VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Joose Raiski

YOLO-POHJAINEN HAHMONTUNNISTUSJÄRJESTELMÄ

Tekniikka
2019

VAASAN AMMATTIKORKEAKOULU
Tietotekniikka

TIIVISTELMÄ

Tekijä	Joose Raiski
Opinnäytetyön nimi	YOLO-pohjainen hahmontunnistusjärjestelmä
Vuosi	2019
Kieli	suomi
Sivumäärä	34
Ohjaaja	Antti Virtanen

Tämän työn tarkoituksena oli hahmontunnistuksen demoympäristön luominen VAMKille käyttäen Raspberry Pi:tä, Movidius NCS:ä ja YOLO-hahmontunnistustekniikkaa. Demoympäristö piti myös testata PC:llä käyttäen kahta eri tasoista NVIDIAn näytönohjainta, jotta tuloksia voidaan vertailla.

Työ on pohjimmiltaan tekoälyprojekti ja lukija perehdytetään tekoälyn perusasioihin. Työ eteni käytännön kokeiluilla olemassa olevien projektien pohjalta ja niitä muokkaamalla tarpeen mukaan. Työssä käytettiin Etcheriä, OpenCV:tä, NCSDK:ta ja OpenVinoa. Työssä käytetty tutkimusmateriaali löytyy internetistä.

Demoympäristön onnistuneen konfiguroinnin jälkeen saatiin selville, että Raspberry Pi:n tehot, edes tekoälykiihdyttimen avulla, eivät riitä tarkkaan reaaliaikaiseen hahmontunnistukseen. Vaikka työtä ei ehditty toteuttaa PC:llä ja vertailun puuttuessa voidaan silti todeta, että Raspberry Pi:tä voi käyttää lähinnä hahmontunnistukseen tutustumisessa ja pienissä projekteissa.

Avainsanat tekoäly, hahmontunnistus, Raspberry Pi, YOLO

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikka

ABSTRACT

Author	Joose Raiski
Title	YOLO Based Object Detection System
Year	2019
Language	Finnish
Pages	34
Name of Supervisor	Antti Virtanen

The aim of this thesis was to create an object detection demonstration environment for VAMK using Raspberry Pi and the YOLO object detection technique. The object detection environment was also to be tested on PC and using two different NVIDIA graphical processing units to get data for comparison.

The project is fundamentally an artificial intelligence project and the reader is introduced with the basics of the artificial intelligence. The project progressed by experimenting on existing projects and by modifying them. Etcher, OpenCV, NCSDK and OpenVino were used in this project. The research material used in this thesis can be found on the internet.

After a successful configuration of the object detection environment it was found out that the performance of the Raspberry Pi was not suitable for real-time object detection even when boosted by an artificial intelligence accelerator. Even without the comparison data it can be said that the Raspberry Pi can be used mainly for introduction to object detection and small-scale projects.

Keywords Artificial Intelligence, Object Detection, Raspberry Pi and YOLO

SISÄLLYS

LYHENNELUETTELO

DNN	Deep Neural Network. Syvä neuroverkko, joka käsittelee dataa useassa eri kerroksessa.
FAT32	File Allocation Table. Tiedostojärjestelmä, jota käytetään nykyään lähinnä USB-muisteilla ja muistikorteilla.
FPS	Frames Per Second. Kuvaa per sekunti.
GB	Gigabyte eli gigatavu. Tallennuskapasiteetin mittayksikkö. Miljardi tavua.
GHz	Gigahertsi. Taajuuden yksikkö. Miljardi hertsiä.
HDMI	High Definition Multimedia Interface. Digitaalisen kuvan ja äänen siirtämiseen tehty liitäntästandardi.
LPDDR3	Low Power Double Data Rate -memory. Vähän virtaa käyttävä muisti
NCS	Neural Compute Stick. Intelin valmistama tekoälykiihdytin.
NCSDK	Neural Compute Stick Development Kit. NCS:lle tehty kehityspaketti
NMS	Non-Maximum Supression. Funktio, jolla hahmontunnistuksessa poistetaan turhat havainnot.
R-CNN	Regional Convolutional Neural Network. Hahmontunnistustekniikka
SD	Secure Digital. Muistikorttityyppi, jossa on kopiointi- ja ylikirjoitussuoja
SDHC	Secure Digital High Capacity. Pienin SD-kortti, jolla on korkea kapasiteetti.
SID	Synthesis of Integral Design. Prosessoreita suunnitteleva järjestelmä 1980-luvulla.
SSD	Single Shot Detector. Hahmontunnistustekniikka.
SSH	Secure SHell. Salatun tietoliikenteen protokolla.
SVM	Support Verctor Machine. Toimii regressio- ja luokitusfunktiona.

USB	Universal Serial Bus. Sarjaväyläarkkitehtuuri tietokoneen oheislaitteiden liittämiselle.
VNC	Virtual Network Computing. Graafisen käyttöliittymän etäkäytön protokolla
VPU	Vision Processing Unit. Konenäköä varten suunniteltu prosessori
YOLO	You Only Look Once. Hahmontunnistustekniikka.

1 JOHDANTO

Tämän työn taustana oli tekoälyyn ja hahmontunnistukseen tutustuminen sekä hahmontunnistuksen demoympäristön luominen VAMKille. Alustaksi valittiin Raspberry Pi ja hahmontunnistustekniikkana käytetään YOLOa. Päätavoitteena oli saada aikaan järjestelmä, joka tunnistaa esineitä reaaliaikaisesti videosyötteeltä. Tavoitteena oli myös käyttää Movidius Neural Compute Stickin antamaa lisätehoa, verrata tuloksia Raspberry Pi:n ja PC:n välillä ja tutkia, kuinka tekoälylle opetetaan uusia objekteja.

Hahmontunnistus on tärkeä osa nykypäivän tekoälyä. Sen avulla voidaan valvoa turvallisuutta, tunnistaa henkilöitä ja automatisoida dynaamisia tai ihmiselle haastavia tuotanto- ja tarkistusprosesseja. YOLOa on tutkimusmateriaalin perusteella käytetty lukuisissa pienissä projekteissa. Raspberry Pi:llä tehdyissä projekteissa oli usein käytetty apuna Movidius NCS-tekoälykiihdytintä johtuen Raspberry Pi:n rajallisista tehoista.

Aluksi työssä tutustutaan työhön liittyvään teoriaan. Sen jälkeen tutkitaan työssä käytettäviä laitteita ja tarvikkeita, jonka jälkeen käydään läpi työssä käytetyt ohjelmistot. Työn teoriapuoleen tutustumisen jälkeen alkaa käytännön osuus. Kyseisessä luvussa käydään työn eri vaiheet läpi. Lopuksi kerätään työstä saadut tulokset ja tehdään johtopäätökset.

2 TEORIATAUSTA

Tässä luvussa tutustutaan lyhyesti tekoälyyn, sen historiaan ja käyttötarkoituksiin. Sen jälkeen tutkitaan, mitä hahmontunnistus on ja kuinka tässä työssä käytetty hahmontunnistustekniikka toimii.

2.1 Tekoäly

Tekoäly on tietojenkäsittelytieteen osa-alue, joka pyrkii kehittämään koneita ja ohjelmistoja, jotka osaavat tehdä älykkäitä päätöksiä /1/. Älykkyydellä yleisesti tarkoitetaan loogisuutta, kykyä ratkaista ongelmia, suunnitella etukäteen, tunneälyä, tilanteiden ja asioiden ymmärtämistä, oppimiskykyä, itsetajuntaa ja luovuutta /2/. Tekoälyn tavoitteena on suoriutua annetusta ongelmasta parhaalla mahdollisella lopputuloksella. Yksinkertaisimmat tekoälyt perustustavat päätöksensä ennalta määrättyjen sääntöjen perusteella. Esimerkiksi ristinollaa pelaava tekoäly, joka tekee tietyn ennalta ohjelmoidun siirron riippuen vastapelaajan tekemästä siirrosta. /3/

Termin tekoäly keksi John McCarthy vuonna 1956 järjestetyssä konferenssissa Dartmouthin yliopistossa /4/. Vuonna 1981 käytettiin SID nimistä järjestelmää, joka suunnitteli prosessoreita. Järjestelmä koostui tuhannesta käsin kirjoitetusta säännöstä ja se teki noin sata kertaa vähemmän virheitä kuin ihmiset. Vuonna 1997 Deep Bluesta tuli ensimmäinen tietokone, joka voitti shakin maailmanmestarit. /2/

Tekoäly on kehittynyt valtavasti viime vuosikymmenien aikana tekniikoiden kehittyessä ja tietokoneiden laskentatehon noustessa. Hyvinä esimerkkeinä on Siri, joka on Applen kehittämä puheohjattava assistentti ja Amazonin kehittämä Alexa, joka toimii älykoteja ohjaavana keskuksena /5/.

2.1.1 Koneoppiminen

Koneoppiminen (Machine Learning) on tapa, jolla tekoäly saadaan opetettua itsenäisesti tekemään ihmisille työläitä ja aikaa vaativia tehtäviä tai sellaisia tehtäviä,

joiden käsin ohjelmoiminen olisi lähes mahdotonta /6/. Koneoppimistavat voidaan luokitella neljään eri kategoriaan: valvottu koneoppiminen (supervised machine learning), valvomaton koneoppiminen (unsupervised machine learning), puolivalvottu koneoppiminen (semi-supervised machine learning) ja vahvistusoppiminen (reinforcement machine learning) /7/.

Valvotussa koneoppimisessa käytetään aluksi valmiiksi annettua dataa. Ennalta annetulle datalle (input) tekoäly yrittää luoda laskukaavan, jolla se saisi halutun lopputuloksen (output). Tuloksien eroavaisuuksien perusteella tekoäly muokkaa laskukaavaa seuraavaa laskua varten. /7, 8/.

Valvomattomassa koneoppimisessa tekoälylle ei anneta valmista dataa. Tekoälyn on tarkoitus löytää keräämästään datasta kuvioita ja toistuvuuksia, joita voidaan hyödyntää. /7/. Kuvioita voidaan tutkia vielä syvemmin etsimällä niistä ryhmittymiä ja yhteneväisyyksiä /8/.

Puolivalvotun koneoppimisen idea on sekoitus kahdesta edellä mainitusta tavasta. Tekoälyllä on käytössä pieni määrä tunnettua dataa, suurinta osaa ei ole merkitty. Tunnetun datan pohjalta tekoäly vähitellen opettaa itseään tunnistamaan tuntematonta ja merkitsemätöntä dataa. Suuri osa tämän hetken tekoälyistä kuuluu tähän kategoriaan, sillä merkityn datan tuottaminen voi viedä paljonkin resursseja. /7, 8/.

Vahvistusoppiminen perustuu yrityksen ja erehdyksen menetelmään (trial and error method) ja palkitsemiseen. Tekoälylle annetaan yksi tai useampi tavoite, joita suorittamalla se muokkaa toimintaansa. Mitä pidemmälle tekoäly pääsee tavoitteissaan, sitä todennäköisemmin se jatkaa samankaltaisella toimintamallilla. /9/.

2.1.2 Syväoppiminen

Syväoppiminen (Deep Learning) on sovellettu koneoppimistekniikka. Se perustuu syvään neuroverkkoon, joka koostuu syötetasosta (input) ja tuotostasosta (output), joiden välillä on piilokerroksia (hidden layers). Jokainen piilokerros koostuu solmuista (nodes). Syötteiden siirtyessä seuraavan tason solmuille, syötteille annetaan

painoarvo (weight). Solmu laskee painotetut syötteet jollain kaavalla ja aktivointifunktio päättää jatkaako solmun syöte seuraavalle kerrokselle. Neuroverkko oppii ajan kanssa kuinka eri piirteet eli painotetut arvot, vaikuttavat toisiinsa ja lopputulokseen. /10/.

2.2 Kuvanluokitus ja hahmontunnistus

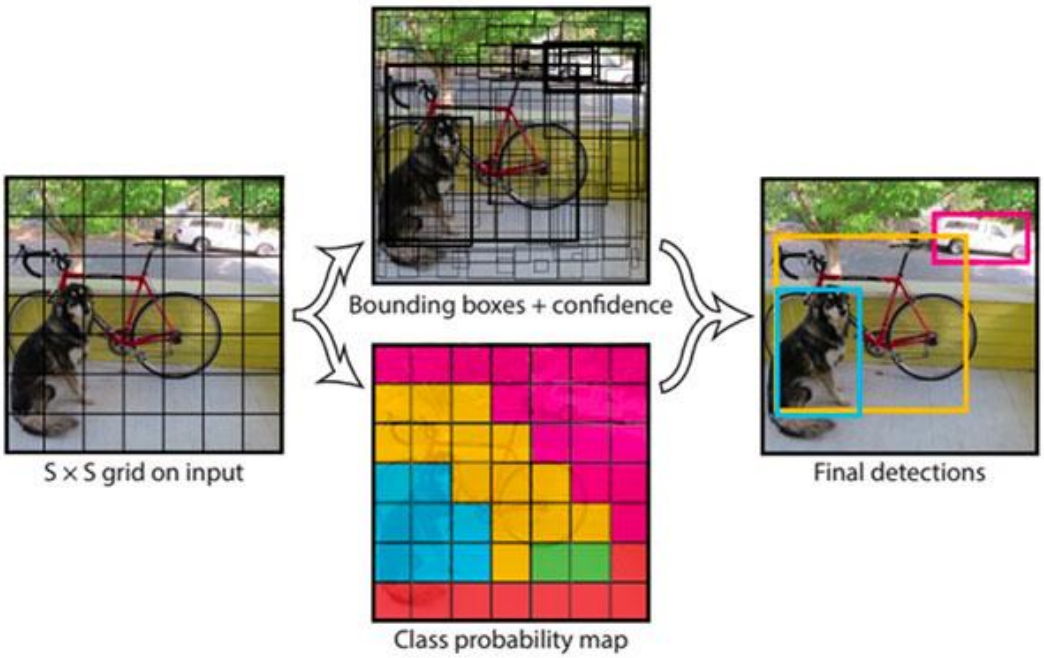
Kuvanluokituksessa kuvasta tunnistetaan mitä kuva sisältää ja kuva voidaan luokitella sisällön perusteella. Hahmontunnistuksessa tunnistetaan kuvanluokituksen lisäksi myös objektin sijainti kuvassa.

Kuvanluokitus soveltuu hyvin suuren datamäärän lajitteluun. Kuvanluokittaja voidaan esimerkiksi opettaa tunnistamaan eri koirarotuja. /11/.

Hahmontunnistusta voidaan käyttää esimerkiksi viallisten piirilevyjen tunnistamiseen. Hahmontunnistusta voidaan käyttää myös liikennemerkkien tunnistamisessa. /12/. Hahmontunnistus on tietokoneella nopeampaa ja tarkempaa, kunhan se saa riittävän koulutuksen tehtäväänsä.

2.3 YOLO

YOLOn tunnistustekniikka eroaa melkein kaikista muista siinä, että se nimensä mukaisesti, käsittelee jokaisen kuvan yhdellä kerralla. Kuva jaetaan ruudukkoon ja jokaisen muodostuneen ruudun eli solun sisälle muodostetaan viisi havaintoa (bounding boxes) ja jokaiselle havainnolle lasketaan varmuus (confidence). Jokaiselle solulle arvioidaan myös luokka eli mikä objekti havaintojen kohteena on. Lopuksi poistetaan kuvasta sellaiset havainnot, jotka eivät yllä annettuun varmuusarvoon ja NMS-funktiolla poistetaan päällekkäisistä havainnoista epävarmimmat havainnot. /13, 14/. Alla olevassa kuvassa havainnollisestaan YOLOn eri vaiheet **(Kuva 1)**.



Kuva 1. YOLOn tunnistuksen eri vaiheet /13/.

2.4 Muita yleisimpiä tunnistustekniikoita

Lähimpänä YOLOa on SSD. Se toimii hieman samalla tavalla kuin YOLO, eli käsittelemällä koko kuvan yhdellä kertaa. Ruudukkoon jakamisen sijaan se käy kuvan läpi liukuvan kehyksen avulla, objektin tunnistus ja paikannus omina tehtävinään ja yhdistetään lopuksi. /15/.

R-CNN käyttää ”selective search” -algoritmia. Algoritmi valitsee kuvasta 2000 aluetta, jotka syötetään neuroverkkoon. Neuroverkko etsii alueista piirteitä ja syöttää ne sen jälkeen SVM:lle luokitettavaksi. Uudemmat versiot R-CNN:stä ovat Fast R-CNN ja Faster R-CNN. /16/.

3 LAITTEET

Tässä luvussa tutustutaan lyhyesti Raspberry Pi:in ja NCS:än. Alla on lista työhön tarvittavista laitteista ja tarvikkeista:

Raspberry Pi 3 Model B

Raspberry Pi Camera V2 -moduuli

Movidius Neural Compute Stick (NCS)

Micro-usb-verkkovirta-adapteri (Työssä käytetty: 5,1V ja 2,5A)

32-GB:n SDHC-muistikortti ja SD-adapteri

TV (tai monitori, jossa on HDMI-portti)

PC oheistarvikkeineen

PC:n sisäänrakennettu SD-muistikorttipaikka (tai ulkoinen SD-kortin lukija)

HDMI-kaapeli

Verkkokaapeli

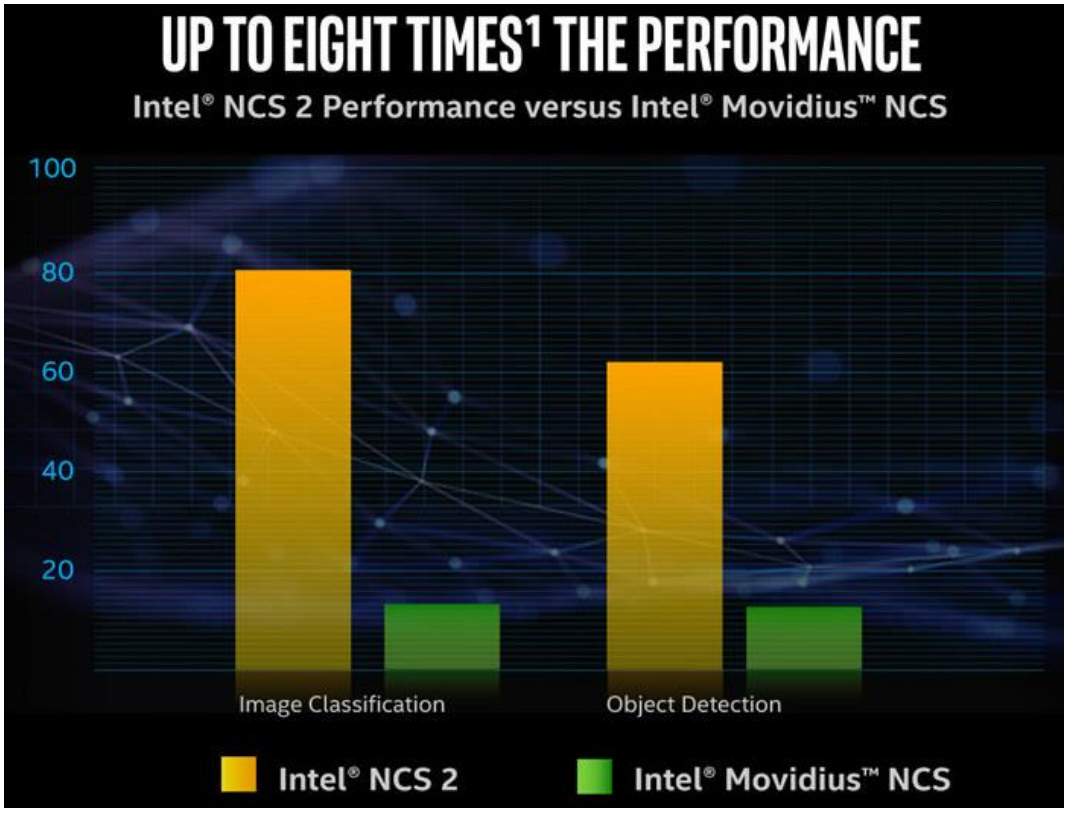
Näppäimistö ja hiiri

3.1 Raspberry Pi 3 Model B

Raspberry Pi on noin pankkikortin kokoinen yhden piirilevyn tietokone. Kokonsa ja hintansa puolesta se sopii hyvin pienen mittakaavan projekteihin. Työssä käytetyssä mallissa on neliytiminen BCM2837 64bit -prosessori, joka toimii 1.2GHz:n taajuudella. Käytettävissä on myös 1GB RAM-muistia. /17/. Käyttöjärjestelmäksi Raspberry Pi:lle valittiin Raspbian Strech. Se on yhteensopiva NCS:n kanssa ja se on Raspbian versioista uusin.

3.2 Movidius Neural Compute Stick

NCS on Intelin valmistama USB-pohjainen ison muistitikun kokoinen tekoälykiihdytin. Se saa tehonsa Myriad 2 VPU -prosessorilta ja 4GB LPDDR3 RAM-muistilta. NCS suunniteltiin tuomaan lisätehoa mm. tekoälyn laskentakuormaan. /18, 19/. Alla olevassa kuvassa vertaillaan NCS:n ja sen uudemman version NCS2:n tehoja **(Kuva 2)**.



Kuva 2. NCS2 toisi jopa 8-kertaisen tehon /20/.

4 OHJELMISTOT JA KIRJASTOT

Tässä luvussa käydään läpi työssä käytetyt ohjelmistot ja kirjastot. Perehdytään myös lyhyesti niiden merkitykseen työn kannalta.

4.1 OpenCV

OpenCV on avoimen lähdekoodin konenäkökirjasto, joka sisältää yli 2 500 konenäköön optimoitua algoritmia /21/. OpenCV on ”3-clause BSD License”-lisenssin alainen eli sitä voi vapaasti käyttää, kunhan sen mukana tulee lista tekijänoikeuksista. Kirjaston muunneltua versiota ei myöskään saa mainostaa tekijänoikeuksien haltijoiden tai edistäjien nimissä ilman etukäteen hankittua kirjallista lupaa. /22/. Tämä kirjasto mahdollistaa hahmontunnistuksen reaaliaikaisen käytön.

4.2 NCSDK

NCSDK on kehitysalusta NCS:lle. Se sisältää työkaluja, funktioita ja esimerkkejä NCS:n käyttöä varten. /23/.

4.3 OpenVino

OpenVino on konenäköä varten luotu työkalukokoelma. Se myös mahdollistaa NCS:n helpon käytön.

4.4 Etcher

Etcher on avoimen lähdekoodin ohjelmisto, jolla voidaan kirjoittaa levyn näköistiedostoja muistikorteille ja -tikuille. Sen avulla voidaan kirjoittaa Raspbian Strech -käyttöjärjestelmä SD-kortille tätä työtä varten.

4.5 Etäyhteydet

SSH on etäkäyttöohjelmisto, jolla voidaan luoda salattuja etäyhteyksiä koneelta toiselle. PuTTY on SSH- ja telnet-asiakasohjelma (client), jolla voidaan luoda SSH-

etäyhteyksiä. VNC on protokolla, jolla voidaan ottaa etäyhteys tietokoneen graafiseen käyttöliittymään.

5 PROJEKTIN TOTEUTUS

Tässä luvussa seurataan vaihe vaiheelta työn eteneminen.

5.1 Raspberry Pi:n käyttöönotto

Käyttöönotto aloitetaan SDHC-kortin alustamisella, jonka voi tehdä Windows-levynhallinta-työkalulla. Painetaan windows + r -näppäimiä ja syötetään kenttään *diskmgmt.msc*. Alustetaan muistikortti ja luodaan siihen uusi asema, jonka tiedostojärjestelmä on FAT32.

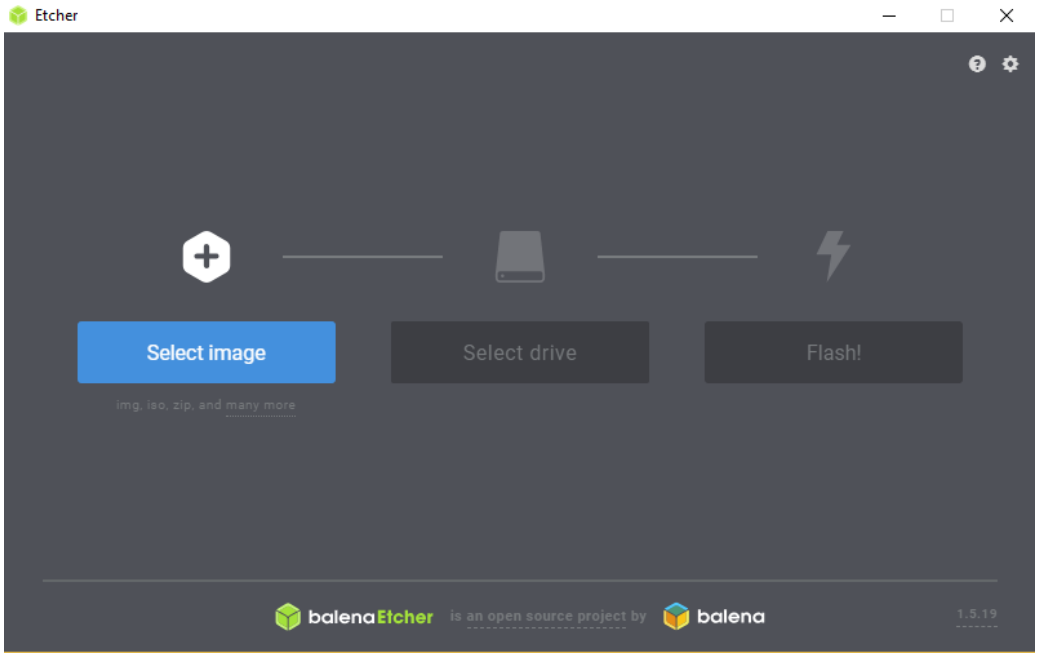
Muistikortille voidaan nyt Etcherin avulla kirjoittaa Raspbian Strech -levyn näköistiedosto (.img). Raspbian Strechin (with desktop) voi ladata osoitteesta:

https://www.raspberrypi.org/downloads/raspbian/

Etcherin voi ladata osoitteesta:

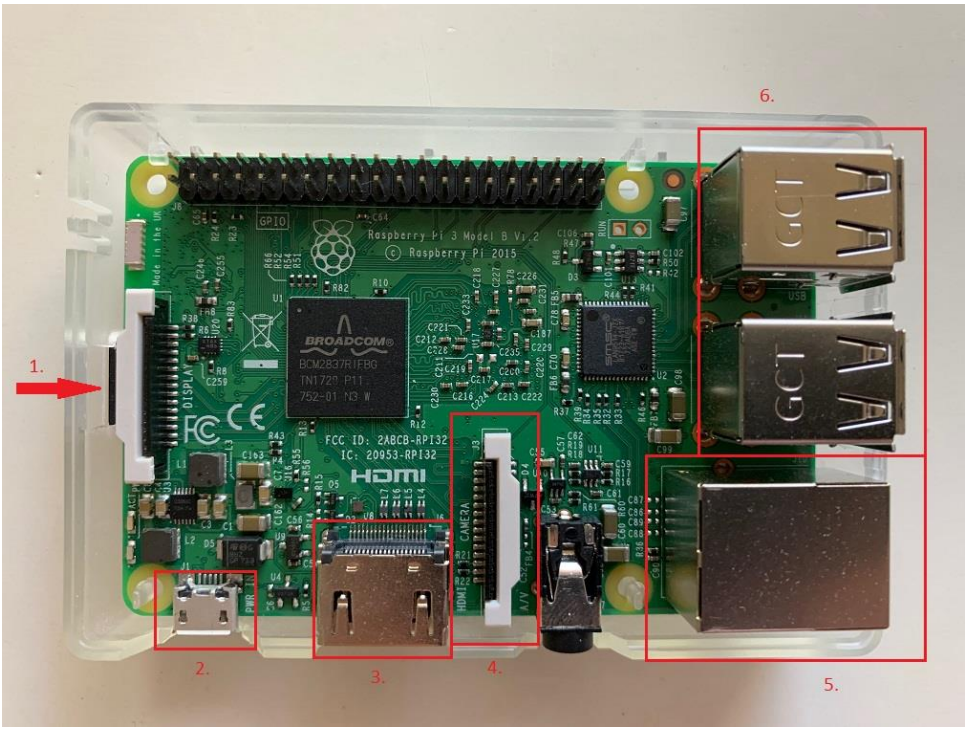
https://www.balena.io/etcher/

Käynnistetään Etcher **(Kuva 3)** ja valitaan ladattu Raspbian Strech -levyn näköistiedosto ja sen jälkeen valitaan alustettu SDHC-muistikortti ja painetaan ”Flash”. Etcherin kirjoitettua voi SDHC-muistikortin asentaa Raspberry Pi:n muistikorttipaikkaan.



Kuva 3. Etcherin käyttöliittymä.

Raspberry Pi kytketään reitittimeen RJ-45 -kaapelilla ja HDMI-kaapelilla televisioon (tai monitoriin). Kameramoduuli asennetaan camera-porttiin niin, että kaapelin sininen puoli jää ethernet-portin puolelle. Lopuksi kytketään virta. Alla olevissa kuvissa näkyy Raspberry Pi:n portit **(Kuva 4)** ja kytketty kameramoduuli **(Kuva 5)**.



Kuva 4. 1. SDHC-muistikorttipaikka (pohjassa), 2. Micro-usb portti, 3. HDMI-portti, 4. Camera-portti, 5. Ethernet-portti, 6. USB-portit

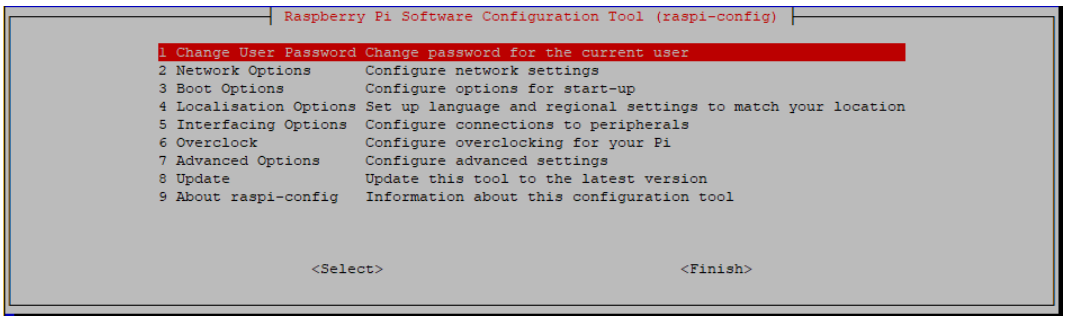
Kuva 5. Camera-porttiin asennettu kameramoduuli.

Raspberry Pi:n käynnistyttyä, työpöydällä näkyy aloitusikkuna, jossa näkyy laitteen IP-osoite. Otetaan osoite talteen ja painetaan ”Next”, jonka jälkeen valitaan lokalisaatioasetukset. Valitaan ”Finland”, jonka jälkeen kielivalinnaksi muuttuu ”Finnish” ja aikavyöhykkeeksi ”Helsinki”. Lokalisaatioasetusten jälkeen pyydetään vaihtamaan salasana. Salasanan vaihdon jälkeen seuraavassa ikkunassa voi muuttaa näytön asetuksia tarpeen vaatiessa. Lopuksi päivitetään ja käynnistetään Raspberry Pi uudelleen.

Näytön yläreunasta löytyy ”Terminaali”, jonne syötetään komento:

sudo raspi-config

Nyt voidaan muuttaa konfigurointiasetuksia käyttöönottoa varten. Alla olevassa kuvassa **(Kuva 6)** näkyy raspi-config-valikko.



Kuva 6. Raspi-config näkymä.

Konfigurointiasetuksia selataan nuolinäppäimillä ja TABilla ja valinnat tehdään Enterillä. Siirrytään valikkoon ”5 Interfacing Options” ja otetaan käyttöön ”P1 Camera”, ”P2 SSH” ja ”P3 VNC”. Lopuksi siirrytään valikkoon ”7 Advanced Options” ja valitaan ”A1 Expand Filesystem”, jonka jälkeen päänäkymässä valitaan ”<Finish>”.

VNC:n voi ladata PC:lle osoitteesta:

https://www.realvnc.com/en/connect/download/viewer/

Raspberry Pi:tä voidaan nyt käyttää SSH:n tai VNC:n kautta tai jatkaa paikallisesti. Jos jatkaa paikallisesti, täytyy NCS:ää varten hankkia usb-jatkojohto, koska NCS ei kokonsa puolesta jätä tilaa hiirelle tai näppäimistölle.

Työtä varten voidaan myös asentaa valinnaiset paketit ”screen” ja ”htop” komennolla:

sudo apt-get install screen htop

“Screen” estää SSH-istunnossa käynnistettyjen prosessien pysähtymisen vaikka SSH-istunto päättyisi, ja ”htop”:lla voi seurata prosessorin ytimien kuormitusta ja aktiivisia prosesseja.

5.2 Python-virtuaaliympäristö

Työtä varten luodaan muusta työympäristöstä eristetty virtuaaliympäristö, jolloin työtä varten ladatut kirjastot eivät sotkeudu muun järjestelmän kanssa. Tämä työvaihe pohjautuu lähteen /24/ ohjeeseen. Virtuaaliympäristön luominen aloitetaan asentamalla pip-paketinhallintatyökalu komennoilla:

wget https://bootstrap.pypa.io/get-pip.py

sudo python3 get-pip.py

jonka jälkeen pipillä asennetaan virtuaaliympäristöjen luonti- ja hallintatyökalut komennolla:

sudo pip install virtualenv virtualenvwrapper

Tämän jälkeen kirjoitetaan tiedostoon ”~/.profile” tiedostopolut virtuaaliympäristöjen käyttöä varten komennolla:

nano ~/.profile

ja lisäämällä tiedoston loppuun rivit:

 # virtualenv and virtualenvwrapper

export WORKON_HOME=$HOME/.virtualenvs

export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3

source /usr/local/bin/virtualenvwrapper.sh

jonka jälkeen virtuaaliympäristöt saadaan käyttöön komennolla:

source ~/.profile

Lopuksi luodaan itse virtuaaliympäristö komennolla:

mkvirtualenv project -p python3

jossa ”project” -virtuaaliympäristön vapaavalintainen nimi ja ”python3” on haluttu pythonversio. Näiden vaiheiden jälkeen luotu virtuaaliympäristö voidaan aktivoida komennoilla:

source ~/.profile

workon project

Alla olevassa kuvassa näkyy komennot virtuaaliympäristön aktivoimiselle ja sulkemiselle **(Kuva 7)**.

```
pi@raspberrypi:~ $ source ~/.profile
pi@raspberrypi:~ $ workon project
(project) pi@raspberrypi:~ $
(project) pi@raspberrypi:~ $ deactivate
pi@raspberrypi:~ $
```

Kuva 7. Komentosarja, jolla aktivoidaan virtuaaliympäristö ja poistutaan sieltä. Komentorivin alussa näkyy virtuaaliympäristön nimi, kun se on käytössä.

5.3 OpenCV:n asennus

Tämän työvaiheen apuna käytettiin lähteitä /24, 25/. Aluksi varmistetaan, että työskennellään kotihakemistossa komennolla:

cd ~

sekä äsken luodussa virtuaaliympäristössä tarkistamalla, että komentorivin alussa lukee virtuaaliympäristön nimi tai antamalla komennot:

source ~/.profile

workon project

Asennus aloitetaan lataamalla OpenCV:n riippuvuudet (eng. dependencies) komennoilla:

sudo apt-get install build-essential cmake pkg-config

sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev

sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev

sudo apt-get install libxvidcore-dev libx264-dev

sudo apt-get install libgtk2.0-dev libgtk-3-dev

sudo apt-get install libatlas-base-dev gfortran

ja lataamalla OpenCV:n lähdekoodi komennoilla:

wget -O opencv.zip https://github.com/opencv/opencv/archive/3.4.6.zip

wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/3.4.6.zip

jonka jälkeen pakatut kansiot puretaan komennoilla:

unzip opencv.zip

unzip opencv_contrib.zip

Hakemistojen nimet voi vaihtaa komennoilla:

mv opencv-3.4.6/ opencv

mv opencv_contrib-3.4.6/ opencv_contrib

Seuraavaksi asennetaan tarvittavia python kirjastoja komennolla:

sudo pip install numpy scipy

Kääntämistä varten siirrytään OpenCV:n purettuun kansioon, luodaan ja siirrytään ”build” -kansioon ja aloitetaan kääntämisen ensimmäinen vaihe (compile) komennoilla:

cd ~/opencv

mkdir build

cd build

*cmake -D CMAKE_BUILD_TYPE=RELEASE *

*-D CMAKE_INSTALL_PREFIX=/usr/local *

*-D INSTALL_PYTHON_EXAMPLES=ON *

*-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules *

-D BUILD_EXAMPLES=ON ..

Kääntämisen toinen vaihe (build) kestää OpenCV:n kanssa useita tunteja. Tämän vaiheen aikana kannattaa pitää erityistä huolta Raspberry Pi:n jäähdytyksestä. Komennolla ”make -j4” aloitetaan kääntämisprosessi. Optiolla ”-j4” otetaan käyttöön Raspberry Pi:n kaikki neljä ydintä, jotta kääntäminen tapahtuisi mahdollisimman nopeasti. Jos kääntäminen edellä mainitulla komennolla ei onnistu, voidaan käyttää komentoa ”make” ilman optioita.

Onnistuneen kääntämisen jälkeen OpenCV asennetaan komennolla:

sudo make install

jonka jälkeen se linkitetään koko järjestelmän käyttöön komennolla:

sudo ldconfig

Tämän jälkeen OpenCV pitää vielä linkittää virtuaaliympäristöön komennoilla:

cd ~/.virtualenvs/project/lib/python3.5/site-packages/

ln -s /usr/local/lib/python3.5/site-packages/cv2/python-3.5/cv2.cpython-35m-arm-linux-gnueabihf.so cv2.so

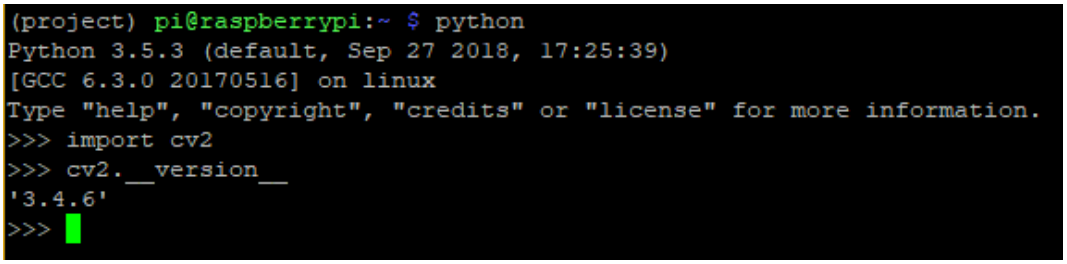
Lopuksi palataan kotihakemistoon ja testataan linkitys avaamalla python, importoimalla OpenCV:n ja antamalla version tarkistuskomennon **(Kuva 8)**:

cd ~

python

import cv2

cv2.__version__



Kuva 8. OpenCV:n version tarkistus pythonilla. Pythonista poistuminen tapahtuu komennolla "exit()"

5.4 OpenVino

Tehdään OpenVinoa varten projektikansio komennolla:

mkdir projekti2

ja siirrytään kansioon komennolla:

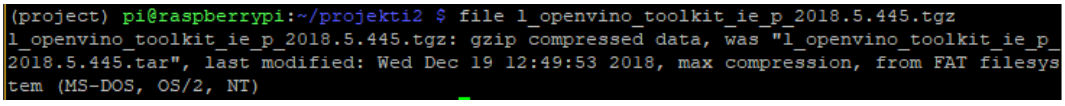
cd projekti2

OpenVino ladataan komennolla:

wget http://download.01.org/openvinotoolkit/2018_R5/packages/l_openvino_toolkit_ie_p_2018.5.445.tgz

Ladattu tiedosto on hyvä tarkistaa oikeaksi **(Kuva 9)**. Tämän jälkeen tiedoston voi purkaa komennolla:

tar -xf l_openvino_toolkit_ie_p_2018.5.445.tgz



Kuva 9. Oikea tiedosto

OpenVinolle pitää antaa asennushakemisto, joka muokataan tiedostoon ”setupvars.sh”. Tiedosto avataan muokattavaksi komennolla:

nano projekti2/inference_engine_vpu_arm/bin/setupvars.sh

Tiedosto muokataan alla olevan kuvan mukaisesti **(Kuva 10)**:

```
INSTALLDIR=/home/pi/projekti2/inference_engine_vpu_arm
export INTEL_CVSDK_DIR=$INSTALLDIR
```

Kuva 10. <INSTALLDIR> korvattu alleviivatulla hakemistopolulla.

Jotta ”setupvars.sh” aktivoituisi automaattisesti aina kun uusi terminaali avataan, pitää muokata tiedostoa ”bash.rc”. Käytetään komentoa:

nano ~/.bashrc

ja lisätään tiedoston loppuun rivit **(Kuva 11)**:

 # OpenVINO

source ~/projekti2/inference_engine_vpu_arm/bin/setupvars.sh

```
# OpenVINO
source ~/projekti2/inference_engine_vpu_arm/bin/setupvars.sh
```

Kuva 11. Bash.rc-tiedostoon lisätyt rivit.

NCS:ää varten pitää asettaa uudet usb-säännöt. Ennen sitä lisätään nykyinen käyttäjä ryhmään ”users” komennolla:

sudo usermod -a -G users "$(whoami)"

Lisäyksen jälkeen Raspberry Pi täytyy käynnistää uudelleen, jotta lisäys tulee voimaan. Raspberry Pi voidaan käynnistää uudelleen komennolla:

sudo reboot

Uudelleenkäynnistyksen jälkeen käynnistetään jälleen virtuaaliympäristö, jonka jälkeen asetetaan usb-säännöt komennolla:

sh projekti2/inference_engine_vpu_arm/install_dependencies/install_NCS_udev_rules.sh

OpenVino pitää linkittää virtuaaliympäristöön komennoilla:

cd ~/.virtualenvs/project/lib/python3.5/site-packages/

ln -s /home/pi/projekti2/inference_engine_vpu_arm/python/python3.5/cv2.cpython-35m-arm-linux-gnueabihf.so cv2.so

5.5 Lähdekoodi

Lähdekoodiksi ladataan github-käyttäjän jincongho projekti ”NCS_TinyYolo”, joka on muunneltu versio ”https://github.com/movidius/ncappzoo/tree/master/caffe/TinyYolo” esimerkkimallista. Projektin voi ladata projektihakemistoon komennoilla:

cd ~/projekti2

git clone https://github.com/jincongho/NCS_TinyYolo

Projektin käyttöä varten pitää myös ladata ja asentaa NCSDK. Se ladataan ja asennetaan komennoilla:

cd ~

git clone https://github.com/movidius/ncsdk

cd ncsdk

make install

Ennen NCS_TinyYolon käyttöä se pitää vielä kääntää komennoilla:

cd ~/projekti2/NCS_TinyYolo

make

Kääntämisen jälkeen tiedostoon ”stream.py” tehdään vielä pieniä muutoksia **(Kuva 12 ja 13)** komennolla:

nano stream.py

```
# only keep boxes with probabilities greater than this
probability_threshold = 0.20
```

Kuva 12. Tunnistuksen varmuusrajaa voi nostaa hieman, ettei tulisi liikaa vääriä tai turhia havaintoja.

```
#Open Webcam
video_capture = cv2.VideoCapture(0, cv2.CAP_V4L)
video_capture.set(cv2.CAP_PROP_FPS, 10)
```

Kuva 13. Argumentti (keltaisella rajattu) lisätty virheilmoituksen välttämiseksi. Alleviivattu arvo muutettu FPS-ongelmien takia.

Muokkausten jälkeen ohjelman voi ajaa komennoilla:

sudo modprobe bcm2835-v4l2

python stream.py

Koodin voi ajaa esimerkiksi VNC:n kautta tai paikallisesti Raspberry Pi:llä. SSH:n kautta ei voi näyttää videokuvaa. FPS-lukeman saa näkyviin videokaappauksen jälkeen lisäämällä seuraavat alla olevien kuvien **(Kuva 14-16)** mukaiset rivit tiedostoon ”stream.py”:

```
import cv2
import numpy as np
from mvnc import mvncapi as mvnc
from imutils.video import FPS
```

Kuva 14. Lisätystä kirjastosta saadaan FPS-laskuri.

```
fps = FPS().start()

#Track Webcam
while True:

    #Read Image from webcam
    ret, frame = video_capture.read()

    fps.update()
```

Kuva 15. Lisätyt rivit FPS:n laskemista varten.

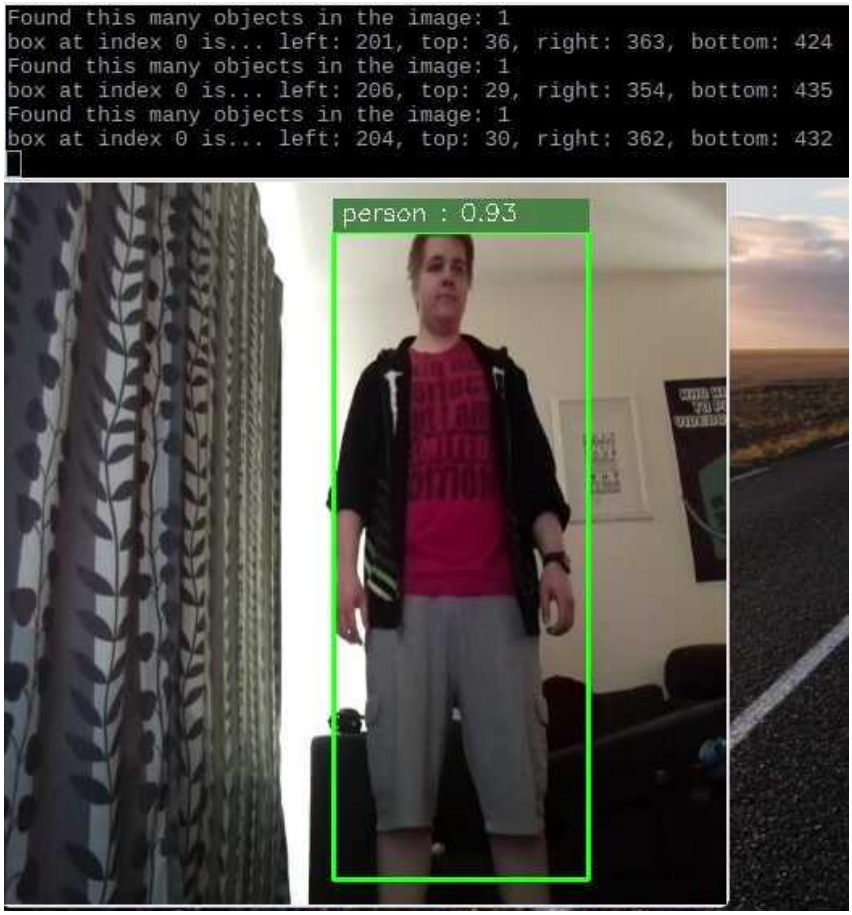
```
# When everything is done, release the capture
video_capture.release()
cv2.destroyAllWindows()

fps.stop()
print("[INFO] elasped time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))
```

Kuva 16. Lisätyt rivit FPS:n tulostamista varten.

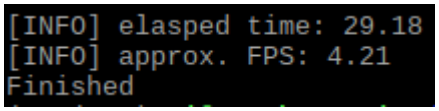
6 TULOKSET

Työvaiheiden jälkeen saatiin aikaan toimiva hahmontunnistusohjelma. Kuvan käsittelynopeudeksi saatiin hieman alle viisi kuvaa sekunnissa **(Kuva 18)**. Ohjelma on koulutettu tunnistamaan lentokone, polkupyörä, lintu, vene, pullo, linja-auto, henkilöauto, kissa, tuoli, lehmä, ruokapöytä, koira, hevonen, moottoripyörä, henkilö, ruukkukasvi, lammas, sohva, juna ja tv. Testauksessa ohjelma tunnisti henkilön **(Kuva 17)**, koiran, tuolin, ruukkukasvin ja auton.



Kuva 17. Tunnistettu kohde "person".

Työn aikana asennetut ohjelmistot ja kirjastot loivat kattavan kehitysalustan ja hyvän projektin alun, joilla voi jatkaa ohjelman kehittämistä ja hahmontunnistuksen mahdollisuuksien tutkimista.



Kuva 18. Laskettu FPS lukema.

Työhön varattu lyhyt aikataulu ei riittänyt kaikkien tavoitteiden saavuttamiseen ja asennusten aikana syntyneet virheet hidastivat työntekoa entisestään. Myös teorian opettelu on ollut hidasta haastavan aiheen vuoksi. Raspberry Pi:n ja PC:n välistä vertailua ei saatu tehtyä, eikä tekoälyn opettamiseen ehditty tutustua.

7 JOHTOPÄÄTÖKSET

Raspberry Pi:llä aikaan saatu hahmontunnistusjärjestelmä toimi hyvin rajallisella nopeudella, vähän yli 4 kuvaa sekunnissa. Tämä johtui Raspberry Pi:n rajallisista tehoista, joten täydelliseen reaaliaikaisuuteen ei päästy edes NCS:n tuomilla lisätehoilla. Reaaliaikaisuutta haittasi myös parin sekunnin viive. Tarkkaa vertailua ei voitu tehdä PC:llä testauksen puutteessa, mutta voidaan todeta, että Raspberry Pi:llä totetutetut hahmontunnistussovellukset soveltuvat lähinnä tutustumistarkoituksiin ja pieniin projekteihin.

Työ yritettiin toteuttaa eri tavoin. Yritykset Caffella ja Tensorflowlla eivät onnistuneet. Myös lukuisia github-projekteja kokeiltiin.

Työ aloitettiin liian myöhään ja sille varattu aika ei riittänyt kaikkien tavoitteiden tekemiseen. Iso osa ajasta kului lähinnä demoympäristön luomiseen, konfiguroimiseen, testaamiseen ja perustason teorian kirjoittamiseen.

Parannusehdotuksina työlle olisi tuottaa puuttuvat demot PC:llä, jotta voitaisiin arvioida Raspberry Pi:n mahdollisuuksia hahmontunnistuksessa sekä riittävän pitkän aikataulun varaaminen.

LÄHTEET

/1/ Skycode Oy. Mitä tekoäly on?. Viitattu 6.5.2019. https://tekoäly.info/mita_tekoaly_on/

/2/ Schulterbraucks, L. 2018. A Short History of Artificial Intelligence. Viitattu 6.5.2019. https://dev.to/lschultebraucks/a-short-history-of-artificial-intelligence-7hm

/3/ Dhruv, S. 2018. AI, Machine Learning, & Deep Learning Explained in 5 Minutes. Viitattu 6.5.2019. https://becominghuman.ai/ai-machine-learning-deep-learning-explained-in-5-minutes-b88b6ee65846

/4/ Moor, J. 2006. The Dartmouth College Artificial Intelligence Conference: The Next Fifty Years. *AI Magazine*. 27, 4, 87-91. Viitattu 7.5.2019. https://pdfs.semanticscholar.org/d486/9863b5da0fa4ff5707fa972c6e1dc92474f6.pdf

/5/ Adams, R.L. 2017. 10 Powerful Examples Of Artificial Intelligence In Use Today. Viitattu 7.5.2019. https://www.forbes.com/sites/robertadams/2017/01/10/10-powerful-examples-of-artificial-intelligence-in-use-today/#1512cc1b420d

/6/ Lison, P. 2012. An introduction to machine learning. Viitattu 15.5.2019. http://folk.uio.no/plison/pdfs/talks/machinelearning.pdf

/7/ Expert System S.p.A. What is Machine Learning? A definition Viitattu 15.5.2019. https://www.expertsystem.com/machine-learning-definition/

/8/ Brownlee, J. 2016. Supervised and Unsupervised Machine Learning Algorithms. Viitattu 15.5.2019. https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

/9/ Osiński B & Budek, K. 2018. What is reinforcement learning? The complete guide. Viitattu 16.5.2019. https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/

/10/ Skymind Inc. A Beginner's Guide to Neural Networks and Deep Learning. Viitattu 16.5.2019. https://skymind.ai/wiki/neural-network

/11/ Le, J. 2019. How to easily build a Dog breed Image classification model.Viitattu 20.5.2019. https://medium.com/nanonets/how-to-easily-build-a-dog-breed-image-classification-model-2fd214419cde

/12/ Hulstaert, L. 2018. A Beginner's Guide to Object Detection. Viitattu 20.5.2019. https://www.datacamp.com/community/tutorials/object-detection-guide

/13/ Redmon, J, Divvala, S, Girshick, R & Farhadi A. 2016. You Only Look Once: Unified, Real-Time Object Detection. Viitattu 17.5.2019. https://pjreddie.com/media/files/papers/yolo_1.pdf

/14/ Maj, M. 2018. What is object detection? Introduction to YOLO algorithm. Viitattu 17.5.2019. https://appsilon.com/object-detection-yolo-algorithm/

/15/ Li, C. 2019. Tips for implementing SSD Object Detection (with TensorFlow code). Viitattu 21.5.2019. https://lambdalabs.com/blog/how-to-implement-ssd-object-detection-in-tensorflow/

/16/ Gandhi, R. 2018. R-CNN, Fast R-CNN, Faster R-CNN, YOLO—Object Detection Algorithms. Viitattu 21.5.2019. https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e

/17/ Raspberry Pi:n kotisivut. Viitattu 20.5.2019. https://www.raspberrypi.org/blog/product_categories/raspberry-pi-boards/

/18/ Intel newsroom. 2017. Intel Democratizes Deep Learning Application Development with Launch of Movidius Neural Compute Stick. Viitattu 14.5.2019. https://newsroom.intel.com/news/intel-democratizes-deep-learning-application-development-launch-movidius-neural-compute-stick/#gs.bckwws

/19/ Oh, N. 2017. Intel Launches Movidius Neural Compute Stick: Deep Learning and AI on a $79 USB Stick. Viitattu 21.5.2019. https://www.anandtech.com/show/11649/intel-launches-movidius-neural-compute-stick

/20/ NCS1:n ja NCS2:n tehojen vertailutaulukko. Viitattu 20.5.2019. https://www.cnx-software.com/wp-content/uploads/2018/11/Intel-NCS2-vsIntel-Movidius-NCS.jpg

/21/ OpenCV:n verkkosivut. Viitattu 17.5.2019. https://opencv.org/about/

/22/ OpenCV:n lisenssi. Viitattu 21.5.2019. https://opencv.org/license/

/23/ Intelin Movidiuksen NCSDK github-projekti. Viitattu 17.5.2019. https://github.com/movidius/ncsdk

/24/ OpenCV:n ja python-virtuaaliympäristön asennusohje. Viitattu 1.5.2019. https://www.pyimagesearch.com/2018/09/26/install-opencv-4-on-your-raspberry-pi/

/25/ OpenCV:n asennusohje. Viitattu 1.5.2019. https://www.alatortsev.com/2018/09/05/installing-opencv-3-4-3-on-raspberry-pi-3-b/