Mikko Antell

JAI GO-5000M-PGE- ja The Imaging Source DFK MKU130-10x22 -kameroiden käyttöönotto

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Automaatiotekniikka

Insinöörityö

10.5.2015

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Mikko Antell JAI GO-5000M-PGE- ja The Imaging Source DFK MKU130-10x22 -kameroiden käyttöönotto 35 sivua + 2 liitettä 10.5.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Automaatiotekniikka
Suuntautumisvaihtoehto	Kappaletavara-automaatio
Ohjaaja(t)	Jari Savolainen, Osaamisaluepäällikkö Timo-Pekka Jäntti, Lehtori

Insinöörityön tavoitteena oli kahden erilaisen konenäkökameran käyttöönotto Metropolia Ammattikorkeakoulun automaatiolaboratoriossa ja ohjeiden kirjoittaminen käyttöönoton toistamista varten.

Työ suoritettiin Metropolia Ammattikorkeakoulun Leiritien toimipisteen automaatiolaboratoriossa, jonne työlle oli varattu oma tietokone, johon kaikki tarvittavat ohjelmistot pystyttiin asentamaan.

The Imaging Sourcen mikroskooppikameran käyttöönotto todettiin helpommaksi, joka johtui USB-liitännästä ja kameran helposta asentamisesta. JAI:n kameran käyttöönotto todettiin haastavammaksi vioittuneen XML-tiedoston ja GigE-kytkennän takia.

OpenCV-kirjastojen toimintaan saaminen Windows-ympäristössä todettiin monimutkaiseksi, kun taas MATLAB-ohjelman toimiminen kameroilla onnistui helposti. Molemmat kamerat saatiin toimimaan OpenCV:n kirjastoilla VisualStudio-ohjelmassa sekä MATLAB-ohjelman Image Acquisition Toolboxilla sekä Computer Vision Toolboxilla.

Kameroiden on tarkoitus olla jatkossa koulutuskäytössä konenäkökameroihin liittyvällä laboratoriokurssilla.

Avainsanat	Konenäkökamera, MATLAB, OpenCV

Author(s) Title Number of Pages Date	Mikko Antell Commissioning of JAI GO-5000M-PGE- and The Imaging Source DFK MKU130-10x22 -cameras 35 pages + 2 appendices 10 May 2015
Degree	Bachelor of Engineering (AMK)
Degree Programme	Automation Engineering (AMK)
Specialisation option	Manufacturing Automation
Instructor(s)	Jari Savolainen, Head of Department Timo-Pekka Jäntti, Senior Lecturer

The goal of this Bachelor's thesis was commissioning of two different kinds of computer vision cameras at the Metropolia University of Applied Sciences automation laboratory and writing a guide of the installation for future utilization.

The Bachelor's thesis was done at the Metropolia UAS automation laboratory where there was a working station reserved for the installation of the cameras. Computer with administrator rights was given to the project for installation of required software.

The Imaging Sources DFK MKU130-10x22-camera proved to be less complicated to install due to USB-cable's plug and play feature and the software's simple installation. JAI GO-5000M-PGE-camera proved to be more difficult to install due to corrupted XML-file inside the camera and the GigE-connection.

Implementation of OpenCV in Windows proved to be troublesome whereas the installation of MATLAB was uncomplicated. Both of the cameras were successfully used with OpenCV and MATLAB's Image Acquisition Toolbox and Computer Vision Toolbox.

The cameras are planned to be part of the computer vision education for Metropolia University of Applied Sciences.

Keywords	Computer vision, OpenCV, MATLAB

Sisällys

Liitteet

Liite 1. GMHR47518MCN-1–linssillä otettu kuva

Liite 2. GMTHR32514MCN–linssillä otettu kuva

Lyhenteet

PoE	Power over Ethernet. Standardi, jonka avulla erilaisten laitteiden virransyöttö voidaan tuoda Ethernet-kaapelin kautta.
GigE	Gigabit Ethernet. Ethernet-yhteys, jonka nopeus on 1000 Mbit/s.
USB3	Universal Serial Bus version 3.
CMOS	Complementary Metal Oxide Semiconductor. Puolijohdekomponentti, jonka tekniikan pohjalta on alettu valmistamaan kennoja kameroille.
XML	Tiedosto kameran sisällä, joka sisältää kameran tiedot ja asetukset.
BSD	Avoimeen lähdekoodiin perustuva lisenssi, jossa on hyvin vähän rajoituksia.
OpenCV	Open Source Computer Vision. Avoimeen lähdekoodiin perustuva konenäkökirjasto.
DLL	Dynamic-link library. Windowsin käyttämä ohjelmien kirjastojen käyttötapa.

1 Johdanto

1.1 Työn tausta

Työ suoritettiin Metropolia Ammattikorkeakoululle. Metropolia Ammattikorkeakoululle oli tilattu uusia konenäkökameroita, jotka saapuivat vuoden 2014 loppupuolella. Uusien kameroiden tarkoitus oli laajentaa laboratoriokäytössä olevien kameroiden määrää ja näin mahdollistaa oppilaiden saavan parempaa käsitystä uusista konenäkökameroista. Insinöörityö suoritettiin vuoden 2015 helmikuun ja huhtikuun välisenä aikana.

1.2 Työn tavoitteet ja toteutus

Insinöörityön tarkoituksena oli käyttöönottaa JAI:n GO-5000M-PGE-konenäkökamera ja The Imaging Sourcen DFK MKU130-10x22 -mikroskooppikamera. Työn tavoitteena oli saada kamerat toimimaan Windows- ja Linux-käyttöjärjestelmillä OpenCV-kirjastojen sekä MATLAB-ohjelman kanssa.

Käytettyinä ohjelmina olivat VisualStudio 2013, jossa käytettiin avoimeen lähdekoodiin perustuvaa OpenCV-kirjastoja sekä MATLAB-ohjelmaa ja sen kuvienkäsittelyyn tarkoitettuja työkalupaketteja, joita olivat Image Contor Toolbox, Image Acquasition Toolbox ja Computer Vision System.

Insinöörityö toteutettiin Metropolia Ammattikorkeakoulun Leiritien toimipisteen automaatiolaboratoriossa, jossa työlle oli varattu yksi tietokone, jolle tarvittavat ohjelmat oli tarkoitus asentaa. Työhön kuului ohjelmien asennuksia ja testauksia sekä kameroiden kytkentää ja C++-koodin tuottamista.

2 Laitteisto

2.1 The Imaging Source DFK MKU130-10x22

The Imaging Sourcen DFK MKU130-10x22 on USB 3.0 -liitännällä varustettu mikroskooppikamera (ks. kuva 1). Kamerassa on kierteet, jotka sopivat tietyn tyyppisten mikroskooppien kanssa yhteen. Näin ollen kamera saadaan kiinnitettyä kiinni mikroskooppiin, jolloin mikroskoopin kohteesta saadaan tuotua kuvaa näytölle ja kuvat voidaan myös tallettaa tietokoneelle. Kamera on tarkoitettu käytettäväksi mikroskoopin kanssa, joten se ei tarkenna kohteisiin hyvin ilman mikroskoopin linssiä. Koska kamera on tarkoitettu kiinnitettäväksi mikroskooppiin, se ei tarvitse erillistä jalustaa käytössä. (1.)

Kamerassa on CMOS-kenno ja se voi tuottaa minimissään 640x480 pikselin tarkkoja kuvia ja maksimissaan 4128x3096 pikselin tarkkoja kuvia. Kuvien tarkkuus vaikuttaa kameran kuvanpäivitysnopeuteen. Kameran liitäntä tietokoneeseen tapahtuu USB 3.0 -tyypin A urosliittimellä. Kamera vaatii toimiakseen tietokoneelta USB 3.0 –tyypin A naarasliittimen. Koska kamera toimii USB 3.0 -liittimen kautta, ei se tarvitse erillistä virtalähdettä, vaan se ottaa tarvitsemansa virran USB 3.0 -liittimen kautta. Tämä helpottaa asennuksissa ja säästää kaapelointikustannuksissa. USB 3.0 -standardin nopeus mahdollistaa tarkemmat kuvat kuin vanhemman USB 2.0 -standardin konenäkökamerat. (1.)

Kuva 1. DFK MKU130-10x22 –kamera.

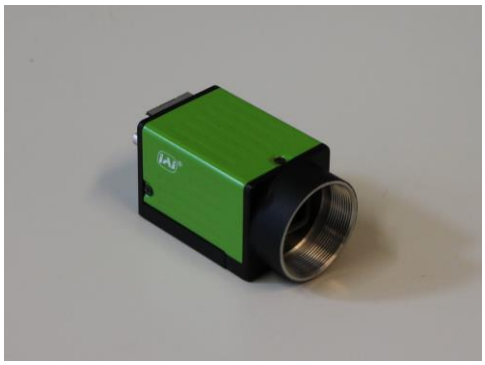
2.2 JAI GO-5000M-PGE

JAI GO-5000M-PGE on CMOS-kennolla varustettu 5 megapikselin kamera (ks. kuva 2). Kameran nopein mahdollinen nauhoitusnopeus on 22 kuvaa sekunnissa. Kameran kytkentöinä toimii Ethernet-kaapelin RJ45-liitin, ja kamera on varustettu PoE-ominaisuudella. Mikäli saatavilla ei ole PoE-tukevaa tietokonetta tai kytkintä/reititintä, on kamerassa myös Hirose-liitin, jonka kautta kameralle voidaan syöttää 12 V:n syöttöjännite. Hirose-liittimellä voidaan kameralle tehdä myös fyysinen laukaisin. GO-5000M-PGE on kooltaan 29,0 x 29,0 x 41,5 mm ja sen ulostulot ovat 8/10/12-bittisiä. Kameran virrankulutus on 240 mA. (2, s. 1-2.)

Kamera on monokromaattinen, eli se tuottaa kuvia, jotka ovat harmaasävykuvia. Kameran monokromaattisuus poistaa värien mahdollisuuden kuvista, mutta se myös nopeuttaa kameran toimintaa. Kamera tukee C-tyypin linssejä. (2, s. 1-2.)

Kamera voidaan kytkeä suoraan tietokoneeseen Ethernet-kaapelilla. Kamera voidaan kytkeä myös lähiverkkoon, jolloin kameraa ei tarvitse kytkeä suoraan tietokoneeseen. Kamera on mahdollista saada toimimaan myös langattomasti, mikäli kamera kytketään kytkimeen, jossa on langaton lähetin. (2, s. 1-2; 3 s.14; 4.)

Kamera tukee GigE Vision 2.0 -standardia ja se toimii parhaiten verkkokorttien kanssa, jotka tukevat GigE-nopeuksia (5, s.6). Kameran PoE-ominaisuus yhdistettynä GigE Visioniin mahdollistaa sen, että monia kameroita voidaan kytkeä yhteen tietokoneeseen pelkän Ethernet-kaapelin ja PoE-syötöllä varustettujen kytkimien tai reitittimien avulla. Suositeltava kaapeli on suoraankytketty vähintään CAT 5e Ethernet -kaapeli, joka tukee GigE-nopeutta (5, s.14).



Kuva 2. GO-5000M-PGE-kamera.

2.3 MOXA EDS-G205A-4PoE

MOXA EDS-G205A-4PoE on PoE-ominaisuudella varustettu 5-porttinen Ethernet-kytkin (ks. kuva 3), joka tukee kaikissa porteissa GigE:iä. Kytkimessä on yksi portti tietokoneeseen kytkemistä varten ja neljä porttia PoE-laitteita varten. Virransyöttö tapahtuu 24 voltilla tai 48 voltilla. Laitteessa on kaksi virransyöttöpaikkaa. Mikäli toiseen virransyöttöön tulisi jokin vika, niin toinen virransyöttöpiste alkaisi toimia saman tien, jolloin kytkimeen kytketyt PoE-laitteet eivät vaikuttuisi virrankatkoksista. (5, s. 9; 6, s.1-2.)

Kytkimessä on Bradcast strom protection, joka saadaan käyttöön DIP-kytkimillä, jotka ovat laitteessa kiinni. Kytkimessä on DIN-kiskokiinnike suoraan, joten se voidaan kytkeä helposti DIN-kiskoon. Kytkin pystyy antamaan jokaiselle PoE-portille enintään 30 wattia tehoa ja se tukee 9,6 kilobitin jumbo frameja. (6, s.1-2.)

Jumbo frame viittaa Ethernet-kommunikointipaketin tietosisällön kokoon. Normaalisti tietosisällön koko paketissa on 1500 tavua, mutta jumbo frame mahdollistaa jopa 9000 tavun tietosisällön koon kommunikointipaketissa. Tietosisällön koon kasvattaminen tarkoittaa sitä, että yhden paketin mukana pystytään toimittamaan enemmän tietoa tietokoneen ja laitteen välillä. Tämä ominaisuus on tärkeä kamerasovelluksissa, sillä kameroiden lähettämä korkearesoluutioinen data vie paljon tilaa. (8.)

Kuva 3. EDS-G205A-4PoE-kytkin.

2.4 Muut laitteet

2.4.1 Linssit

JAI:n kameraan kytkettäviä linssejä oli kaksi. Molemmat linssit ovat GOYO OPTICAL Inc:n valmistamia C-tyypin kierteellä varustettuja linssejä.

GMHR47518MCN-1–linssi (ks. kuva 4) on yksituumainen korkean resoluution linssi, jonka polttoväli on 75 mm. Valotusaukko on F1.8–16 ja kuvakulma on 9,7° x 7,3° x 12,0°. (9.)

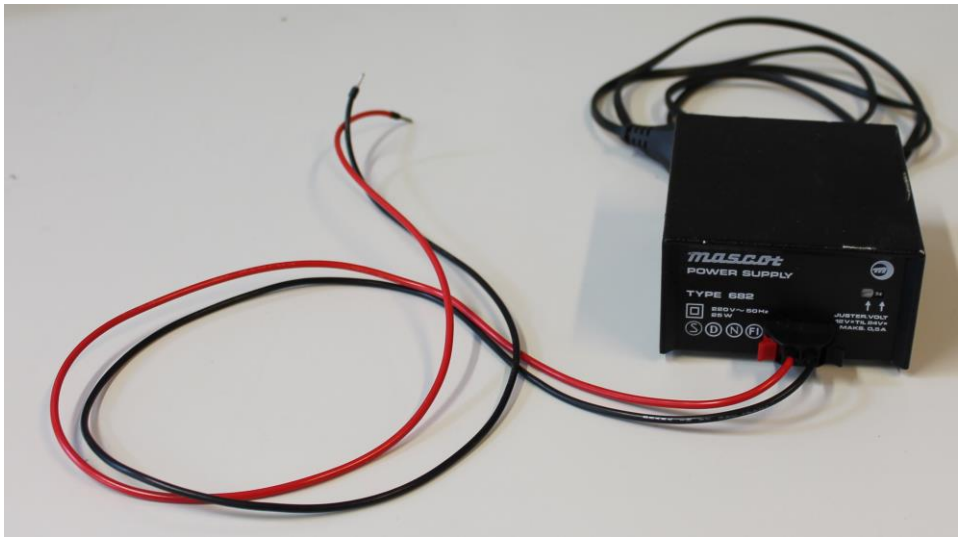
Kuva 4. GMHR47518MCN-1–linssi.

GMTHR32514MCN–linssi (ks. kuva 5) on korkean resoluution linssi, jonka polttoväli on 25 mm, valotusaukko on F1.4–16 ja kuvakulmat ovat 14,6° x 11,0° x 18,2° (1/2”) ja 21,1° x 15,1° x 25,0° (2/3”). (10.)

Kuva 5. GMTHR32514MCN–linssi.

2.4.2 Virtalähde

Virtalähteenä toimi Mascot TYPE 682 (ks. kuva 6), joka muuntaa 230 V:n vaihtovirtaa halutuksi tasavirraksi. Työssä käytettiin 24 V:n tasavirtaa, joka syötettiin MOXA:n Ethernet-kytkimelle, joka antoi virtaa PoE:n välityksellä JAI:n kameralle. Virtalähteen tuottama enimmäisvirta oli 500 mA.



Kuva 6. Mascot TYPE 682-virtalähde.

2.4.3 Tietokone ja verkkokortti

Tietokoneena toimi 64-bittinen Windows 8:lla varustettu tietokone. Tietokoneeseen hankittiin erillinen verkkokortti, Intel 1000/PRO, jota suositellaan käytettäväksi GigE Vision -kameroiden kanssa (5; s.13). Toinen verkkokortti tietokoneessa helpottaa myös koneen käyttöä, koska tällöin internetiä pystyy myös käyttämään toisen verkkokortin kautta. Intel 1000/PRO tukee myös jumbo frameja, jota mahdollistavat kytkettyjen laitteiden tavallista suurempien pakettien lähettämisen tietokoneelle.

2.4.4 Kaapelit

Ethernet-kaapeleina käytettiin yhden metrin pituisia suoria CAT 6 –kaapeleita, jotka tukivat GigE:iä.

3 Ohjelmat

3.1 OpenCV

OpenCV on BSD-lisenssiin perustuva avoimen lähdekoodin konenäkö- ja koneoppimiskirjasto, jonka nimi tulee sanoista Open source computer vision library. Kirjastossa on yli 2500 erilaista algoritmia, joiden avulla voidaan tehdä monia erilaisia toimintoja. Algoritmeja voidaan käyttää muun muassa silmien liikkeiden seurantaan sekä kasvojen ja maiseman tunnistukseen. (11.)

Kirjastossa on C++-, C-, Python-, Java- ja MATLAB-käyttöliittymät, joten se tukee monia eri ohjelmointikieliä. OpenCV tukee myös monia eri käyttöjärjestelmiä, kuten Windowsia, Linuxia, Mac OS:ää ja Androidia. (11.)

OpenCV:llä on hyvin laaja käyttäjäyhteisö, josta voi pyytää apua ja hakea esimerkkikoodeja. OpenCV:n käytöstä ja asennuksesta on myös tehty ohjeita, joita voi käyttää apuna.

3.2 Visual Studio 2013

Visual Studio 2013 on Microsoftin tekemä ohjelmistonkehitysympäristö, jonka avulla voidaan käyttää monia eri ohjelmointikieliä (12). Tässä insinöörityössä käytettiin C++-ohjelmointikieltä. Visual Studio 2013 ei itsessään toimi konenäköohjelmana, vaan siihen täytyy lisätä erilaisia kirjastoja, jotta konenäköön tarkoitettuja toimintoja voitaisiin suorittaa. Tässä työssä käytettiin avoimeen lähdekoodiin perustuvaa OpenCV-kirjastoa, joka on kehitetty konenäkösovelluksille.

Visual Studio 2013 -ohjelmaan liitettiin OpenCV-kirjastot, joiden avulla molempien kameroiden käyttö tuli mahdolliseksi Visual Studio 2013 -ohjelmistonkehitysympäristössä. Tarkoituksena oli testata OpenCV:n käyttöä Windowsissa ja tämä onnistui Visual Studio 2013 -ohjelmistonkehitysympäristössä.

3.3 MATLAB

MATLAB on Mathworks-yrityksen kehittämä ohjelma, joka perustuu numeeriseen laskentaan ja omaan ohjelmointikieleensä. MATLAB:iin pystytään asentamaan monia erilaisia lisäosia moniin eri tarkoituksiin, joista yksi on kuvankäsittely ja konenäkösovellukset. (13.)

Tässä työssä käytettäviin kuvankäsittely- ja konenäkötyökalulaatikkoihin kuuluivat Image Processing Toolbox, Image Acquisition Toolbox ja Computer Vision System Toolbox.

Image Processing Toolboxiin kuuluu kuvien erilainen analysointi, johon kuuluu esimerkiksi segmentointi, suodatukset, kuvan muunnokset ja kuvien ja videoiden esittämiseen tarkoitetut Image Viewer ja Video Viewer. (14.)

Image Acquisition Toolboxin tarkoituksena on MATLAB:in kautta ohjata kameran asetuksia ja toimintaa. Image Acquisition Toolbox tukee monia eri teollisuuden standardeja, joihin lukeutuvat DCAM, GigE Vision ja Camera Link. Ohjelman avulla käyttäjä pystyy muuttamaan kameran asetuksissa esimerkiksi resoluutiota, värikylläisyyttä ja tallennettavan kuvan muotoa. Tallennettuja kuvia voidaan käyttää suoraan MATLAB:in muissa ohjelmissa, kuten esimerkiksi Simulinkissä, jonka avulla voidaan luoda videokuvaa analysoivaa ohjelmaa. (15.)

Computer Vision System Toolboxiin kuuluu erilaisia algoritmeja, jotka mahdollistavat esimerkiksi kohteen tunnistuksen, kohteen ja muotojen seurauksen ja kameran kalibroinnin. (16.)

Tässä työssä käytettiin eniten Image Acquisition Toolboxia, jonka avulla varmistettiin, että kamerat saatiin kytkettyä MATLAB:iin. Jotta kamerat saatiin kytkettyä, joutui MATLAB:iin asentamaan vielä adaptereita, jotka tukivat käytössä olleiden kameroiden kommunikointimenetelmiä. Toinen adaptereista löytyi suoraan MATLAB:in omista adaptereista ja toinen löytyi The Imaging Source -kameran valmistajan sivuilta.

3.4 JAI Control Tool

JAI Control Tool on ohjelma, joka on tarkoitettu toimimaan monien eri teollisuuden kommunikointistandardien kanssa. Ohjelma on kehitetty JAI:n kameroille, mutta sillä pystytään käyttämään myös muiden valmistajien kameroita, mikäli ne tukevat ohjelman käyttämiä kommunikointistandardeja. Standardeihin kuuluu GenTL, GigE Vision ja GenICam (3, s.4). Ohjelman avulla pystytään ottamaan yhteys kameraan, ja kameran asetuksia pystytään muuttamaan hyvinkin laajasti. Ohjelmalla pystytään myös tallentamaan kameralta saatuja kuvia.

Tässä työssä ohjelmaa käytettiin JAI GO-5000M-PGE -kameran toiminnan testaukseen ja varmistukseen. Ohjelman avulla yritettiin myös selvittää, minkä takia kamera ei toiminut, vaikka JAI Control Tool havaitsi kameran.

3.5 IC Full Screen Presenter

IC Full Screen Presenter on The Imaging Sourcen kameroille tehty ohjelma, joka on tarkoitettu kameroiden kokeiluun ja toiminnan varmistamiseen. Ohjelmassa pystytään muuttamaan esimerkiksi kameran resoluutiota. Tässä työssä ohjelmalla tarkistettiin, että The Imaging Sourcen kamera DFK MKU130-10x22 toimi ja oli käytettävissä.

3.6 JAI Camera Update Tool

JAI Camera Update Tool on tarkoitettu GenICam-standardin omaavien laitteiden päivitykseen. Ohjelma etsii tietokoneeseen kytkettyjä kameroita ja kameroihin asennettuja tiedostoja ja tarkistaa, ovatko tiedostot ajan tasalla. Tässä työssä ohjelmaa käytettiin JAI GO-5000M-PGE -kameran XML-tiedoston päivittämiseen, jonka takia kamera ei toiminut. Tiedostojen päivityksen jälkeen kamera alkoi toimia.

4 OpenCV-kirjastojen liittäminen Visual Studio 2013:sta

4.1 Tarvittavat ohjelmat

OpenCV:n käytössä tarvitaan jokin ohjelmistokehitysympäristö, joita ovat esimerkiksi Visual Studio ja QT. Tässä työssä käytettiin Visual Studiota. Ohjelma, jota tarvitaan, on Path Editor, jonka avulla tietokoneeseen lisätään uusi DLL-kirjasto (17). Tärkein tarvittava asia on itse OpenCV-kirjastot, jotka voidaan ladata pakattuna tiedostona Sourceforge.net-sivuilta (18). Mikäli asennuksessa ei halua käyttää esiasennettuja kirjastoja, on vielä joitain ohjelmia, mitä tarvitaan, mutta niitä ei käytetty tässä työssä.

4.2 Kirjastojen asennusvaihtoehdot

OpenCV:tä asennettaessa on kaksi vaihtoehtoa: joko käytetään esiasennettuja kirjastoja tai luodaan kirjastot itse lähdekoodista. Esiasennettujen kirjastojen asentaminen on helpompi ja nopeampi vaihtoehto ja myös vaihtoehto, jota käytettiin tässä insinöörityössä. Kirjastot voidaan tehdä myös itse lähdekoodista, mutta tämä vie enemmän aikaa. Omien kirjastojen luomisessa etuna on vain tarvittavien kirjastojen käyttö, eikä tällöin turhia kirjastoja tule mukaan. Tässä työssä käytettiin esiasennettuja kirjastoja. (19.)

4.3 OpenCV:n lataaminen ja purkaminen

OpenCV:n asennus aloitettiin lataamalla Sourceforge.net-sivuilta haluttu versio OpenCV:stä (18). Tiedosto tuli EXE-tiedostona, joka piti asentaa haluttuun paikkaan. Tiedostot asennettiin helposti saataviksi, sillä tiedostopolkua tarvitsi useasti, kun kirjastoja lisättiin Visual Studioon. Asentaminen täytyi suorittaa järjestelmänvalvojan oikeuksilla. Sourceforgen sivuilta löytyi monia eri versioita OpenCV:stä, joista tässä työssä käytettiin versiota 2.4.10 (18).

4.4 OpenCV:n ympäristömuuttujat

Seuraava vaihe asennuksessa oli OpenCV:n ympäristömuuttujien asettaminen ja niiden lisääminen järjestelmän polkuun. OpenCV:n ympäristömuuttujien asettaminen tapahtui Windowsin komentorivillä. Komentorivi käynnistettiin järjestelmänvalvojan oikeuksilla. Tämä onnistui kirjoittamalla Windowsin Käynnistä-valikon hakupalkkiin *cmd* ja klikkaamalla avautuvaa komentorivin ikonia hiiren oikealla, jolloin vaihtoehdoksi tuli 'Suorita järjestelmänvalvojana'. Avautuvaan komentoriviin kirjoitettiin koodi (ks. esimerkkikoodi 1), joka riippui käytettävästä Visual Studion versiosta ja käyttöjärjestelmästä.

```
setx -m OPENCV_DIR D:\OpenCV\Build\x86\vc11       (Visual  Studio
2012, 32 bit. Windows)
setx -m OPENCV_DIR D:\OpenCV\Build\x64\vc11       (Visual  Studio
2012, 64 bit. Windows)
setx -m OPENCV_DIR D:\OpenCV\Build\x86\vc12       (Visual  Studio
2013, 32 bit. Windows)
setx -m OPENCV_DIR D:\OpenCV\Build\x64\vc12       (Visual  Studio
2013, 64 bit. Windows)
```

Esimerkkikoodi 1. Ympäristömuuttujan koodivaihtoehdot.

Koodin komentojen jälkeen lisättiin polku, jonne OpenCV on asennettu. Käyttöjärjestelmä määräsi, valitaanko kansioksi x86 vai x64. Visual Studion versio määräsi, valitaanko viimeiseksi kansioksi vc10, vc11 vai vc12. Tässä työssä käytettiin 64-bittistä Windows-käyttöjärjestelmää ja Visual Studion 2013 versiota (ks. esimerkkikoodi 2).

```
setx -m OPENCV_DIR D:\OpenCV\Build\x64\vc12.
```

Esimerkkikoodi 2. Ympäristömuuttujan koodi.

4.5 Staattiset ja dynaamiset kirjastot

Seuraava vaihe asennuksessa oli valinta staattisten ja dynaamisten linkkikirjastojen (DLL) välillä. Staattisissa kirjastoissa ei tarvitse tehdä muita lisäyksiä järjestelmään, sillä staattisissa kirjastoissa kirjaston tiedot ovat ohjelmassa. Dynaamisissa kirjastoissa käyttöjärjestelmä kutsuu tarvittavia kirjastoja vain silloin, kun niitä tarvitaan. (20.)

Staattisten kirjastojen hyötyjä ovat esimerkiksi se, että ohjelma tietää aina, että tarvittavat kirjastot ja niiden oikeat versiot ovat olemassa. Toisena etuna on se, että ohjelman toimimiseen ei tarvitse kuin yhden EXE-tiedoston. (20.)

Dynaamisissa kirjastoissa yhtenä etuna on tarvittavan ohjelman pienempi koko, kun kirjastoja ei tarvitse sisällyttää ohjelmiin. Koska itse kirjastot eivät ole osana ohjelmaa, tämä tarkoittaa sitä, että monet ohjelmat voivat samanaikaisesti käyttää kirjaston tietoja, mikäli ohjelmat vain tietävät DLL-kirjastojen sijainnin. (20.)

Työssä käytettiin dynaamisia kirjastoja, joiden polku piti kertoa käytettävällä käyttöjärjestelmälle. Jos polkua ei kerrota, DLL-kirjastot pitää laittaa jokaisen projektin EXE-tiedoston kanssa samaan kansioon, mikä ei olisi käytännöllistä, jos tehdään useita projekteja.

DLL-kirjastojen paikka kerrottiin käyttöjärjestelmälle Path Editor -ohjelman avulla. Ohjelman avulla kirjoitettiin tietokoneen käyttöjärjestelmän rekisteriin osoite, jonka avulla käyttöjärjestelmä löytää kirjastot (19). Ohjelma käynnistettiin, ja hiiren oikealla painikkeella painettiin pää ikkunasta ja valittiin ’Insert New Item’. Tähän kohtaan kirjoitettiin osoite ’%OPENCV_DIR%\bin’, jonka jälkeen muutokset tallennettiin rekisteriin.

4.6 Paikallinen ja globaali menetelmä Visual Studiossa

Kirjastojen lisäämiseen Microsoftin Visual Studioon on kaksi menetelmää. Globaalissa menetelmässä Visual Studioon lisätään kirjastojen tiedot kaikkiin tuleviin ohjelmiin. Hyvänä puolena on se, että tämä täytyy tehdä vain kerran. Huonona puolena on se, että kaikissa tulevissa ohjelmissa tulee olemaan lisättyjen kirjastojen tiedot, vaikka ohjelma ei edes tarvitsisi niitä. Paikallisessa menetelmässä hyvänä puolena on se, että tulevissa projekteissa, joissa ei ole käyttöä OpenCV-kirjastoille, ei ole myöskään viitteitä kirjastoista. Huonona puolena on se, että linkit kirjastoon, pitää tehdä jokaiseen ohjelmaan erikseen. Tässä insinöörityössä valittiin paikallinen menetelmä. (20.)

4.7 OpenCV-sovellukseen vaadittavat tiedot

OpenCV-sovelluksen toteutukseen vaaditaan, että ohjelmointikielen kääntäjälle kerrotaan, miltä OpenCV kirjastot näyttävät, ja linkkerille pitää kertoa, mistä tarvittavat OpenCV:n funktiot löytyvät. Kääntäjälle kerrottavat OpenCV:n kirjastojen tiedot toteutetaan header-tiedostojen avulla. (20.)

4.8 Uuden projektin avaus

Kääntäjälle ja linkkerille kerrottavien tietojen laittaminen aloitettiin luomalla uusi tyhjä projekti. Tämä tapahtui avaamalla ’File’ ja seuraavaksi ’New Project’. Kieleksi valittiin Visual C++, NET Frameworkiksi 4, ja sovellustyypiksi Win32 Console Application. Uudelle projektille ja ratkaisulle annettiin haluttu nimi ja tallennuspaikka. Avautuvasta Application Wizardista valittiin ’Next’, jonka jälkeen valittiin ’Console Project’ ja ’Empty Project’. Tämän jälkeen poistuttiin Application Wizardista painamalla ’Finish’.

4.9 Paikallinen metodi

Jokainen projekti, mitä tehdään Visual Studio 2013:lla, sisältää omat sääntöpaketit. Näihin sääntöpaketteihin kuuluu kaikki tieto, mitä Visual Studio tarvitsee projektin rakentamiseen. Projektin rankentamisessa on kaksi eri vaihtoehtoa, ’debug’ ja ’release’. Deabug-moodi on tehty projektin koodin virheiden etsintää varten, kun taas release-moodi on optimoitu versio, jonka tarkoituksena on saada sovellus toimimaan mahdollisimman nopeasti. (20.)

Rakennusmoodien erilaisuudesta johtuen molemmilla on erilaiset sääntöpaketit, jotka pitää asettaa oikeanlaisiksi (20). Sääntöpaketteja muokataan Visual Studiossa Property Manager -välilehdellä. Välilehden saa avattua valitsemalla ’View’ – ’Property Pages’.

Property Managerissa näkyy käynnissä oleva projekti. Avaamalla projektista alasvetovalikon, tulee vaihtoehdoiksi kaksi kansiota: Debug Win32 ja Release Win32. Näihin kansioihin tehdään uudet sääntöpaketit, jotka ottavat huomioon OpenCV:n kirjastot.

4.9.1 Debug

Debug Win32 -kansiota klikattiin hiiren oikealla painikkeella, jolloin avautui valikko, josta valittiin ’Add New Project Property Sheet’. ’Property Sheet’ sisältää kaikki sääntöpaketin tiedot, mitä Visual Studio tarvitsee sovelluksen rakentamiseen.

Muodostunut uusi tiedosto nimettiin haluamalla tavalla. Tässä työssä tiedosto nimettiin OpenCV_Debug:ksi. Tiedosto avattiin, jotta päästiin muuttamaan Visual Studion tarvitsemia sääntöjä.

Ensimmäinen lisättävä asia oli C:n/C++:n alasvetovalikon General-välilehdellä, jonne lisättiin Additional Include Directories -kohtaan sijainti, jonne OpenCV:n include-kansio oli tallennettu (ks. esimerkkikoodi 4).

```
D:\OPENCV\opencv\build\include
```

Esimerkkikoodi 4. Include-kansion sijainti.

Seuraavaksi lisättiin Linker-alasvetovalikon General-välilehdeltä Additional Library Directories -kohtaan sijainti, jossa oli OpenCV:n lib-kansio (ks. esimerkkikoodi 5).

```
D:\OPENCV\opencv\build\x64\vc12\lib
```

Esimerkkikoodi 5. Lib-kansion sijainti.

Seuraavaksi linkkerille määriteltiin, mitä kirjastoja se tulisi käyttämään. Tämä tapahtui Linker-alasvetovalikosta, Input-välilehdeltä, josta valittiin Additional Dependencies. Avautuvaan ikkunaan kirjoitettiin kaikki kirjastot (ks. esimerkkikoodi 6), jotka OpenCV:stä löytyivät, vaikka kaikkia kirjastoja ei välttämättä käytettäisikään.

```
opencv_calib3d249d.lib
opencv_contrib2410d.lib
opencv_core2410d.lib
opencv_features2d2410d.lib
opencv_flann2410d.lib
opencv_gpu2410d.lib
opencv_highgui24910d.lib
opencv_imgproc2410d.lib
opencv_legacy2410d.lib
opencv_ml2410d.lib
opencv_nonfree2410d.lib
opencv_objdetect2410d.lib
opencv_ocl2410d.lib
opencv_photo2410d.lib
opencv_stitching2410d.lib
opencv_superres2410d.lib
opencv_ts2410d.lib
opencv_video2410d.lib
opencv_videostab2410d.lib
```

Esimerkkikoodi 6. Deabug-kirjastot.

Kirjastojen nimiin vaikutti se, mitä versiota OpenCV:stä käytettiin. Tässä insinöörityössä käytettiin OpenCV:n versiota 2.4.10, joten kirjastojen nimien lopussa on numero 2410. Mikäli OpenCV:n versiota ei tiedetä, voitiin kirjastojen nimet katsoa kansiosta, jossa ne sijaitsivat (ks. esimerkkikoodi 7).

```
D:\OPENCV\opencv\build\x64\vc12\lib
```

Esimerkkikoodi 7. Lib-kansion sijainti.

Tämän jälkeen tallennettiin muutokset Property Sheetiin.

4.9.2 Release

Release-moodin Property Sheetin muokkaus tapahtui samalla tavalla kuin Debug-moodin. Aluksi luotiin uusi Property Sheet, johon tehtiin edellä mainitut muutokset. Ainoa eroavaisuus oli kirjastojen nimissä (ks. esimerkkikoodi 8). Release-kirjastojen nimistä puuttui kirjaston versionumeron jälkeinen d-kirjain, joka tulee sanasta Debug.

```
opencv_contrib2410.lib
opencv_core2410.lib
opencv_features2d2410.lib
opencv_flann2410.lib
opencv_gpu2410.lib
opencv_highgui24910.lib
opencv_imgproc2410.lib
opencv_legacy2410.lib
opencv_ml2410.lib
opencv_nonfree2410.lib
opencv_objdetect2410.lib
opencv_ocl2410.lib
opencv_photo2410.lib
opencv_stitching2410.lib
opencv_superres2410.lib
opencv_ts2410.lib
opencv_video2410.lib
opencv_videostab2410.lib
```

Esimerkkikoodi 8. Release-kirjastot.

Kirjastojen lisäämisen jälkeen tallennettiin muutokset Property Sheetiin.

4.9.3 Sääntöpakettien tallennus ja käyttö

Visual Studio luo Property Sheetistä omat tiedostot, jotka kannattaa varmuuskopioida johonkin helposti löydettävään polkuun. Näin luotuja sääntöpaketteja voidaan käyttää uudestaan seuraavissa projekteissa, jolloin Property Sheetejä ei tarvitse mennä muokkaamaan. Sen sijaan, että valittaisiin ’Add New Project Property Sheet’, valitaankin

’Add Existing Property Sheet’, jolloin päästään lisäämään jo luotu Property Sheet uuteen projektiin.

4.9.4 32-bittinen ja 64-bittinen sovellus

Luodut Property Sheetit olivat osittain 32-bittiselle sovellukselle. Käytössä oleva tietokone oli 64-bittinen, piti joitain muutoksia vielä tehdä. Configuration Managerista piti lisätä x84-koneen tilalle tai seuraksi x64-kone. Tämä päästiin tekemään menemällä Visual Studiossa ’Build’-valikkoon, josta valittiin ’Configuration Manager’. Avautuvassa ikkunassa pystyttiin muuttamaan käytettävän projektin bittityyppiä 32-bittisestä 64-bittiseen.

Vaihdettaessa 32-bittisestä sovelluksesta 64-bittiseen sovellukseen kannatti Visual Studion Property Sheetistä tarkistaa vielä, että asetukset olivat varmasti tallentuneet. Property Managerista avattiin 64-bittisen ohjelman kansio, jonne lisättiin äsken luodut Property Sheetit. Property Sheetit avattiin ja tarkistettiin Linker-valikosta, että ’Target Machine’ on 64-bittinen.

4.10 Ongelmakohtia

4.10.1 64-bittinen järjestelmä

Työssä ongelmakohdaksi todettiin 32-bittisen järjestelmän ja 64-bittisen järjestelmän sekoittaminen keskenään. Tehtyä projektia ajettaessa Visual Studio 2013 antoi virheilmoituksen, joka valitti virhettä käyttöjärjestelmän bittimäärässä (virheilmoitus LNK1112). Keskustelupalstojen avulla selvisi, että ongelma oli Property Sheetien asetuksissa, jotka olivat asetettu 32-bittiselle järjestelmälle (21). Ongelman ratkaisi Property Sheetien korjaus.

4.10.2 Microsoft SDK

Toinen ongelma, joka työssä tuli vastaan, oli Microsoftin SDK:n puute Property Sheetistä. Ongelma ilmeni Visual Studion virheilmoituksessa, kun projektia ajettiin debug-moodissa. Ratkaisu ongelmaan löytyi jälleen OpenCV:n käyttäjäyhteisön kes-

kustelupalstoilta. Property Sheetin Addition Library Directoreihin piti lisätä, mistä Visual Studio löytäisi Microsoftin SDK:t (ks. esimerkkikoodi 9).

```
C:\Program Files (x86)\Microsoft SDKs\ Windows\v7.1A\lib\x64
```

Esimerkkikoodi 9. Microsoft SDK:n sijainti.

4.11 OpenCV:n toiminnan testaus kameran avulla

Toiminnan testaus toteutui parhaiten JAI:n GO5000M-PGE-kameralla, sillä valmistajan esimerkeistä löytyi valmiita OpenCV-esimerkkejä, joista avattiin yksi esimerkki, kun kamera oli kytkettynä tietokoneeseen. Projektiin lisättiin OpenCV:lle tehdyt Property Sheetit. Viimeinen muutos oli projektin koodissa olevien kirjastojen nimien versioiden muokkaus, joka piti laittaa vastaamaan asennuksen OpenCV:n versiota. Tämän jälkeen projekti voitiin ajaa Debug-moodissa. Projekti avasi sovelluksen, jonka avulla pystyttiin tuottamaan kuvaa GO5000M-PGE-kamerasta näytölle.

5 Kameroiden asentaminen

5.1 The Imaging Sourcen DFK MKU130-10x22 –kameran asentaminen

5.1.1 Kytkennät

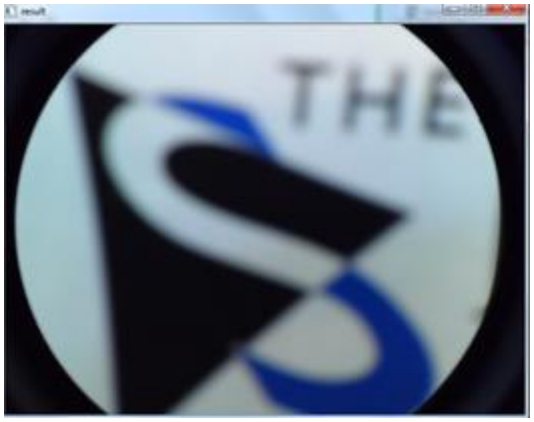
The Imaging Sourcen DFK MKU130-10x22 -kameran asennus aloitettiin kameran mukana tulleen USB 3.0 -kaapelin liittämisellä ensin kameraan ja sitten tietokoneen USB 3.0 -porttiin. Kamera vaatii toimiakseen USB 3.0 -portin, joten käytettävästä tietokoneesta on löydyttävä se.

5.1.2 Ajurit ja IC Full Screen Presenter

Seuraava vaihe oli ajureiden ja testausohjelmien asentaminen. Ajurit ja ohjelmat löytyivät valmistajan sivuilta, joilta ne voitiin ladata ilmaiseksi (1). Ajurin asennuksen jälkeen asennettiin IC Full Screen Presenter -ohjelma, jonka avulla voitiin helposti tarkistaa kameran toiminta.

5.1.3 Mikroskooppiin kiinnittäminen

Kameran toiminnan testauksen jälkeen kameraan olisi järkevintä asentaa mikroskooppi kiinni. Kun tiedetään, että kamera toimii, voidaan se asentaa haluttuun paikkaan. Tässä työssä ei päästy kokeilemaan kameraa mikroskoopin kanssa, joten kameran ottamat kuvat olivat epäselviä (ks. kuva 7).



Kuva 7. DFK MKU-130-10x22 kameran ottama kuva The Imaging Sourcen logosta.

5.1.4 MATLAB ja DFK MKU130

MATLAB:issa The Imaging Sourcen DFK MKU130-10x22 -kameraa voidaan käyttää muutamalla erilaisella tavalla. Ensimmäinen tapa on ladata MATLAB:iin laajennus, joka sallii web-kameroiden kuvankaappamisen MATLAB-ohjelmistoon (22). Tietokone tunnistaa DFK MKU130-10x22 -kameran web-kameraksi, jolloin tätä tapaa voidaan käyttää.

Toinen ja parempi tapa käyttää kameraa MATLAB:issa, on ladata valmistajan sivuilta MATLAB-liitännäinen, joka sisältää MATLAB:ille tarvittavat ajurit kameraa varten. Kun tiedostot on ladattu, ne pitää vielä asentaa. Tämä tapahtuu MATLAB:issa kirjoittamalla tarvittavat koodit (ks. esimerkkikoodi 10).

```
32-bittinen järjestelmä:
imaqregister('[polku]\win32\TISImaq_R2013.dll')
64-bittinen järjestelmä:
imaqregister('[polku]\x64\TISImaq_R2013.dll')
```

```
[polku] = tietokoneessa oleva polku, jonne IC Matlab Plugin on
asennettu.
```

Esimerkkikoodi 10. MATLAB-koodi ajureita varten.

Liitännäisen asentamisen jälkeen kameran toimivuuden testaaminen MATLAB:issa pystyttiin varmistamaan avaamalla Image Acquisition Toolbox. Image Acquisition Toolboxin Hardware Browserissa näkyi nyt kameran mallin nimi ja käytössä olevat ajurit. Kamera avattiin Hardware Browserista, ja valittiin ilmestyvästä listasta haluttu kameran resoluutio, jotka ovat DFK MKU130-10x22 -kamerassa 640x480 pikselin ja 4128x3096 pikselin välillä. Resoluutioksi valittiin oletuksena oleva 1280x720, jonka jälkeen valittiin Preview-ikkunasta ’Start Preview’, jonka jälkeen ikkunaan ilmestyi kamerasta tuleva kuva.

5.1.5 OpenCV ja DFK MKU130-10x22

OpenCV-ympäristössä kameran toiminnan testaaminen tapahtui helposti web-kameralle kirjoitettujen ohjelmien avulla. Tässä testaamisessa tarvittiin tieto siitä, että OpenCV-kirjastot ovat asennettuina koneelle johonkin ohjelmistonkehitysympäristöön. Tässä insinöörityössä kirjastot asennettiin Visual Studio 2013:een. Asentamisen jälkeen luotiin uusi ohjelmatiedosto, joka tässä työssä oli C++-tiedosto. Tiedoston luonnin jälkeen lisättiin tarvittavat Debug- ja Release-Property Sheetit, jonka jälkeen tarkistettiin, että ohjelman asetuksiin oli merkitty käyttöjärjestelmän bittimäärä oikeaksi. Viimeiseksi lisättiin nettikameralle tarkoitettu koodi (ks. esimerkkikoodi 11). Koodin käytössä hyödynnettiin sitä, että tietokone lukee USB 3.0 -portin samalla tavalla virtuaaliseksi COM-portiksi kuin USB 2.0 -portinkin, joten koodia ei tarvinnut siltä osin muuttaa (23).

Ensimmäinen toimintaan saatu OpenCV-ohjelma DFK MKU130-10x22-kameralle oli suhteellisen yksinkertainen. Ohjelma kaappasi kamerasta tulleen kuvan, jonka se muutti mustavalkoiseksi. Muutoksen jälkeen ohjelma erotti kuvasta reunat hyödyntäen pikseleiden arvojen eroja. Ohjelman avulla saatiin erotettua reunoja kuvasta ja tuloksen ohjelma näytti ruudulla mustavalkoisena. Toimivia koodeja löytyi internetistä muutamia (24; 25).

```
#include "opencv2/opencv.hpp"
using namespace cv;
int main(int, char**)
{
```

```
        VideoCapture cap(0);
        if (!cap.isOpened())
                return -1;
        Mat edges;
        namedWindow("edges", 1);
        for(;;)
        {
                Mat frame;
                cap >> frame;
                cvtColor(frame, edges, CV_BGR2GRAY);
                GaussianBlur(edges, edges, Size(7,7), 1.5,
                1.5);
                Canny(edges,edges, 0, 30 ,3);
                imshow("edges", edges);
                if (waitKey(30) >=0) break;
        }
        return 0;
}
```

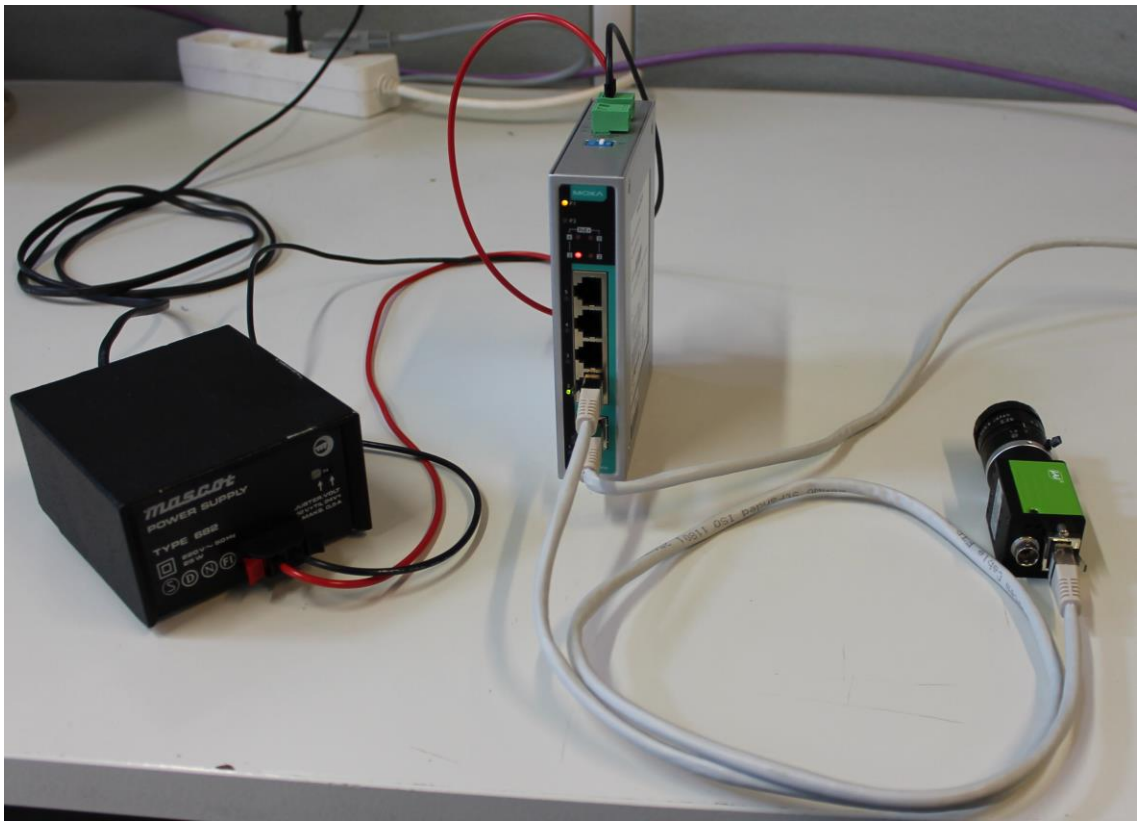
Esimerkkikoodi 11. Reunantunnistus ohjelma DFK MKU130-10x22-kameralle (25).

5.2 JAI GO-5000M-PGE -kameran asentaminen

5.2.1 Kytkentä

JAI GO-5000M-PGE -kameran asentaminen aloitettiin C-tyypin linssin kiinnityksellä kameraan. Tämän jälkeen piti valita, halutaanko käyttää PoE-ominaisuutta vai syötettiinkö virta erillisellä kaapelilla Hirose-liittimen kautta. Mikäli päädyttäisiin käyttämään Hirose-liitintä, tarvittaisiin kameralle 12 voltista 24 volttiin tasavirtajännitelähteen. Mikäli käytetäisiin PoE-ominaisuutta, tarvitaisiin verkkokortti, jossa on PoE-ominaisuus, tai tarvittaisiin erillinen kytkin tai reititin, joka toimii PoE-virtalähteenä.

Tässä työssä käytettiin PoE-kytkintä, johon pystyi kytkemään enintään neljä PoE-laitetta, esimerkiksi kameraa. PoE-kytkimenä toimi MOXA EDS-G205A-4PoE -kytkin, johon syötettiin virtaa muuntajan kautta. Kytkimen virtalähdepaikkaan kytkettiin ensin virtalähteestä tulevat johdot, jonka jälkeen virtalähteeseen kytkettiin virrat päälle. Tämän jälkeen kameran ja kytkimen välille sekä tietokoneen ja kytkimen välille laitettiin GigE:iä tukevat suoraankytketyt CAT5e–kaapelit (ks. kuva 8).

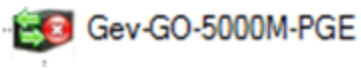


Kuva 8. GO-5000M-PGE-kameran kytkentä.

5.2.2 JAI Control Toolin toiminta

Kaapeleiden kytkemisen jälkeen pystyttiin aloittamaan kameran toiminnan testaus, joka tapahtui helpoiten JAI Control Toolin avulla, joka oli asennettuna tietokoneeseen. JAI Control Toolin käynnistämisen jälkeen vasemmassa ikkunassa näkyi kaikki tietokoneeseen kytketyt kamerat, jotka ohjelma tunnisti. Kameroiden tunnistaminen riippui ohjelman asetuksista. Ohjelma tunnisti heti aluksi GigE-kamerat, mutta osa muista liitäntämenetelmistä oli asetuksissa oletuksena estetty.

Jos kamera näkyy, mutta sen päällä on punainen merkki, tarkoittaa se, että kameraan ei ole yhteyttä (ks. kuva 9). Yhteyden kameraan saa pelkästään klikkaamalla kameran symbolia. Mikäli ohjelma antaa virheilmoituksen yhteyttä muodostaessa, voi vikana olla kameran sisällä oleva XML-tiedosto. Jos kameran päällä on salaman symboli, on tällöin käytössä filter-ajurit. Filter-ajurit ovat verkkokorttien ajureita, jotka ovat suunniteltu konenäkösovelluksien tarpeisiin. Ne tarjoavat parempaa suorituskykyä kamerasovelluksille, mikä huomataan esimerkiksi ruudunpäivitysnopeuden paranemisena sovelluksissa ja prosessorin kuormituksen vähentymisenä (26, s. 3).



Kuva 9. Kameraan ei ole yhteyttä.

Filter-ajurit voidaan kytkeä päälle tai pois tarpeen mukaan. Tämä tapahtuu Windowsin Verkko- ja jakamiskeskuksesta, josta valitaan kameran käytössä oleva verkon yhteyden tila. Yhteyden tila ikkunasta avataan ominaisuudet, josta voidaan valita, onko filter-ajurit käytössä vai ei.

Yhteyden muodostamisen jälkeen aloitettiin kameran toiminnan testaus. Kameran toiminnan testaus saatiin helposti suoritettua JAI Control Toolin Acquisition -toiminnon avulla. Painamalla Start Acquisition -painiketta, ohjelma aloitti kuvankaappauksen, jolloin nähtiin, että kamera toimi.

5.2.3 MATLAB ja GO-5000M-PGE

JAI:n GO-5000M-PGE-kamera saatiin toimimaan helposti myös MATLAB:in kanssa. MATLAB:iin piti olla asennettu Image Acquisition Toolbot -lisäosa, jonka avulla voitaisiin kontrolloida ja kaapata kuvia kamerasta. Image Acquisition Toolbox ei tunnistanut itsestään kaikkia kommunikointimenetelmiä, kuten esimerkiksi GigE Vision, vaan niiden ajurit piti asentaa paketteina Image Acquisition Toolboxiin. GigE Vision -kameroiden ajurit löytyivät MATLAB:in valmiiden ajureiden listoilta.

Ajurien asennuksen jälkeen Hardware Browserista piti hakea uudestaan kameroita, jonka jälkeen kaikki kytketyt GigE-kamerat näkyi Hardware Browserissa. Hardware Browserin alasvetovalikosta valittiin haluttu tallennusmetodi (5, s.28), jotka GO-5000M-PGE-tapauksessa olivat:

- Mono8
- Mono10
- Mono10 packed
- Mono12
- Mono12 packed

Tallennusmoodin valinnan jälkeen Preview-ikkunasta painettiin ’Start Preview’, jolloin ikkunaan tuli suoraan kuvaa kameralta.

5.2.4 OpenCV ja GO-5000M-PGE

JAI Control Toolboxin asentamisen yhteydessä tietokoneelle asentui esimerkkikoodeja, joiden avulla pystyttiin helposti kokeilemaan kameran toimintaa OpenCV-ympäristössä. Esimerkkikansiosta avattiin OpenCV:lle tehty esimerkkikansio, josta avattiin SLN-tiedosto. Tiedosto avautui Visual Studio 2013:ssa.

Ohjelmaan piti lisätä oikeat debug- ja release-Property Sheetit, jonka jälkeen piti vielä valita oikea käyttöjärjestelmätyyppi, joka tässä työssä oli 64-bittinen ohjelma. Ohjelmakoodista piti muuttaa käytettävien kirjastonimien versiot oikeaksi, jonka jälkeen ohjelman oli valmis kokeiltavaksi.

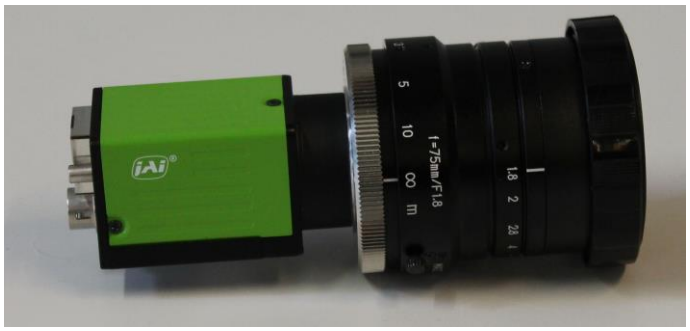
5.2.5 Linssien kokeilu

Linssien kokeilu suoritettiin käyttämällä JAI Control Toolia. JAI Control Tool käynnistettiin ja kameralla aloitettiin kuvan hankinta ’Acquisition’-toiminolla. Kameraan kiinnitettiin vuorotellen käytössä olevat linssit ja linssien tuottamia kuvia vertailtiin toisiinsa.

Ensiksi kamerassa oli kiinni GMHR47518MCN-1–linssi (ks. kuva 10). Linssin tuottamat kuvat olivat tarkoitettu lähietäisyydelle (Liite 1). Linssi voisi sopia esimerkiksi kappaletavaroiden tarkistamiseen.

Kuva 10. GMHR47518MCN-1–linssi GO-5000M-PGE–kamerassa.

Go-5000M-PGE-kameraan seuraavaksi kiinnitetty linssi oli GMTHR32514MCN (ks. kuva 11). Linssi todettiin fyysisesti selvästi GMHR47518MCN-1–linssiä suuremmaksi. Linssin avulla otetuista kuvista voitiin päätellä, että linssi oli tarkoitettu kauempien kohteiden kuvantamiseen (Liite 2). Linssiä voitaisiin käyttää suurempien alueiden seurantaan ja esimerkiksi liikkeentunnistukseen.



Kuva 11. GMTHR32514MCN–linssi GO-5000M-PGE–kamerassa.

Linssejä kokeiltaessa huomattiin kameran likaantumisen helppous. Kameran kennolle oli päässyt jokin roska, vaikka kamerassa oli lähes aina jokin linssi kiinni. Pienen roskan aiheuttama vääristymä näkyi kameralla otetuissa kuvissa (Liite 1; Liite 2), kunnes kameran kenno puhdistettiin.

5.2.6 Ongelmakohtia

Ongelmakohtia työssä oli vioittunut XML-tiedosto ja sen korjaus. Vian määrittäminen XML-tiedostoon vei oman aikansa, mutta tiedoston päivitys oli onneksi nopeaa. Kameran sisäisten tiedostojen päivitys onnistuu JAI Camera Update Toolin avulla. Tämä ohjelma lataa internetistä uusimmat tiedostot käytössä oleville kameroille ja tämän jälkeen asentaa ne kameraan. Kameran tiedostojen päivityksen jälkeen kamera toimi moitteettomasti JAI Control Toolin ja muiden ohjelmien kautta.

Ennen vian selvittämistä epäilyksenä oli, että ongelma olisi saattanut johtua käytetyistä kaapeleista, joten kameran kanssa kokeiltiin myös ristiinkytkettyä CAT5e-kaapelia, tuloksetta. Kaapeleiden testauksen jälkeen vian arveltiin olevan verkkokortissa, joka ei ollut valmistajan suositusten mukainen. Käytössä olevaan tietokoneeseen hankittiin valmistajan suosittelema Intel 1000/PRO -verkkokortti, joka ei sekään ratkaissut ongelmaa, sillä ongelma oli kameran tiedostoissa.

6 Kameroiden vertailu

6.1 Liitäntätapojen vertailu

Kameroiden kytkennässä erona on kameroiden liitäntätapa tietokoneeseen. Toinen kameroista on varustettu USB 3.0 -liittimellä, kun taas toinen on varustettu GigE-liitännällä.

DFK MKU130:n etuna on nopeampi tiedonsiirto, kun taas GO-5000M-PGE:n etuna on pidempi kytkentäetäisyys (ks. taulukko 1). Molemmat saavat virtansa tarvittaessa tietoa lähettävän kaapelin kautta, mutta tämä vaatii enemmän työtä GO-5000M-PGE-kamerassa, koska kamera saa virtansa PoE:n kautta. DFK MKU130 -kamerassa riittää, kun tietokoneessa on USB 3.0 -liitin. GO-5000M-PGE-kamera voi olla kaukana tietokoneesta, koska kaapelin maksimipituus 100 m. GO-5000M-PGE-kameran ei tarvitse olla suoraan kytkettynä tietokoneeseen, vaan se voi olla kytkettynä myös reitittimen tai kytkimen kautta tietokoneeseen.

Taulukko 1. Kameroiden ominaisuudet (27; 2, s.1-2; 28).

Kameran malli	**Liitäntämenetelmä**	**Kaapelin enimmäispituus**	**Nopeus**	**Virransyöttö**
GO-5000M-PGE	GigE Vision	100 m	1000 Mbit/s	PoE/Hirose
DFK MKU130	USB 3.0	3 m (käytännöllinen pituus)	5 Gbit/s	USB 3.0

6.2 Asennuksen vertailu

Kameran asentaminen ja toimintaan saaminen oli helpompaa DFK MKU130 -kameralla. Kameran liittäminen tietokoneeseen oli helppoa ja ajurit sekä ohjelmat toimivat ensimmäisen asennuskerran jälkeen moitteetta. DFK MKU130 -kameran käyttötarkoituksen takia kamera tarvitsee mikroskoopin, jotta sitä voidaan käyttää, mutta muita lisäosia ei kameraan tarvitse. Mikroskoopin tarve pienentää kameran käyttömahdollisuuksia, sillä kameraan ei voida käyttää kaukana olevien objektien kuvaamisen, kuten GO-5000M-PGE:tä voidaan halutessa käyttää.

GO-5000M-PGE-kameran asennus oli DFK MKU130:tä hankalampaa. Virransyöttö piti ensin saada toimimaan PoE:n kautta, joka tarvitsi tässä työssä erillisen kytkimen ja virtalähteen. Virransyötön jälkeen kameran pystyi kytkemään suoraan tietokoneeseen reitittimen kautta. Ohjelmien asentaminen oli GO-5000M-PGE-kameralle hieman työläämpää. Kameran vioittuneen XML-tiedoston takia kameran toimintavalmiuteen saattaminen kesti pitkän aikaa. Kamera tarvitsee asennukseensa vielä lisäksi adapterin, jolla kamera saadaan kytkettyä kolmijalkaan tai muuhun vastaavaan telineeseen. Kameralle tarvitsee myös hankkia kunnollinen valaistus. GO-5000M-PGE kuumenee käytössä enemmän kuin DFK MKU130-10x22, jonka takia se tarvitsisi mahdollisesti jonkin viilennysratkaisun.

6.3 Loppuvertailu

GO-5000M-PGE ja DFK MKU130-10x22 -kamerat ovat tehty täysin erilaisia sovelluksia varten, joten niitä ei voida verrata täysin toisiinsa. DFK MKU130-10x22 -kamerassa oli helpompi käyttöönotto kuin GO-5000M-PGE:ssä. DFK MKU130-10x22 tarvitsee myös vähemmän erillisiä osia käyttöönottoonsa, sekä siinä on parempi tiedonsiirtonopeus. DFK MKU130-10x22 ei myöskään kuumene yhtä paljon kuin GO-5000M-PGE, joka myös vähentäisi tarvittavia erillisiä osia käytössä.

GO-5000M-PGE on parempi vaihtoehto, jos haluaa lisätä monia kameroita samaan tietokoneeseen kiinni. Kameroista lähetettävä tieto välittyisi tietokoneeseen yhden kaapelin avulla, joka vähentäisi asennuskustannuksia. Kameroiden sijoituspaikka voi myös olla paljon kauempana käytettävästä tietokoneesta kuin DFK MKU130:n tapauksessa. GigE Vision -standardi ei myöskään estä 10 Gbit/s:n Ethernetin käyttö, joten tulevaisuudessa Ethernet-kameroilla saattaa olla nopeampi tiedonsiirtonopeus kuin USB 3.0 -kameroilla. GO-5000M-PGE kamera on myös monikäyttöisempi kuin DFK MKU130-10x22, sillä siihen voidaan liittää monia erilaisia linssejä kiinni.

DFK MKU130-10x22:n USB 3.0:n kanssa sopisi parhaiten projekteihin, joissa tarvitaan vain yhtä kameraa. Kameran käyttöönotto onnistuu myös todella helposti, joten käyttöönottaja voisi olla myös maallikko.

GO-5000M-PGE sopisi parhaiten projekteihin, joissa kuvien käsittelyä tekevä tietokone ei voisi olla kameroiden lähellä. Kamera sopii myös projekteihin, joissa tarvitaan useaa kameraa yhtä aikaa, tai projektiin, jota mahdollisesti tulevaisuudessa tarvitsee helposti laajentaa uusilla kameroilla.

7 Ohjelmien vertailu

7.1 Kameranvalmistajien omat ohjelmat

The Imaging Sourcen kameralle löytyvät ohjelmat löytyvät selkeästi ja helposti valmistajan sivujen kautta, kyseisen kameran tuotesivujen kautta. Laajennuksia, ajureita ja

ohjelmia on monia erilaisia, joista voi valita haluamansa. Käytetyt ohjelmat asentuivat helposti ja toimivat ongelmitta.

JAI:n kameraan tarvittavat ohjelmat löytyvät valmistajan sivuilta kameran tuotesivun linkistä. JAI:lla on yksi ohjelmapaketti, joka sisältää kaikki tarvittavat ohjelmat kameran käyttöönottoa varten. Paketti sisältää myös monia hyviä esimerkkejä erilaisiin sovelluksiin, joita kameran kanssa pystyy käyttämään. Kameran päivitykseen tarvittavan JAI Camera Update Toolin pystyi lataamaan, kun valmistajan tekninen tuki vastasi tukipyyntöön ja lähetti internetosoitteen, josta kyseisen ohjelman sai ladattua.

Molempien valmistajien ohjelmat toimivat hyvin ja olivat helppoja asentaa; suurin ero on siinä, että JAI:lla tarvittavat ohjelmat tulevat yhdellä asennuksella, kun taas The Imaging Sourcella käyttäjä pystyy valitsemaan haluamansa ohjelmat lataukseen.

7.2 OpenCV ja MATLAB

7.2.1 Avoin lähdekoodi ja kaupalliset vastineet

OpenCV perustuu avoimen lähdekoodin BSD-lisenssiin. Tämä tarkoittaa sitä, että kirjaston käyttö ja muokkaus on käyttäjälle ilmaista. OpenCV:n kirjastot sisältävät hyvin monia erilaisia algoritmeja, mutta mikä erottaa sen kaupallisista vastineista, on käyttöliittymän puute, hinta ja tekninen tuki.

Käyttöliittymän puute tarkoittaa sitä, että päästäkseen hyödyntämän OpenCV:n kirjastoja käyttäjä tarvitsee jonkin ohjelman, jonka avulla hän pystyy käyttämään kirjastoja, esimerkiksi Visual Studion. Käyttöliittymän puute tarkoittaa myös sitä, että kirjastojen hyödyntäminen vaatii enemmän ohjelmointiosaamista käyttäjältä.

OpenCV:n BSD-linsenssin avulla ohjelman käyttö on ilmaista, toisin kuin esimerkiksi MATLAB:in käyttö, joka vaatii ohjelman ja sen lisäosien ostamista. OpenCV:n käyttö vaatii tosin jonkin ohjelmistonkehitysympäristön, joka saattaa maksaa. OpenCV:n käyttö tulee kuitenkin olemaan halvempi kuin jonkin kaupallisen ohjelman käyttö.

Avoimen lähdekoodin ohjelmissa ei ole yleensä teknistä tukea, vaan apua haetaan käyttäjäyhteisöistä, jonne samaa ohjelmaa käyttävät ihmiset vastaavat keskustelupalstoille jätettyihin kysymyksiin. Tämä tarkoittaa sitä, että vastausta haluamaasi ongelmaan ei välttämättä löydy nopeasti ja helposti. Toisaalta laaja käyttäjäyhteisö mahdollistaa sen, että joku on jo aikaisemmin kohdannut saman ongelman ja siihen löytyy vastaus jo valmiiksi kirjoitettuna. Kaupallisissa sovelluksissa ongelman voi esittää tekniseen tukeen ja tällöin saa apua henkilö- ja ohjelmakohtaisesti.

7.2.2 OpenCV vai MATLAB

Valinta OpenCV:n ja MATLAB:in välillä ei ole yksinkertainen, sillä OpenCV:n kirjastot pystytään lisäämään MATLAB:iin, jolloin MATLAB:in käyttöön tulee vielä laajemmat kirjastot.

Ensiksi pitää määritellä, kuinka paljon rahaa on projektiin käytössä, sillä MATLAB:in hankkiminen tulee kalliimmaksi kuin OpenCV:n hankkiminen. Molempien internetsivuilta löytyy erilaisia harjoituksia ja ohjeita, joita voi hyödyntää käyttöönotossa, mutta Mathworksin sivuilla osa video-ohjeista vaatii rekisteröitymistä.

Seuraavaksi pitää kartoittaa ohjelmointiosaaminen. Vaikka MATLAB:in ja OpenCV:n käyttö vaatii ohjelmointitaitoa, MATLAB:in käytön aloitus ja asentaminen on helpompaa kuin OpenCV:n. MATLAB:in avulla pääsee helposti testaamaan kameroiden toimintaa ja kameroiden asetuksia pystyy muuttamaan laajennuksien avulla helposti ilman ohjelmointitaitoa.

Työssä todettiin MATLAB:in olevan helpommin asennettava ja lähestyttävä kuin Visual Studio 2013:een asennettu OpenCV. OpenCV:n kirjastojen liittämismahdollisuus MATLAB:iin on myös erittäin suuri etu MATLAB:issa. Insinöörityön perusteella MATLAB:in hankinta kannattaisi toteuttaa, mikäli projektin resurssit mahdollistavat sen.

8 Insinöörityön tavoitteet ja niiden täyttyminen

Insinöörityön alkuperäisiin tavoitteisiin kuului GO-5000M-PGE- ja DFK MKU130-10x22 -kameroiden käyttöönotto MATLAB:issa ja OpenCV:llä Windows- ja Linux-ympäristössä. Tavoitteet saavutettiin osittain. Kameroiden käyttöönotto ehdittiin toteut-

taa vain Windowsissa. Alkuperäisen suunnitelman mukaan kameroita olisi pitänyt testata myös Linux-käyttöjärjestelmällä, mutta ajanpuutteen takia tätä ei ehditty toteuttaa.

Toteutunut käyttöönotto Windowsissa OpenCV:n ja MATLAB:in avulla onnistui viivästymisten jälkeen alkuperäisten suunnitelmien mukaisesti.

Kaikkiin alkuperäisiin tavoitteisiin olisi todennäköisesti päästy, mikäli insinöörityön tekemiseen olisi ollut hieman enemmän aikaa. Saavutetut tavoitteet riittivät kuitenkin täyttämään Metropolia Ammattikorkeakoulun työlle asettamat vaatimukset, jotka olivat kameroiden käyttöönotto.

9 Yhteenveto

Insinöörityössä suoritettiin JAI GO-5000M-PGE- ja The Imaging Sourcen DFK MKU130-10x22 -kameroiden käyttöönotto Windows-käyttöjärjestelmällä OpenCV:n kirjastoilla Visual Studio 2013:ssa ja MATLAB-ohjelmalla.

Työssä tuli esille, että kameroiden sisällä olevien tiedostojen takia projektien käyttöönotto voi myöhästyä muutamilla viikoilla. Selvittämättä jäi kameroiden käyttöönotto Linux-käyttöjärjestelmällä.

Työtä voisi jatkaa tutustumalla MATLAB:in ja OpenCV:n algoritmeihin paremmin ja soveltamalla niitä käytössä olleisiin kameroihin. Työtä voisi jatkaa kiinnittämällä DFK MKU130-10x22 kamera kiinni mikroskooppiin, jolloin kameraa pystyttäisiin hyödyntämään paremmin. Työtä voitaisiin jatkaa myös rakentamalla GO-5000M-PGE –kameralle teline ja tilat, jonne se voitaisiin laittaa, kun se otetaan laboratoriokäyttöön.

Työn tulosten avulla voidaan jatkossa helpommin löytää joitakin virheitä GigE Vision -kameroiden asennuksessa, sekä OpenCV:n kirjastojen käyttämisen aloittaminen on helpommin lähestyttävissä Windows-käyttöjärjestelmässä. Työ mahdollistaa Metropolia Ammattikorkeakoululle nopean mahdollisuuden ottaa kamerat käyttöön opetustarkoituksessa. Työn tuloksena MATLAB -ohjelman konenäkötyökalulaatikkojen hankintaa voidaan suositella Metropolia Ammattikorkeakoulun opetuskäyttöön.

Lähteet

1. DFKMKU130-10X22. 2015. Verkkodokumentti. The Imaging Source. <http://www.theimagingsource.com/en_US/products/microscope-cameras/usb-cmos-color/dfkmku13010x22/>. Luettu 25.2.2015.

2. GO-5000-PGE 5-megapixel CMOS global shutter. 2015. Verkkodokumentti. JAI. <http://www.jai.com/ProtectedDocuments/datasheets/Datasheet_GO-5000-PGE.pdf>. Luettu 25.2.2015.

3. Getting Started Guide JAI SDK Software Development Kit and Control Tool. 2015. Verkkodokumentti. JAI. Document Version: K. <http://www.jai.com/SiteCollectionDocuments/Camera_Solutions_Manuals/JAI-SDK-Getting-Started-Guide_revK.pdf>. Luettu 25.2.2015.

4. Acquiring from GigE Vision Cameras with Vision Acquisition Software - Part I. 2013. Verkkodokumentti. National Instruments. <http://www.ni.com/white-paper/5651/en/>. 11.7.2013. Luettu 26.2.2015.

5. User Manual GO-5000M-PGE GO-5000C-PGE 5M Digital Progressive Scan Monochrome and Color Camera. 2015. Verkkodokumentti. JAI. Document Version: 1.1. <http://www.jai.com/ProtectedDocuments/Manuals/Manual_GO-5000-PGE.pdf>. Päivitetty tammikuu 2015. Luettu 26.2.2015.

6. EDS-G205A-4PoE Hardware Installation Guide Moxa EthernetDevice Switch. 2014. Verkkodokumentti. Moxa. Second Edition. <http://www.moxa.com/doc/man/EDS-G205A-4PoE_HIG_2e.pdf>. Luettu 1.3.2015.

7. EDS-G205A-4PoE Series. 2014. Verkkodokumentti. Moxa. <http://www.moxa.com/doc/specs/EDS-G205A-4PoE_Series.pdf>. Updated Apr. 9, 2014. Luettu 1.3.2015.

8. Jumbo Frame. 2015. Verkkodokumentti. Wikimedia Foundation, Inc. <http://en.wikipedia.org/wiki/Jumbo_frame>. Modified on 9 March 2015. Luettu 5.4.2015.

9. 1inch Format High Resolution Lenses Item No.GMHR47518MCN-1. 2015. Verkkodokumentti. GOYO OPTICAL Inc. <http://www.goyooptical.com/products/industrial/GMHR47518MCN-1.pdf>. Luettu 10.3.2015.

10. High Resolution Lenses Item No.GMTHR32514MCN. 2015. Verkkodokumentti. GOYO OPTICAL Inc. <http://www.goyooptical.com/products/industrial/GMTHR32514MCN.pdf>. Luettu 10.3.2015.

11. About. 2015. Verkkodokumentti. Itseez. <http://opencv.org/about.html>. Luettu 1.3.2015.

12. Introducing Visual Studio. 2015. Verkkodokumentti. Microsoft Corporation. <https://msdn.microsoft.com/en-us/library/fx6bk1f4(v=vs.100).aspx>. Luettu 20.3.2015.

13. Features. 2015. Verkkodokumentti. The MathWorks, Inc. <http://se.mathworks.com/products/matlab/features.html>. Luettu 2.3.2015.

14. Image Processing Toolbox. 2015. Verkkodokumentti. The MathWorks, Inc. <http://se.mathworks.com/products/image/>. Luettu 2.3.2015.

15. Computer Vision System Toolbox. 2015. Verkkodokumentti. The MathWorks, Inc. <http://se.mathworks.com/products/computer-vision/>. Luettu 2.3.2015.

16. Image Acquisition Toolbox. 2015. Verkkodokumentti. The MathWorks, Inc. <http://se.mathworks.com/products/imaq/>. Luettu 2.3.2015.

17. Path Editor. 2015. Verkkodokumentti. Softpedia. <http://www.softpedia.com/get/System/System-Miscellaneous/Path-Editor.shtml>. Luettu 27.2.2015.

18. OpenCV. 2013. Verkkodokumentti. Slashdot Media. <http://sourceforge.net/projects/opencvlibrary/files/opencv-win/2.4.10/opencv-2.4.10.exe/download>. Updated 2013-04-05. Luettu 27.2.2015.

19. Gábor Bernát. 2015. Installation in Windows. Verkkodokumentti. Opencv. <http://docs.opencv.org/doc/tutorials/introduction/windows_install/windows_install.html#windows-installation>. Updated on Feb 25, 2015. Luettu 27.2.2015.

20. Gábor Bernát. 2015. How to build applications with OpenCV inside the Microsoft Visual Studio. Verkkodokumentti. Opencv. <http://docs.opencv.org/doc/tutorials/introduction/windows_visual_studio_Opencv/windows_visual_studio_Opencv.html#windows-visual-studio-how-to>. Updated on Feb 25, 2015. Luettu 28.2.2015.

21. Fatal error LNK1112: module machine type 'x64' conflicts with target machine type 'X86'. 2010. Verkkodokumentti. stack exchange inc. <http://stackoverflow.com/questions/3563756/fatal-error-lnk1112-module-machine-type-x64-conflicts-with-target-machine-typ>. Luettu 16.3.2015.

22. USB Webcam Support with MATLAB. 2015. Verkkodokumentti. The MathWorks, Inc. <http://se.mathworks.com/hardware-support/matlab-webcam.html>. Luettu 20.3.2015.

23. Alexon Jan. 2015. Create a USB Virtual COM Port. Verkkodokumentti. <http://janaxelson.com/usb_virtual_com_port.htm>. Luettu 16.3.2015.

24. Iswara Agung Mega. 2011. OpenCV C/C++ Examples (Camera Capture). Verkkodokumentti. <http://hxr99.blogspot.fi/2011/12/opencv-examples-camera-capture.html>. Luettu 15.3.2015.

25. Class VideoCapture . 2015. Verkkodokumentti. Opencv. <http://docs.opencv.org/java/2.4.4/org/opencv/highgui/VideoCapture.html>. Luettu 16.3.2015.

26. Israel Scott. 2015. GigE cameras Myth & Facts. Verkkodokumentti. 1stVision Inc. <http://www.1stvision.com/elearning/GigECameras_Myths_Facts.pdf>. Luettu 10.3.2015.

27. GigE Vision - True Plug and Play Connectivity. 2015. Verkkodokumentti. AIA. <http://www.visiononline.org/vision-standards-details.cfm?type=5>.Luettu 10.4.2015.

28. USB 3.0 . 2015. Verkkodokumentti. Wikimedia Foundation, Inc. <http://en.wikipedia.org/wiki/USB_3.0>. Luettu 10.4.2015.

GMHR47518MCN-1–linssillä otettu kuva

GMTHR32514MCN–linssillä otettu kuva