on võimalik hoida madalamana võrreldes teiste turul olevate toodetega
- Süsteemile püstitatud esialgseid nõudeid oli võimalik teostada kasutades vabavaralist tarkvara ning puudus vajadus väliste teenuste kasutamiseks.
- Realiseeritud prototüüp suudab hästi märgata liikumist ning leida nägusid, kuid nägude tuvastamisel on tulemused ebakindlad ning vähese täpsusega.
- Raspberry Pi vähene jõudlus ei osutunud takistuseks turvasüsteemi arendusele, kuid seadis piiranguid erinevate protseduuride täpsusele ning mahule.

Töö tulemusena valminud prototüüpi on võimalik kasutada tegevuste salvestamiseks ning nägude tuvastamisel, kuid praegusel hetkel tuleb arvestada võimalike valehäiretega isiku tuvastamise osas. Süsteemi edasi arendamisel on võimalik parandada turvasüsteemi täpsust ning samuti eelnevas peatükis [7] on välja toodud mõned võimalikud viisid arendatud prototüübi edasiarendamiseks.

Kasutatud kirjandus

[1] Yolanda, „Home Burglary and Crime Statistics,“ https://reolink.com/home-burglary-crime-statistics/.

[2] Arvutitark, „Raspberry Pi 3 model B,“ [Võrgumaterjal]. Available: https://arvutitark.ee/est/tootekataloog/0/Raspberry-Pi-3-model-B-208670.

[3] Arvutitark, „Veebikaamerate otsing,“ [Võrgumaterjal]. Available: https://arvutitark.ee/est/Otsing?q=veebikaamera.

[4] R. Group, „What is 2FA?,“ https://alarm.riscogroup.com/en-GB/blog/burglar-alarms-through-ages-brief-history-modern-home-security.

[5] K. Zetter, „Wired,“ [Võrgumaterjal]. Available: https://www.wired.com/2014/07/hacking-home-alarms/.

[6] „Netatmo,“ [Võrgumaterjal]. Available: https://www.netatmo.com/en-US/product/security/.

[7] „BuddyGuard,“ [Võrgumaterjal]. Available: https://www.buddyguard.io/.

[8] „Floodlight Cam,“ [Võrgumaterjal]. Available: https://eu.ring.com/collections/security-cams/products/floodlight-cam?variant=2732208685079 .

[9] L. Z. K. C. a. H.-A. C. Changmin Lee, „Securing smart home: Technologies, security challenges, and security requirements,“ %1 *2014 IEEE Conference on Communications and Network Security*, San Francisco, CA, USA, 2014.

[10] Atlassian, „Jira Software,“ [Võrgumaterjal]. Available: https://www.atlassian.com/software/jira.

[11] Atlassian, „Trello,“ [Võrgumaterjal]. Available: https://trello.com/.

[12] R.W.Saaty, „The analytic hierarchy process—what it is and how it is used,“ %1 *Mathematical Modelling*, 4922 Ellsworth Avenue, Pittsburgh, PA 15213, U.S.A., Elsevier, 1987, pp. Volume 9, 161-176.

[13] H. a. T. J. K.N Karthick kumar, „Motion Activated Security Camera using Raspberry Pi,“ http://ieeexplore.ieee.org/document/8286658/, 2017.

[14] V. ,. S. S. Shakthi Murugan.K.H, „Security System Using Raspberry Pi,“ %1 *2017 Third International Conference on Science Technology Engineering & Management (ICONSTEM)*, Chennai, India, 2017.

[15] T. L. E. O. a. S. H. Abdelhakim Ahmim, „Design and Implementation of a Home Automation System for Smart Grid Applications,“ %1 *2016 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, 2016.

[16] T. Y. X. Y. G. Chao Shen, „User practice in password security: An empirical study of real-life passwords in the wild,“ %1 *Computers & Security*, Beijing, Elsevier, 2016, pp. Volume 61, 130-141.

[17] M. Y. V. S. a. S. W. Mehrdad Zaker Shahrak, „Two-Way Real Time Multimedia Stream Authentication Using Physical Unclonable Functions,“ %1 *2016 IEEE 18th International Workshop on Multimedia Signal Processing (MMSP)*, Montreal, QC, Canada, 2016.

[18] Google, „Cloud Vision API,“ [Võrgumaterjal]. Available: https://cloud.google.com/vision/.

[19] Amazon, „Amazon Rekognition,“ [Võrgumaterjal]. Available: https://aws.amazon.com/rekognition/.

[20] B. Virdee-Chapman, „Face Recognition: Kairos vs Microsoft vs Google vs Amazon vs OpenCV,“ [Võrgumaterjal]. Available: https://www.kairos.com/blog/face-recognition-kairos-vs-microsoft-vs-google-vs-amazon-vs-opencv.

[21] B. Amos, „OpenFace,“ [Võrgumaterjal]. Available: https://cmusatyalab.github.io/openface/.

[22] R. Brown, „Django vs Flask vs Pyramid,“ [Võrgumaterjal]. Available: https://www.airpair.com/python/posts/django-flask-pyramid.

[23] D. Z. N. C. James J. Dicarlo, „How does the brain solve visual object recognition,“ Neuron, 2012.

[24] OpenCV, „Image Thresholding,“ [Võrgumaterjal]. Available: https://docs.opencv.org/3.4.0/d7/d4d/tutorial_py_thresholding.html.

[25] OpenCV, „Structural analysis and shape descriptors,“ [Võrgumaterjal]. Available: https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html.

[26] K. Rocha, „Pexels,“ [Võrgumaterjal]. Available: https://www.pexels.com/photo/people-brasil-guys-avpaulista-109919/.

[27] M. S. S. M. I. A. Rania A. Elsayed, „Hand gesture recognition based on dimensionality reduction of histogram of oriented gradients,“ 2017 Japan-Africa Conference on Electronics, Communications and Computers (JAC-ECC), Alexandria, Egypt, 2017.

[28] Intel, „Histogram of Oriented Gradients (HOG) Descriptor,“ [Võrgumaterjal]. Available: https://software.intel.com/en-us/ipp-dev-reference-histogram-of-oriented-gradients-hog-descriptor.

[29] OpenCV, „Introduction to Support Vector Machines,“ [Võrgumaterjal]. Available: https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html.

[30] I. Corporation, „Haascascade frontalface default,“ [Võrgumaterjal]. Available: https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_frontalface_default.xml.

[31] P. J. M. B. Joseph Howse, OpenCV: Computer Vision Projects with Python, Packt Publishing Ltd, 2016.

[32] Z. Ye, „Viola-Jones Face Detection,“ [Võrgumaterjal]. Available: https://sites.google.com/site/5kk73gpu2012/assignment/viola-jones-face-detection#TOC-Image-Pyramid.

[33] OpenCV, „Cascade Classification,“ [Võrgumaterjal]. Available: https://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html#cascadeclassifier-detectmultiscale.

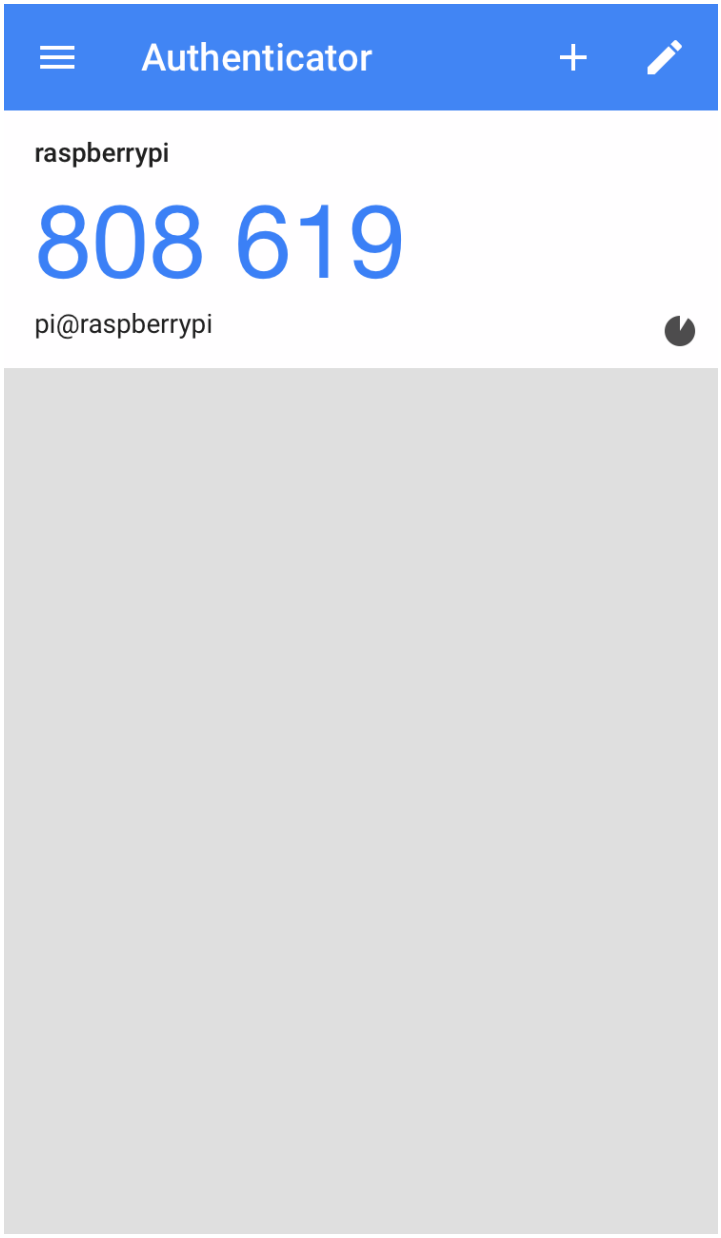
[34] M. Torres, „Group of people in dress suits,“ [Võrgumaterjal]. Available: https://www.pexels.com/photo/group-of-people-in-dress-suits-776615/.

[35] Google, „google-authenticator-libpam,“ [Võrgumaterjal]. Available: https://github.com/google/google-authenticator-libpam.

[36] SecurEnvoy, „SecurEnvoy,“ [Võrgumaterjal]. Available: https://www.securenvoy.com/two-factor-authentication/what-is-2fa.shtm.

Lisa 1 – Kahetasemeline autentimine kaugsisselogimisel

Nutiseadme rakenduses Google Authenticator kuvatakse hetke läbipääsu kood, mis muutub iga 30 sekundi tagant.

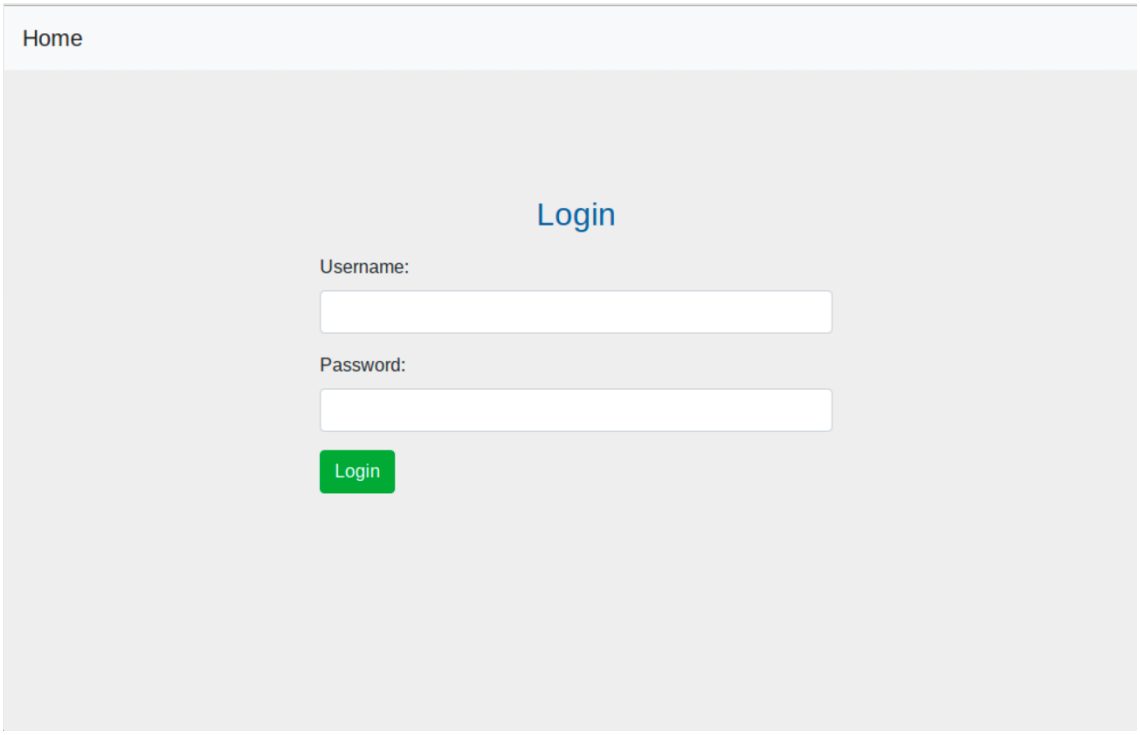


Rakenduses kuvatud kood tuleb esimesena sisestada kaugsisselogimisel ning selle korrektsel sisestusel saab sisestada süsteemi parooli.

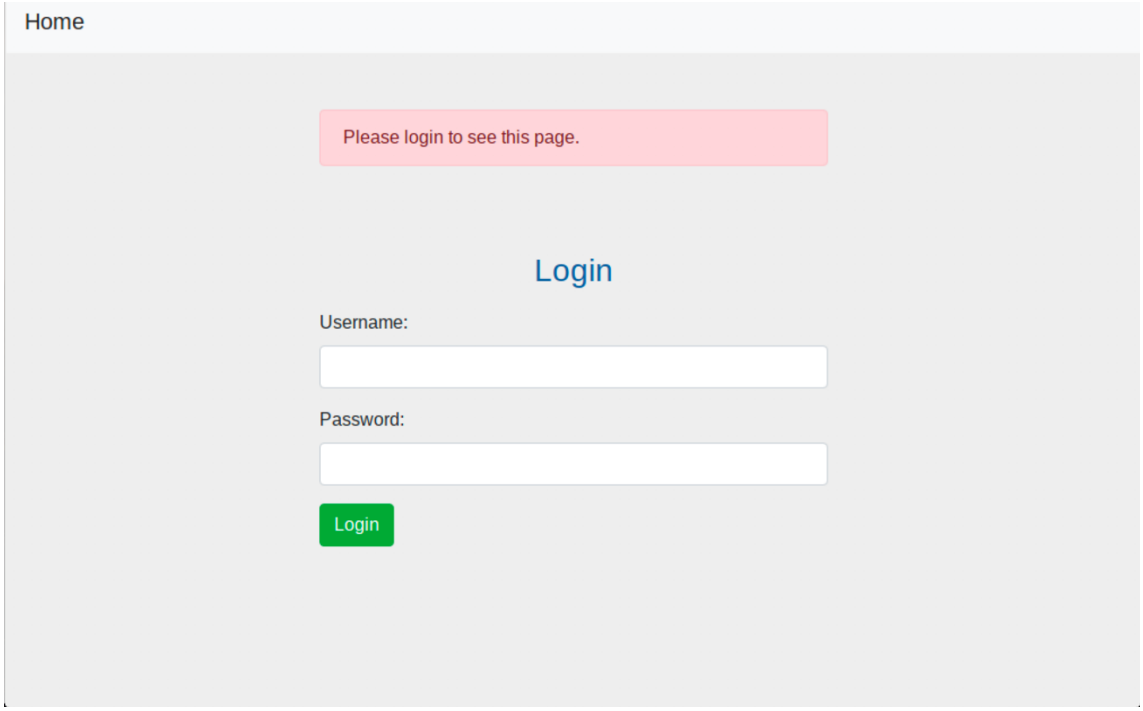
```
[Reimos-MacBook-Pro:~ reimorooparg$ ssh pi@192.168.0.14
[Verification code:
[Password:
```

Lisa 2 – Veebirakenduse kasutaja vaade

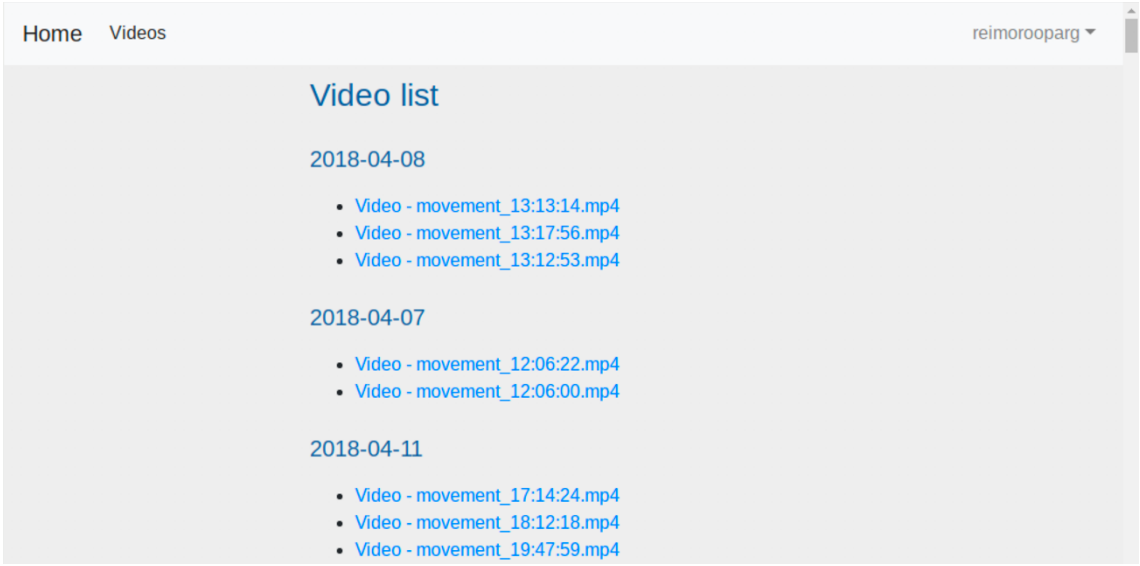
Veebirakendusse sisse logimine.



Sisselogimata ei lubata veebilehe sisus vaadata, vaid suunatakse sisselogimise lehele.



Nimekiri salvestatud videotest päevade kaupa.



Turvavideo salvestuse vaatamine.