KEMI-TORNION AMMATTIKORKEAKOULU

BGA-komponentin asennustyökalu Dima FP-600 ladontakoneeseen

Juha Jakunaho, Panu Rikula

Tietotekniikan opinnäytetyö
Digitaalitekniikka
Insinööri(AMK)

KEMI 2012

ALKUSANAT

Tahdomme kiittää työn antajaa Antti Niemelää tästä haastavasta sekä opettavaisesta työstä, Mika Mörtiä, Janne Vaaraniemeä sekä Jukka Leinosta neuvoista sekä opastuksesta työssä, valvojaa Tapani Ruokasta ohjauksesta ja Teemu Lakelaa, joka auttoi meitä 3D-kuvissa sekä oikoluvussa.

TIIVISTELMÄ

Kemi-Tornion ammattikorkeakoulu, Tekniikan ala	
Koulutusohjelma	Tietotekniikka
Opinnäytetyön tekijät	Juha Jakunaho, Panu Rikula
Opinnäytetyön nimi	BGA-komponentin asennustyökalu Dima FP-600 ladontakoneeseen
Työn laji	Opinnäytetyö
Päiväys	12.3.2012
Sivumäärä	205 + 19 liitesivua
Opinnäytetyön ohjaaja	DI yliopettajaTapani Ruokanen
Yritys	Kemi-Tornion Ammattikorkeakoulu
Yrityksen yhteyshenkilö/valvoja	Projektipäällikkö Antti Niemelä

Tehtävänä oli suunnitella ja toteuttaa Kemi-Tornion Ammattikorkeakoulun Tekniikan TKI:n Sulautettujen järjestelmien laboratorioon Dima FP-600 -ladontakoneeseen BGA-pintaliitoskomponentin asennuksen apulaite. Oikea asennuskohta katsottaisiin konenäön avulla ja komponenttia kuljettavaa kättä ohjattaisiin moottorein.

Rajauksina oli, että kuvankäsittely tuli tehdä avointa OpenCV 2.3.1 -kuvankäsittelykirjastoa käyttäen ja laitteen tarkkuusmääritelmäksi saatiin 0,5 mm. Apulaite tuli toteuttaa mahdollisimman edullisesti ja se ei saanut haitata laitteen normaalia käyttöä.

Työ aloitettiin esitutkimuksella, jonka tuloksena saatiin selville mahdolliset toteutustavat. Työhön parhaiten sopivat laitteet karsiutuivat pois korkeiden kustannusten takia, jolloin jouduttiin etsimään vaihtoehtoisia toteutustapoja. Toteutustavan selvittyä tutkittiin eri tekniikoita sekä laitteita, mitä käytettäisiin laitteen suunnitteluun.

OpenCV:n ohjelmointi tapahtui Linux-koneella, johon oli Ubuntu-käyttöjärjestelmän 10.04.4 versio asennettuna. Kuvankäsittely ja ohjelmointi aloitettiin kirjaston Python-kielelle käännetyllä versiolla, mutta sen hetken version bugisuuden takia se jouduttiin vaihtamaan C++-versioon kirjastossa. Samalla vaihdettiin ohjelmointiympäristö Qt:hen, jolla myös ohjelman käyttöliittymä oli tarkoitus toteuttaa.

Työn alussa tiedettiin sen olevan laaja ja haastava ja tästä syystä koetettiin varata eniten aikaa niille alueille, joiden uskottiin vievän sitä eniten. Työssä ei päästy tavoitteeseemme rakentaa prototyyppiä laitteesta työn laajuuden ja ajan loppumisen vuoksi. Työstä tuli kattava esiselvitys siitä, kuinka apulaite voitaisiin toteuttaa ja kuinka se toimisi. Työssä käytiin läpi kuinka päästä alkuun OpenCV:llä ohjelmoinnissa Pythonilla sekä Qt:llä. Työssä kerrottiin miten apulaitteen voi rakentaa edullisesti eritavoin.

Asiasanat: konenäkö, ohjelmointi, suunnittelu, ladontakone, servotekniikka.

ABSTRACT

Kemi-Tornio University of Applied Sciences, Technology

Degree Programme	Information Technology
Names	Juha Jakunaho, Panu Rikula
Title	Installation Tool to Dima FP-600 Pick&Place System for BGA-Components
Type of Study	Bachelor's Thesis
Date	13 March 2012
Pages	205 + 19 appendices
Instructor	Tapani Ruokanen, MSc, Eng
Company	Kemi-Tornio University of Applied Sciences
Supervisor from Company	Antti Niemelä, Project Manager

The task was to design and engineer an accessory equipment for BGA surface mounted device component installation to Dima FP-600, manual pick & place system that is used in Embedded Systems Laboratory of R&D unit in Kemi-Tornio University of Applied Sciences. The right installation place would be defined by using machine vision and pick & place head would be moved by motors.

The definitions were that the digital image processing had to be done by using an OpenCV 2.3.1, which is an open source image processing library. The accuracy of the device was set to 0.5 mm and it had to be made as cost-effective as possible. The system had to be made so that it would not obstruct the device's normal use.

The work began by a feasibility study which gave an idea how the device would function. After the feasibility study the task was to study how and with what the device was going to be engineered. The devices that would suit the system best were too expensive to use so alternative ways had to be found to replace them.

A PC with Ubuntu version 10.04.4 was used to program OpenCV. The programming of the digital image processing began with library that was compiled to Python programming language, but there were so many bugs in the current version that it had to be exchanged to a library version that was made by using C++. Qt SDK was used to give an easy way of implementing OpenCV code and GUI.

At the beginning of the work it was known that it would be wide and challenging, so more time was reserved for those areas that were more challenging. Even though that was done, the goal was not achieved because the time ran out. The thesis was changed to a guide on how one can engineer such a system as cost-effectively as possible, how to begin with OpenCV by using Python or Qt and on different variations of the way how the system could be engineered.

Keywords: machine vision, programming, designing, pick and place machine, servotechnique.

SISÄLLYSLUETTELO

KÄYTETYT MERKIT JA LYHENTEET

ADC	Analog-to-Digital Converter, A/D-muunnin
BGA	Ball Grid Array, pintaliitoskomponentti, jossa pallomaiset johtimet ovat riveissä komponentin alla
Bugi	Ohjelmointivirhe
CNC	Computerized Numerical Control, tietokoneistettu numeerinen ohjaus
ESD	Kipinäpurkaus, electrostatic discharge, on sähkö staattinen purkaus
Flux	Juote
Fotoni	Valokvantti, sähkömagneettisen vuorovaikutuksen välittäjähiukkanen
FPGA	Field Programmable Gate Array, digitaalinen mikropiiri, minkä logiikka voidaan ohjelmoida uudelleen
fps	Frames/sec, kuvaa sekunnissa.
HD	High-definition, korkealaatuinen
I/O	Input/Output, syöttö/tulostus
I/O laite	Informaatiota käsittelevä aite, joka kommunikoi toisten I/O laitteiden tai ihmisen kanssa
Karuselli	Pyörivä ympyränmuotoinen levy
Kontrasti	Kohteen erottuminen taustasta
Parametri	Arvo, jota muuttamalla saadaan aikaan muutoksia systeemissä
PCB	Printed Circuit Board, kytkentäalusta, joka on valmistettu
Rework	Rikkinäisen komponentin korvaus uudella
SDK	Software Developement Kit
SMD	Surface Mounted Device, pintaliitoskomponentti
Stream	Streamaus, suoratoisto

1. JOHDANTO

Opinnäytetyön aiheena oli suunnitella BGA-kohdistustyökalu sulautettujen järjestelmien tutkimusryhmän (ProPal2) laboratorion Dima FP-600 ladontakoneeseen. Laitteella pystyttäisiin parempaan tarkkuuteen komponenttien ladonnassa kuin käsin ladottaessa. Työn aiheen saimme ollessamme työharjoittelussa Kemi-Tornion Ammattikorkeakoulun TKI:n Propal-projektissa. Suunnitteilla oli piirilevy, missä käytettäisiin BGA-moduulia ja komponentin asentamiseen mietittiin työkalua, sillä sitä ei voitaisi asentaa tarpeeksi tarkasti sulautettujen järjestelmien laboratorion ladontakoneella. Vaikeaksi BGA-komponentin latomisen tekee juotosten tuleminen komponentin alle, minne ei näe ilman röntgenlaitteita. Olimme etsineet jo jonkin aikaa opinnäytetyön aihetta ja ehdotimme asiaa projektin johtajalle Antti Niemelälle.

TKI:llä tarkoitetaan Ammattikorkeakoulujen Tutkimus, Kehitys ja Innovaatiotoimintaa, millä tähdätään alueen kilpailuedun parantamista tutkimuksen sekä kehittämisen avulla, samalla se tukee alueen työelämää ja edistää aluekehitystä. Kemi-Tornion Ammattikorkeakoulun tekniikan TKI:n pääpainopisteitä ovat optinen mittaustekniikka, materiaalien käytettävyys sekä kunnossapito. Näiden ympärille on rakennettu kattava tuki, kuten Sulautetut järjestelmät sekä T&K laboratorio.

Propal (ProPal2) on Euroopan unionin Euroopan aluekehitysrahaston (EAKR) rahoittama hanke, mikä kuuluu TKI:n sulautettujen järjestelmien alle. Propalin tavoitteita ovat mm. T&K toiminnan sekä opetuksen kehittäminen.

Työn alkaessa tiedettiin aiheen olevan laaja ja haastava, mutta se toisi meille kokemusta sekä tietämystä. Työ aloitettiin esiselvityksellä, missä tutkittiin jo olemassa olevia tekniikoita ja laitteita BGA-komponentin asennukseen. Esiselvityksessä löydetyt tekniikat: BGA-rework asemat, CNC-koneet, ladontakoneet sekä kotitekoiset ladontakoneet toimivat suunnitelmien lähteinä ja antoivat ideoita millaisen laitteen suunnittelisimme itse. Esiselvityksen aikana tutustuttiin sekä asennettiin myös avoin kuvankäsittelykirjasto OpenCV Propalin PC:lle, mihin

oli asennettu entuudestaan Ubuntu käyttöjärjestelmän versio 10.04.4, millä työn ohjelmointia myös toteutettiin. OpenCV:llä tuli toteuttaa kuvankäsittely työssä.

Laitteen tarkkuudeksi saimme sulautettujen järjestelmien insinööriltä Mika Mörtiltä 0,5 mm, mikä tulee BGA-komponenttien itse korjauksesta.

Opinnäytetyön antaja halusi toteuttaa työn mahdollisimman kustannustehokkaasti, koska laitteella tultaisiin harvoin suorittamaan BGA-komponentin asennus. Laite ei myöskään saanut häiritä koneen normaalia käyttöä, joten sen tuli olla helposti irroitettavissa ja pois normaalin käytön tieltä.

Toteutustavan selviämisen jälkeen ryhdyttiin tutkimaan millaisin laittein työ toteutettaisiin: mitkä moottorit sopivat parhaiten, millä moottoreita ohjataan, millaiset kamerat tarvittaisiin, millainen valaistus sopisi työhömme parhaiten. Tutkiminen aloitettiin markkinoilta löytyvien laitteiden tutkinnalla sekä haastattelemalla Kemi-Tornion AMK:n Tekniikan TKI:n Optisen mittauksen laboratoriossa projekti-insinööri Jukka Leinosta. Leinosen mukaan 0,5 mm tarkkuus ei ollut vaikea saavuttaa. Saimme Leinoselta lainaan web- sekä konenäkökameran OpenCV:n testausta varten. Käden ohjaukseen etsittiin esimerkkiä CNC-koneista, sekä laitteista, joita optisen mittauksen laboratoriossa oli valmistettu.

Opinnäytetyö suoritettiin parityönä, missä molemmilla oli omat vastuualueet. Palavereita pidettiin viikoittain pitkin opinnäytetyötä. Katselmointeja ohjaajien kanssa pidettiin harvemmin.

2. TÄRKEITÄ TAUSTOJA

Tässä kappaleessa käydään läpi tärkeitä taustoja asennettavaan komponenttiin ja käytettävään ladontakoneeseen liittyen.

2.1. Ball Grid Array - BGA

Pintaliitoskomponentti

Pintaliitoskomponentti on elektroniikan komponentti, jonka kotelo tulee samalle puolelle piirilevyä nastojen ja näiden juotosten kanssa. Nämä ovat läpiladottavien komponenttiin (missä juotos on eripuolella piirilevyä kuin komponentin kotelo) verrattaessa paljon pienempiä ja niitä voidaankin asentaa molemmille puolille piirilevyä. Kuvassa 1 on esitetty erilaisia pintaliitoskomponentteja.

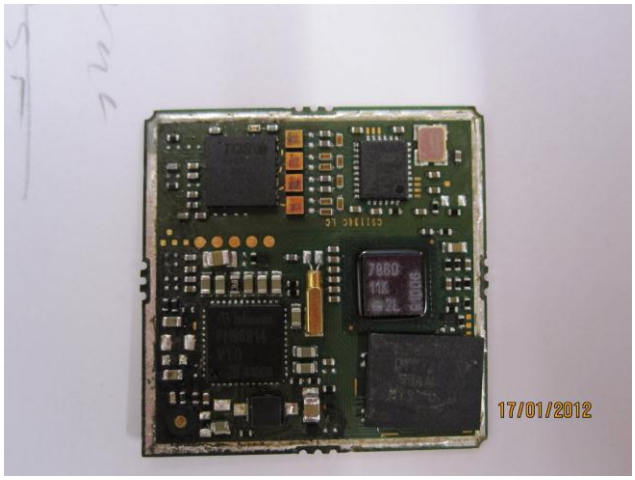
Kuva 1. Erilaisia pintaliitoskomponentteja /23/

BGA

BGA, eli Ball Grid Array, on piirilevylle asennettava pintaliitoskomponentti. BGA-komponentteja löytyy yleisemmin puhelimista sekä tietokoneista. Yleisempiä käyttökohteita ovat muistit sekä prosessorit. /4/, /6/, /177/, /178/

Toisin kuin mikropiirit, joissa liitosnastat ovat komponentin sivuilla, BGA:ssa kaikki nastat ovat komponentin alapuolella matriisimaisessa kuviossa, josta komponentin nimi tulee. Tällä tavoin saadaan kokoonsa nähden enemmän nastoja kuin tavallisessa mikropiirissä ja täten enemmän toiminnallisuutta. Tästä syystä niiden suosio on jatkanut kasvuaan komponenttien ja laitteiden pienentyessä. /2/, /4/, /59/, /177/, /178/

Alkuliite BGA:n edessä usein kuvastaa joko kotelon materiaalia, kuten CBGA, Ceramic-BGA, tai kokoa, kuten µBGA, micro-BGA. Komponenttien pienentyessä tarjolla on myös BGA-moduuleja, jotka ovat kuin pieniä piirilevyjä BGA-alustalla. Tällä tavoin voidaan valmistaa ja myydä toiminnallisuuksia, jotka voidaan asentaa erilliselle piirilevylle. Kuvassa 2 on esitetty BGA-moduuli, mikä sisältää useita eri komponentteja. Moduulissa keskellä oikealla on flip-chip BGA-komponentti. /1/, /51/, /65/, /154/, /160/, /177/, /178/



Kuva 2. Avattu Telit GSM-modeemi

.

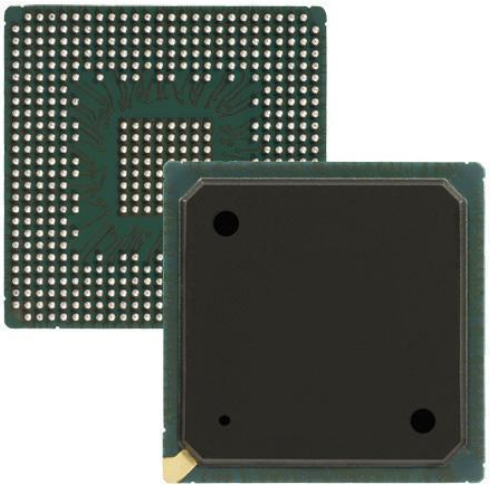
2.1.1. BGA:n ominaisuuksia

Hyviä ominaisuuksia

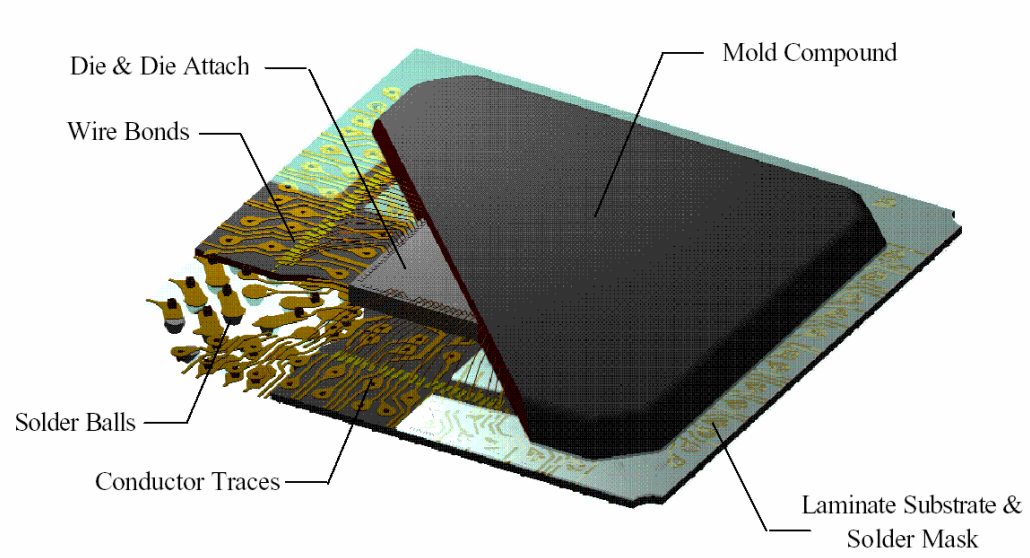
- Erittäin hyvä lämmönjohtavuus: kotelon sisällä syntyvä lämpö johtuu piirilevylle nopeammin ja tehokkaammin kuin muissa mikropiireissä. Tämä vähentää ylikuumentumisen riskiä. /3/, /4/, /6/, /9/, /59/, /107/, /108/
- Pieni pinta-alavaatimus: nastoja mahtuu enemmän komponentin alapuolelle kuin reunoille. /9/, /59/, /108/, /154/
- IC-piirien nopeuden hyödyt: kotelon sisäiset johtimet ovat lyhyempiä. /3/, /9/, /107/, /108/, /154/
- Kestävämpiä ja helpompia käsitellä ennen asennusta ja asennuksen aikana: miltei mahdoton saada irti ilman kuumennusta. Ei ole nastoja, jotka voisivat vioittua. /3/, /6/, /9/, /65/, /107/, /108/

Huonoja ominaisuuksia

- Alttiita kosteudelle ja hankaussähkölle: mikäli komponentin alle pääsee kosteutta, se saattaa aiheuttaa ei-haluttuja kytköksiä pallojen välille. /108/
- Pienen lämmönvastuksen takia BGA-komponentteja ei kannata käyttää ankarissa olosuhteissa tai paikoissa, missä on suuria lämpötilojen vaihtelua. /108/
- Juotosten tutkiminen asennuksen jälkeen erittäin hankalaa ja kallista, sillä näissä joudutaan käyttämään röntgen- tai ultraäänilaitteita komponentin alle näkemiseen. /2/, /3/, /4/, /6/, /59/, /108/



Kuva 3. PBGA, Plastic-BGA /101/



Kuva 4. BGA:n perusrakenne, missä erottuu piisiru koteloinnin alta, sekä pallomaisen matriisin johteet /110/

Kuvassa 3 ja 4 esitellään PBGA-komponentti ja BGA:n rakenne tarkemmin. Kuvassa 5 on taulukoita eri tyyppisten BGA-komponenttien juotospallojen suositeltuja kokoja.

Table 3. Recommended Pad Sizes for SMD & NSMD Pads

BGA Pad Pitch	BGA Pad Opening (A) (mm)	Solder Ball Diameter (B) (mm)	Recommended SMD Pad Size (mm)	Recommended NSMD Pad Size (mm)
1.27 mm (Plastic Ball Grid Array (PBGA))	0.60	0.75	0.60	0.51
1.27 mm (Super Ball Grid Array (SBGA))	0.60	0.75	0.60	0.51
1.27 mm (Tape Ball Grid Array (TBGA))	0.60	0.75	0.60	0.51
1.27 mm (flip-chip) *(1)*	0.65	0.75	0.65	0.55
1.00 mm (wirebond) *(1)*	0.45	0.63	0.45	0.38
1.00 mm (flip-chip) *(1)*	0.55	0.63	0.55	0.47
1.00 mm (flip-chip) *(1)* APEX 20KE	0.60	0.65	0.60	0.51
0.80 mm UBGA (BT Substrate)	0.4	0.55	0.4	0.34
0.80 mm UBGA (EPC16U88)	0.4	0.45	0.4	0.34
0.50 mm MBGA	0.3	0.3	0.27	0.26

Note to Table 3:

(1) FineLine BGA® packages that use flip-chip technology are marked "Thermally Enhanced FineLine BGA" and wirebond packages are marked "Non-Thermally Enhanced FineLine BGA" in the *Altera Device Package Information Data Sheet*.

Kuva 5. Suositeltuja mittoja eri BGA-valmisteille /9/

2.1.2. Komponentin asennus

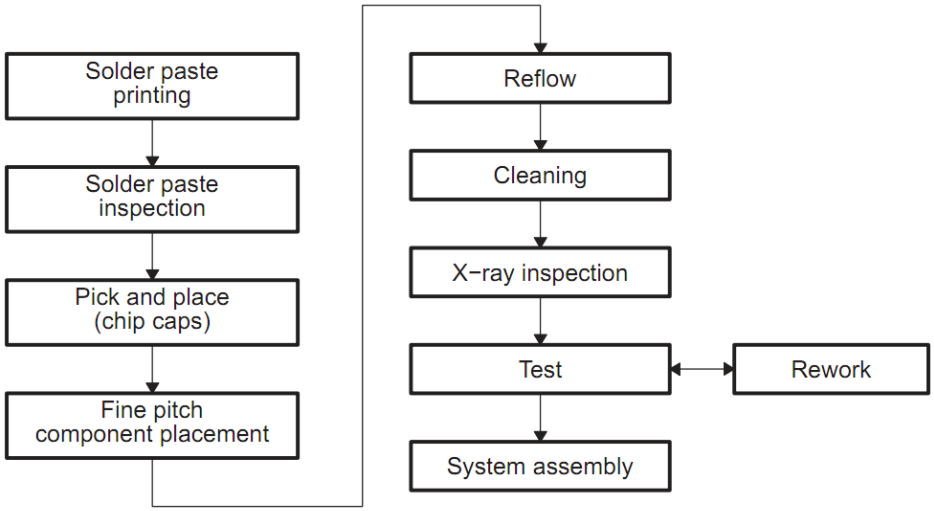
BGA-komponentin asennukseen haasteellisuutta tuo juuri se, miksi se on hyvä ja jatkuvasti suositumpi elektroniikkateollisuudessa. Nastojen ollessa komponentin alla, ei voida perinteisin laittein varmistaa että ovatko juotospallot kohdallaan ja onko komponentti juottunut oikein.

Asennusta helpottaa se, että juotteen pintajännityksen vuoksi pallot vetävät itsensä oikeille paikoilleen kohdistusvirheen ollessa jopa 50 % koteloinnista riippuen. Pintajännitys pyrkii keskittämään pallot liitospintojen kanssa. /160/, /163/

BGA-komponentin voi asentaa joko korjausasemassa tai latomalla pick-and-place koneella. BGA:ta ei voi juottaa kolvilla piirilevyyn, vaan tarvitaan esimerkiksi voimakasta infrapunaa tai kuumaa ilmaa. /2/

Ainut tapa todeta ensimmäisen nastan paikka on katsoa asennusmerkistä tai komponentin kulmista. Nasta on merkitty yleensä pallomaisella merkillä joko komponentin päälle tai alle tai pyöristetyillä kulmilla. BGA:n kotelon materiaalit ovat erilaisia, mikä tulee ottaa huomioon valaistuksessa. Jotkin materiaalit heijastavat runsaasti valoa takaisin. /2/, /65/

Figure 21. Typical SMT Assembly Process Flow for Flip Chip BGA Packages



SPRU811 *Flip Chip Ball Grid Array Package* 35

Kuva 6. Asennusvuokaavio /160/

Kuvassa 6 on esitelty tyypillinen asennusprosessi Flip Chip-komponentille.

2.1.3. Kohdistusmerkit

Kun kohdistus tapahtuu kameralla, piirilevylle ja isoille komponenteille tarvitaan kohdistusmerkkejä. Komponentit voidaan asentaa suhteellisen tarkasti pelkkiä CAD-tietoja käyttämällä, mutta piirilevyjen valmistustavasta riippuen ne eivät ole aivan samanlaisia piirilevyn kanssa. Muutokset ovat pieniä, mutta tarkassa ladonnassa ne voivat aiheuttaa ongelmia. Kohdistusmerkkejä on pääasiassa kolmenlaisia: /162/, /164/, /168/

- aihioon tulevat kohdistusmerkit (Panel Fiducials)
- piirilevyn alueelle tulevat kohdistusmerkit (Board Fiducials)
- komponenttien kohdistusmerkit (Component Fiducials).

Aihioon tulevien kohdistusmerkkien tarkoituksena on lisätä tarkkuutta. Ne sijaitsevat piirilevyjen ulkopuolella ja tämä riittää yksinkertaisimmille komponenteille. Piirilevylle sijoitettavat kohdistusmerkit tulevat vastakkaisiin nurkkiin, mahdollisimman kauas toisistaan. Näitä käytetään yleensä kolmesta neljään kappaletta korjaamaan valmistuksessa tapahtuneita virheitä, kuten piirilevyn venymisiä. Todella isojen ja vaativien komponenttien, kuten BGA:n, ladontaan käytetään omia kohdistusmerkkejä. Ne ovat komponentin vastakkaisissa nurkissa ja niiden tulisi olla mahdollisimman erillään ja erilaisia komponentin nastoihin nähden, jotta ne eivät sekoittuisi. Yleisesti käytetään neljää eri merkkiä: /162/, /164/, /168/

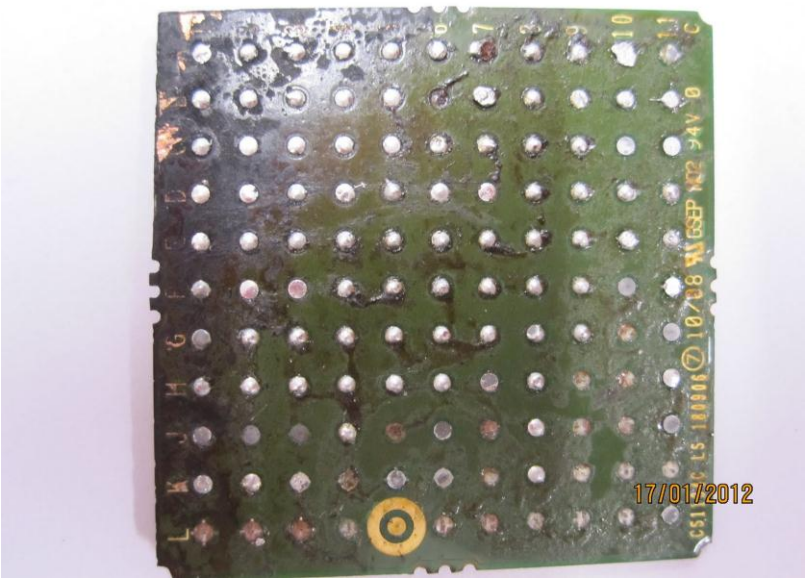
- ympyrä
- neliö
- suorakulmio
- ”+”-merkki.

Merkit ovat kuparia ja ne tehdään nastojen ja johteiden kanssa samassa prosessissa. Tästä johtuen vääristymät sekä virheet ovat suhteessa samanlaiset asennusmerkeillä sekä nastoilla, mikä tekee siitä varmemman tavan kuin esimerkiksi merkintäpainatuskerrokseen (silk-screen) piirretyt merkit. Merkin ympärillä tulisi olla tyhjää vähintään kaksi kertaa merkin verran, jolloin se erottuu taustasta selvästi. Merkit ovat kooltaan yleensä noin 1-3 mm ja sijaitsevat noin 2 mm piirilevyn reunoista. /162/, 168/

2.1.4. Rework

BGA:n korjaukseen tarvitaan erilliset laitteet. Tämä voi olla hankalaa ja kallista, koska juotoksen alapuolelle ei näe ilman läpivalaisua. Jos laite ei toimi oikein tai voidaan epäillä, että juotos on epäonnistunut, komponentin voi irrottaa kuumentamalla sitä niin, että juotokset sulavat. Komponenttia kuumennettaessa täytyy huomioida, ettei kuumenna muita osia levystä. Kuumennuksen jälkeen komponentti voidaan irrottaa. Tämän jälkeen komponentti ja komponentin paikka putsataan juotoksesta liuotteella (fluxilla). /2/, /177/, /178/

BGA-moduulia irrotettaessa ei voida komponenttia itsessään lämmittää, sillä sisällä olevat juotokset voivat vaurioitua, kuten kuvassa 7.



Kuva 7. Irrotetun BGA-moduulin pohja on pahoin vaurioitunut

2.2. Ladontakoneet

Kappaletta on rajattu siten, että keskitytään erityisesti Dima FP-600:seen liittyviin sekä yleisiin asioihin.

Ladontakone on elektroniikassa pintaliitoskomponenttien asentamiseen piirilevylle suunniteltu apukone ja usein linjaston kallein laite. Yleensä sen kustannukset ovat 50 % koko tuotantolinjan laitteiden kustannuksista. Komponenttien pienentyessä koko ajan tarvitaan apuvälineitä, joilla komponentit saadaan ladottua nopeasti ja tarkasti. Ladontanopeudesta puhuttaessa, konetyypistä riippuen, liikutaan tuhansista yli sataan tuhanteen komponenttia tunnissa, tarkkuuden ollessa vain joitain kymmeniä mikrometrejä. Parannettaessa tarkkuutta ei tarvitse käyttää aikaa enää niinkään paljon virheiden etsintään ja korjaukseen, missä säästyy aikaa ja rahaa. /43/, /146/, /163/

Ladontakoneet voidaan jakaa manuaali-, puoliautomaatti- ja automaattiladontakoneisiin. Puoliautomaattisissa on yleensä mahdollisuus ohjattuun komponenttien ladontaan ja automaattinen toimii yleensä itsenäisesti ilman valvontaa. /15/, /146, s.82./

Ladontakoneet voidaan vielä jakaa eri ryhmiin ladontatavan mukaan Pick-and-Place ja Chip Shooteriin sekä komponenttien poiminnan mukaan. Mitä automaattisempi ja nopeampi systeemi on, sitä enemmän korostuu syöttäjien rooli laitteistossa. /146, s.82./

Manuaaliset ladontakoneet

Manuaalisella ladontakoneella tarkoitetaan käsin ohjattavaa ladontakonetta. Tällainen ladontakone koostuu usein karusellista tai syöttäjästä, josta komponentit poimitaan, nosto- ja asennuspäästä sekä piirilevyn paikasta. Yleisesti manuaalisia ladontakoneita käytetään prototyyppien, pienten erien valmistamiseen tai yksittäisten komponenttien vaihtoon korjauksessa. /15/, /146, s.79. /

Joihinkin manuaalisiin ladontakoneisiin on saatavilla erilaisia lisälaitteita avuksi ladontaan, kuten suurennuslaseja, mikroskooppeja, työvalaisimia ja kameroita. /15/, /146, s.79./

Pick-and-Place

Pick-and-Place on ladontatapa, jossa noudetaan ja asennetaan komponentin yksi kerrallaan piirilevylle: Poimintapää ottaa komponentin syöttäjältä, kuljettaa sen piirilevylle oikeaan kohtaan ja asentaa komponentin piirilevylle, jonka jälkeen poimintapää palaa jälleen syöttäjän luo uutta komponenttia varten. Nykyään Pick-and-Place ladontakoneita käytetään suurempien tai erikoismuotoisten komponenttien asentamiseen, sillä niillä voidaan asentaa laajoja kirjoja erilaisia komponentteja piirilevylle. /15/, /43/, /146, s.82./, /179/

2.3. Dima FP-600 ladontakone

DIMA FP-600 on manuaalinen ladontakone, joka toimii Pick-and-Place periaatteella: Jokainen komponentti otetaan yksitellen komponenttikarusellista ja asetellaan piirilevylle. FP-600:ssa on komponenttien liikuttelua helpottava Easyglide-varsi, jota on erittäin kevyt ja vakaa liikutella X- ja Y-suunnissa. Varressa on Sensatip-poimintapää, joka aktivoi tyhjiöön perustuvan imun, kun pää koskettaa komponenttia. Kun komponentti on saatu paikalleen, pää lopettaa automaattisesti imun, jolloin komponentti irtoaa päästä. Komponentin ollessa kiinni päässä, sitä voidaan pyörittää 360 astetta oikeaan asentoon. Koneessa on integroitu tinanannostelijayksikkö. Digitaalinen ajastin yhdistettynä tarkkaan ilmanpaineen säätimeen takaa tarkan ja toistettavan tuloksen. Kuvat 8-11 näyttävät minkälainen DIMA FP-600 ladontakone on. /35/, /48/

Kuva 8. Dima FP-600

Kuva 9. Dima FP-600

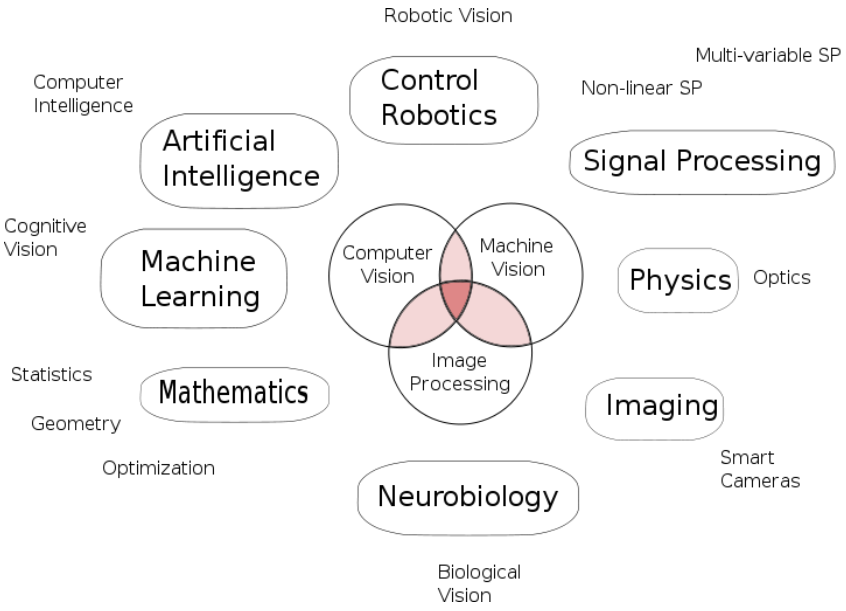
Kuva 10. Dima FP-600 takaa

Kuva 11. Dima FP-600

2.4. Konenäkö

2.4.1. Tietokonenäkö

Konenäöstä ja tietokonenäöstä puhuttaessa on hankala erottaa, kumpi on kyseessä. Tietokonenäkö pyrkii tuomaan "näön" ja "älyn" koneelle, tällä pyritään luomaan itsenäinen kone, jota ei välttämättä tarvitse valvoa koko aikaa. Järjestelmä ei pelkästään ota kuvia, vaan analysoi myös kuvan sisältöä ja kertoo käyttäjälle, mitä kuvassa on tai mitä siinä tapahtuu. Kone tekee muutoksia toimintoihin tietojen perusteella itsenäisesti tai kertoo käyttäjälle muutosten tarpeista. Tällaista järjestelmää voidaan käyttää monissa eri tarkoituksissa, esim. automaattisissa pölynimureissa. Kuvasta 12 nähdään mitä kaikkea tarvitaan tietokonenäössä, konenäössä ja kuvan käsittelyssä, ja mihin näitä voidaan käyttää. /44/, /49/, /180/



Kuva 12. Konenäön, tietokonenäön ja kuvankäsittelyn suhde sekä sovelluskohteita /180/

2.4.2. Konenäköjärjestelmä

Konenäöksi kutsutaan tietokonenäön sovelluksia teollisuudessa. Näiden järjestelmien rakentamiseen tarvitaan laajaa osaamista elektroniikasta, ohjelmoinnista, matematiikasta, optiikasta, valaistuksesta sekä mekaniikasta. Järjestelmässä **keskitytään tiettyyn, rajattuun ongelmaan, johon haetaan ratkaisua konenäön keinoin** ja sitä käytetään mm. mittaamiseen, paikoitukseen, lajitteluun, luokitteluun sekä valvontaan. Sillä pyritään korvaamaan ihminen yksitoikkoisissa töissä, kuten kappaleiden laskennassa liukuhihnalla, havaitsemaan asioita joita ihminen ei kykene havaitsemaan tai tilanteissa, joissa prosessi voi aiheuttaa hengenvaaran. /29/, /44/, /117/

Konenäköjärjestelmä koostuu yleensä: /29/

- kuvattavasta kohteesta
- I/O-laitteista
- kamerasta
- kameran objektiivista
- valaistuksesta
- tietoliikenneyhteyksistä
- kuvankäsittelyprosessorista.

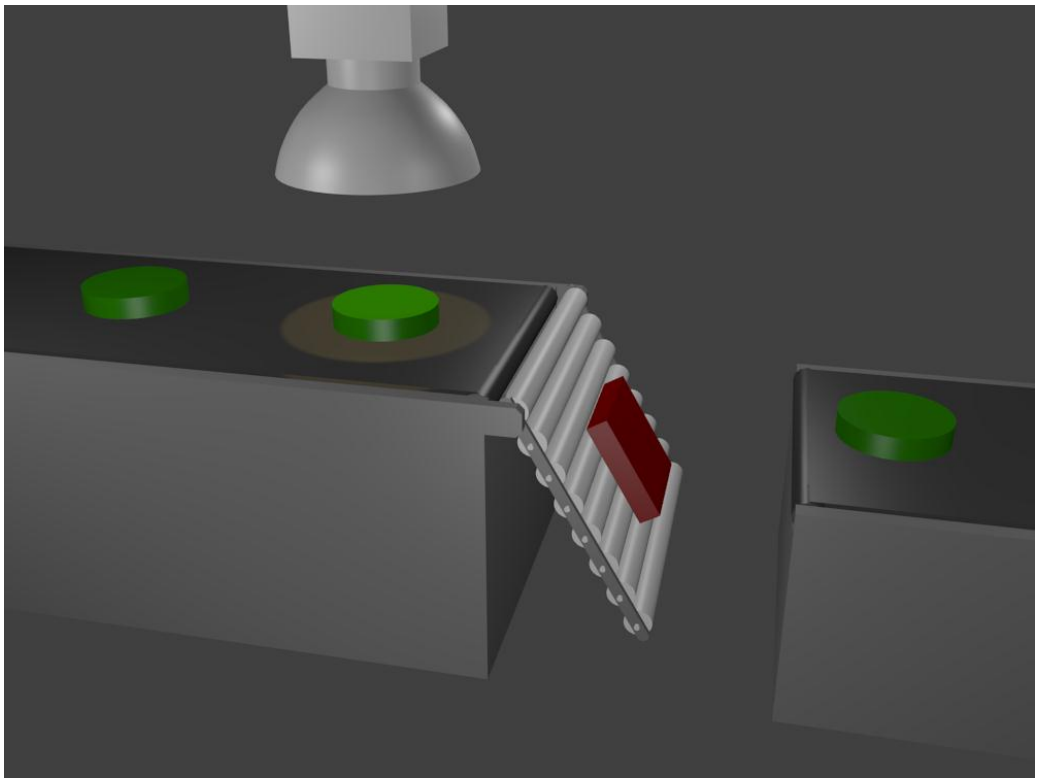
Kamerat ovat yleensä harmaasävykameroita, jotka ovat tarkin, nopein ja edullisin vaihtoehto. Kuvankäsittelyn kannalta prosessit ovat yleensä yksinkertaisia ja nopeita, joiden pohjalta suoritetaan ennalta määrätyt tehtävät. Koska toiminnot ovat ennalta määrätyt, olosuhteiden muuttuessa konenäkö ei pysty sopeutumaan. Juuri tämä on suurin ero konenäön ja tietokonenäön välillä. /29/

Kuvankäsittelyn prosessoreina käytetään tietokoneita, sulautettuja järjestelmiä tai FPGA-piirejä. /29/

On olemassa ns. älykameroita, jotka tarjoavat kameran, optiikan, kuvan prosessoinnin sekä tietoliikenteen kameran ja I/O-laitteiden välille pienessä koossa, mutta eivät välttämättä ole edullisin vaihtoehto. Kameroita, objektiiveja, valaistusta sekä tietoliikenneyhteyksiä käsitellään opinnäytetyössä tuonnempana. /29/

Esimerkki konenäköjärjestelmästä

Kuvassa 13 on esimerkki konenäköjärjestelmästä. Kamera ja valaistus on sijoitettu liukuhihnan päälle, jossa kappaleet kulkevat. Kamera kuvaa kappaleita ja vertaa niitä valmiisiin malleihin. Viallisen kappaleen havaitessa järjestelmä laskee luukun, jolloin kappale poistuu linjastolta. Tämän jälkeen luukku palaa paikalleen ja prosessi jatkuu.



Kuva 13. Esimerkki konenäköjärjestelmästä

2.4.3. OpenCV

Open Source Computer Vision, OpenCV, on alunperin Intelin lanseeraama, BSD-lisenssin alla toimiva funktiokirjasto reaaliaikaiseen konenäköön. Se sisältää yli 500 funktiota kuva- ja videoanalyyseihin C, C++ sekä Python ohjelmointikielillä. /17/, /111/

BSD-lisenssissä sanotaan, että lähdekoodi on "free" eli vapaassa käytössä. Jos siihen tekee muutoksia, tulee se jakaa kaikkien muiden kanssa ja säilyttää lisenssin tekstit lähdekoodissa. Levitettävien ohjelmien lähdekoodeja ei tarvitse jakaa, mikä auttaa kaupallistamaan sovelluksia. Tarvittaessa ohjelman voi julkaista myös eri lisenssillä.

Historia

Idea avoimesta konenäkökirjastosta lähti, kun eräs Intelin työntekijä vieraili eri yliopistoissa ja huomasi, että opiskelijat olivat ajan saatossa luoneet omat avoimet konenäkökirjastonsa, jotka kulkivat vanhalta opiskelijalta uudelle. Tällöin uusien opiskelijoiden ei tarvinnut aloittaa alusta, vaan he pääsivät suoraan opiskelemaan parhaimmilla työkaluilla. /17/, /111/

Taka-ajatuksella, että kasvavat konenäkösovellukset kasvattaisivat nopeiden prosessoreiden tarvetta ja Intel tuottaisi näin enemmän voittoa kuin myymällä erinäisiä ohjelmistoja, Intelin "Performance Library Team" aloitti OpenCV:n vuonna 1999. Ryhmä teki algoritmien perusrakenteet sekä -spesifikaatiot, jotka lähetettiin Vladim Pisarevskyn johtamalle Intelin "Russian Library" ryhmälle. Pisarevsky on vielä tänäänkin keskeisiä OpenCV:n ylläpitäjiä ja kehittäjiä. /17/, /111/

Alfaversio julkaistiin vuonna 2000. Ensimmäinen virallinen versio 1.0 julkaistiin vuonna 2006 ja 2.0 vuonna 2009. 2.3.1 on tämän hetken uusin versio kirjastosta. Uudessa versiossa kaikki osa-alueet on viety yhden kirjaston alle, joten vanhoja dokumentaatioita katsellessa se on hyvä muistaa. Uudistus kohdistui myös joihinkin funktioihinkin ja dokumentointi asiasta on vielä kesken, mutta sitä parannetaan seuraavaan versioon. /17/, /111/

Kirjasto

OpenCV on suunniteltu toimimaan Android-, Maemo-, FreeBSD-, OpenBSD, iOS-, Linux-, Mac OS- sekä Windows-käyttöjärjestelmissä. Tulevaisuudessa kirjasto käännetään myös Java-ohjelmointikielelle sekä Meego-alustalle. Toisin sanoen se toimii Mac-, PC- sekä mobiilialustoilla. /17/, /111/

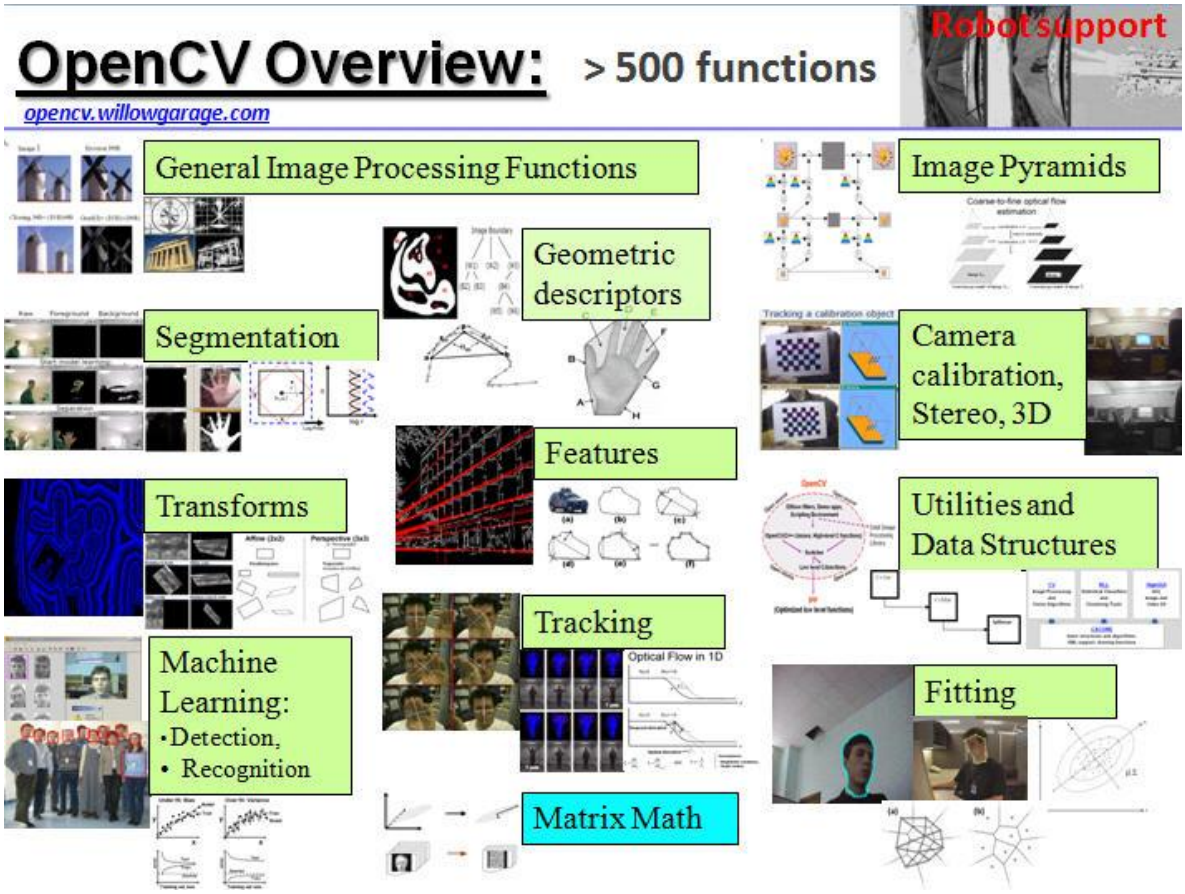
Kirjasto sisältää funktioita mm. histogrammien laskemiseen, kameroiden kalibrointiin, suodatuksiin sekä kuvien väri- ja kokomuunnoksiin. Kirjasto jaetaan moduuleihin seuraavasti: /17/, /111/

- core: ns. runkokirjasto, jossa on datarakenteet, matriisit sekä perusfunktioita, joita muut kirjaston moduulit käyttävät.
- highgui: graafinen käyttöliittymäkirjasto, jonka tarkoituksena on tarjota helppoja työkaluja käyttäjän käytettäväksi, kuten ikkunoita ja vierityspalkkeja.
- video: videokuvan analysointiin tarkoitettu kirjasto. Sisältää mm. kappaleen seuraamiseen sekä kappaleen suunnan määrittämiseen algoritmeja.
- calib3d: mm. yhden sekä kahden kameran kalibrointi sekä 3D-kuvan rakennusta.
- imgproc: kuvankäsittelymoduuli, jossa mm. kuvan suodatus- sekä muunnosfunktioita.
- features2d: funktioita muotojen tunnistamiseen
- objdetect: valmiiksi määriteltyjen kohteiden tunnistukseen, kuten silmien, kasvojen ja autojen tunnistuksiin.
- gpu: graphics processing unit, moduuli NVIDIA:n grafiikkaprosessorien (GPU) ohjelmointiin. GPU ei ole näytönohjain, vaan prosessori, joka on suunniteltu juurikin kuvien käsittelyyn.

Näiden lisäksi on olemassa kirjastoja mm. FLANN (Fast Library for Approximate Nearest Neighbors) sekä Machine Learning. /111/

Yleinen värien järjestys on punainen, vihreä, sininen eli RGB. Värien järjestys OpenCV:ssä on käänteinen yleiseen järjestykseen nähden. /111/, /133/

Kuvassa 14 on esitetty erilaisia käyttömahdollisuuksia sekä sovelluksia OpenCV:stä.



Kuva 14. OpenCV:n käyttömahdollisuuksia /113/

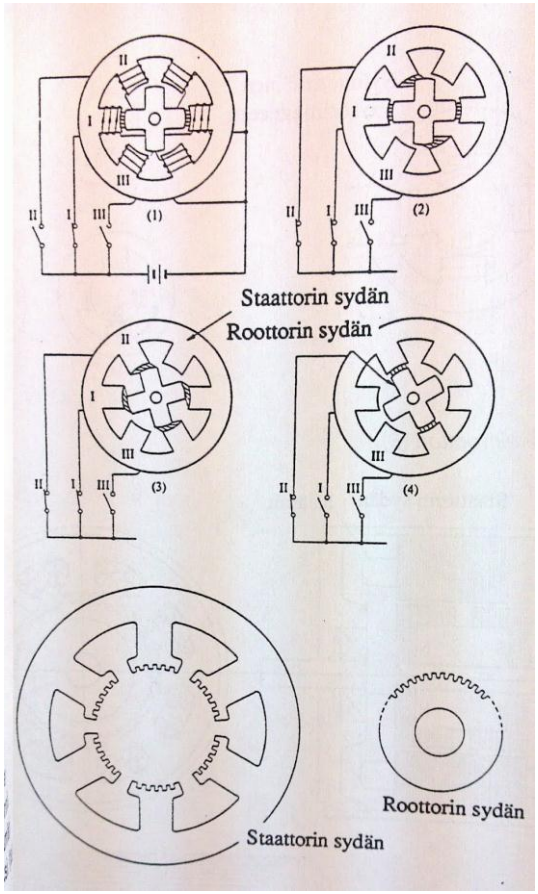
3. LAITTEET

Tässä luvussa käydään läpi moottoreita, moottoreiden tekniikkaa, kameroita, kameroiden tekniikkaa, valaisimia ja valaisutekniikoita. Tarkoituksena on auttaa ymmärtämään ja valitsemaan oikeanlaisia tekniikoita oikeaan työhön.

3.1. Moottorit ja moottoreiden ohjaus

3.1.1. Askelmoottori

Askelmoottoria ohjataan lähettämällä sille digitaalisia pulsseja, jotka moottori muuttaa mekaaniseksi pyörimisliikkeeksi. Pulssimäärä kertoo, montako askelta moottori pyörähtää ja pulssitiheys kertoo, kuinka nopeasti askeleita otetaan. Askelmoottori ei tarvitse takaisinkytkentää, vaan moottori pyörii niin paljon kuin sitä on käsketty. Askelmoottori “hukkaa” askeleet, kun moottori ei jaksa pyöriä, tällöin askelmoottorilta ei saada asematietoa pelkästään laskemalla lähetettyjä signaaleja, vaan tarvitaan erilliset anturit mittaamaan paikkaa. Kuvassa 15 nähdään askelmoottorin tyypillinen rakenne. /8/, /206/, /208/, /209/



Kuva 15. /8, kpl 5, s.29/

Askelmoottori on harjaton ja synkroninen moottori, jonka yksi kokonainen pyörähdys on jaettu askeliin. Normaali tasavirtamoottori pyörii jatkuvasti kun siihen syötetään jännitettä, mutta askelmoottorin pyöriminen on jaettu askeliin. Askelmoottorit jaetaan niiden askeleiden lukumäärän perusteella: moottorit voivat olla 30 asteen tai jopa 0,45 asteen askelmoottoreita. Askeleen pituus saadaan jakamalla koko kierros, 360 astetta, askeleiden määrällä, esim. 200 askeleen moottorin askel on 360/200 = 1,8 astetta. Askelmäärää voidaan kasvattaa muutamalla moottorin askeltilaa. Askelmoottorit voidaan jakaa kolmeen eri tyyppiin: /92/, /206/, /208/, /209/

- muuttuva reluktanssi
- kestomagneetti
- hybridi.

Muuttuvan reluktanssin askelmoottorissa on hammastettu roottori ja käämitettyjä staattoreita. Esimerkiksi kolmivaiheisessa moottorissa on nelihampainen roottori ja kolme paria käämejä. Yksi pari per vaihe. Roottorin liike saadaan aikaiseksi aktivoimalla käämit virtaa syöttämällä, jolloin roottori pyörähtää lähimmän käämin kohdalle. Askelkulma saadaan laskettua jakamalla 360 astetta vaiheiden lukumäärän ja hampaiden lukumäärän tulolla. Kolmivaiheisen moottorin askelkulma on siis 360/(3*4) = 30. Kuvasta 16 nähdään moottorin rakenne sisältä. /11/, /206/, /208/

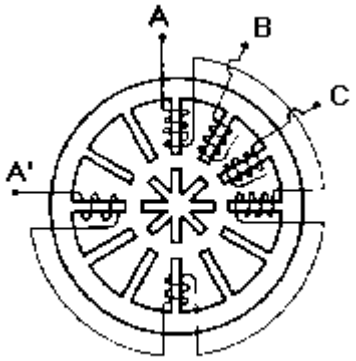
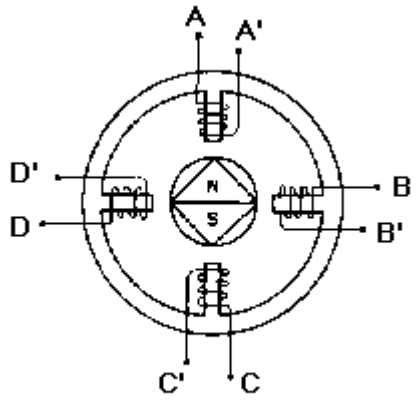


Figure 5: Variable Reluctance Motor

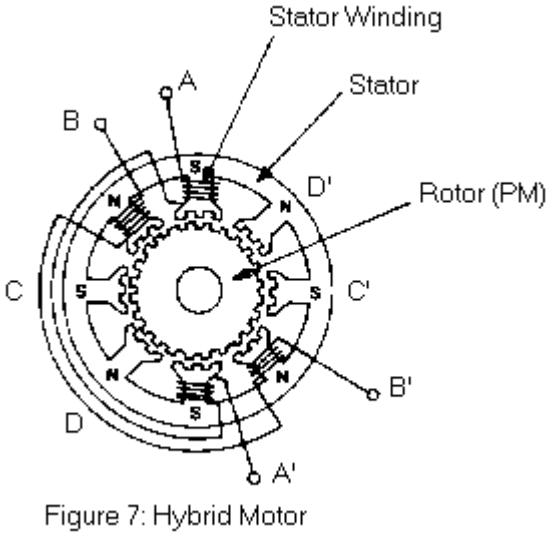
Kuva 16. Muuttuva reluktanssi moottorin rakenne /11/

Kestomagnetoidussa askelmoottorissa roottorissa ei ole hampaita ollenkaan, vaan se on sylinterimäinen kestomagneetti. Roottoria liikutetaan syöttämällä virtaa käämeihin, jolloin roottori kääntyy sitä vetävään magneettisen navan suuntaan. Kuva 17 näyttää moottorin rakenteen. /11/



Kuva 17. Kestomagnetoitu askelmoottori /11/

Hybridiaskelmoottori yhdistää muuttuvan reluktanssin askelmoottorin ja kestomagnetoidun askelmoottorin ominaisuuksia. Roottori on hammastettu kuten muuttuvan reluktanssin moottori, mutta siinä on myös kestomagneetti. Tällä tavalla saadaan entistä tarkempi askelmoottori, mutta nämä maksavat myös huomattavasti enemmän kuin toiset. Normaalisti näiden moottorien askelkulma on 0,9 - 5 astetta. Kuvassa 18 on moottorin rakenne. /11/



Kuva 18. Hybridiaskelmoottori /11/

Askeltilat

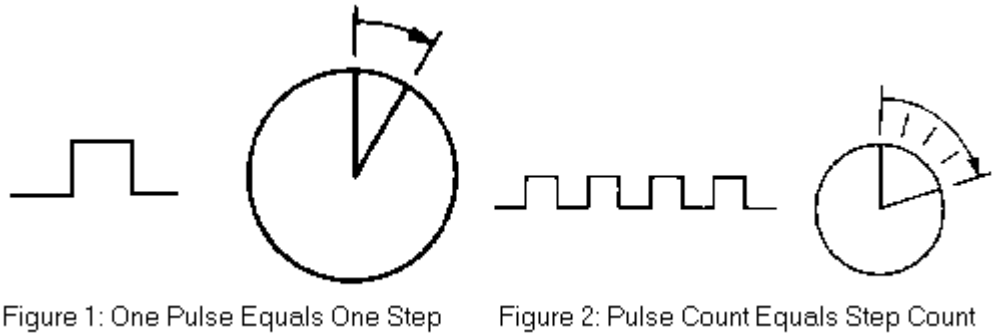
Askelmoottoreissa voidaan valita millä askeltilalla sitä käytetään. Kokoaskellustilassa askeleen koko ei ole kovin iso, mutta se saa suurimman väännön siinä. Tässä tilassa virtaa syötetään yhteen käämi kerrallaan, jolloin roottori pyörähtää tämän käämin kohdalle. Tätä voidaan käyttää myös syöttämällä virtaa kahteen käämiin kerrallaan, jolloin roottori pyörähtää niiden käämien väliin. Kuvassa 19 nähdään tyypillisiä askelmoottoreita. /79/, /92/, /203/, /206/, /207/

Kuva 19. Erikokoisia askelmoottoreita /145/

Puoliaskeltilassa otetaan puolikkaisia askeleita. Tässä käämeille syötetään virtaa eri tahdissa. Ensin syötetään yhteen käämiin virtaan ja seuraavana syötetään kahteen käämiin virtaa. Askelkulma saadaan puolitettua tällä tavalla. /79/, /92/, /203/, /206/, /207/

Mikroaskeltilassa kutakin käämiä ohjataan erikseen, jolloin saadaan aikaiseksi paljon tarkempi liike. Tätä tilaa käytetään yleensä kun tarvitaan mahdollisimman kitkatonta liikettä ja vähän melua. /79/, /92/, /203/, /206/, /207/

Kuvassa 20 on havainnollistettu askelmoottorin toimintaa. Askelmoottoreita ohjataan digitaalisilla pulssisignaaleilla. Jokainen pulssi ohjaa vain yhden askeleen verran. Tällä tavoin voidaan laskea liike ilman takaisinkytkentää, jota servomoottoreissa tarvitaan, mutta silloin ei saada tietoa siitä, että saiko moottori suoritettua kaikki askeleet vai ei. Moottorin pyörimisnopeus on suoraan verrannollinen pulssitaajuuteen. Heikkoutena pidetään sitä, että askelmoottoria käynnistettäessä, ei voida olla varmoja siitä missä asennossa se on.



Kuva 20. Askelmoottoria ohjataan pulssittamalla /11/

3.1.2. Lineaarijohteet

Aktuaattorit ovat laitteita, jotka muuttavat sähköisen energian liikkeeksi. Aktuaattoreilla voidaan saada aikaan lineaarista, pyörivää tai värähtelevää liikettä. Lineaarisia aktuaattoreita käytetään tarkkuutta vaativissa töissä. Lineaarisen liikkeen ansiosta sitä voidaan käyttää monissa eri automaatiota vaativissa tarkoituksissa. Kuularuuveja käytetään liikuttamiseen, hihnakäyttöisiäkin on. DC-lineaariaktuaattorissa DC-moottori pyörittää kuularuuvia, näissä käytetään potentiometria paikan ilmoittamiseen. Kaikissa aktuaattoreissa näitä ei käytetä, joissain malleissa on kytkimet päissä, jotka pysäyttävät liikkeen kun ruuvi on pyörinyt loppuun asti. Tarkan liikkeen vuoksi nämä sopivat hyvin CNC-koneisiin, mutta kallis hinta voi olla monille liikaa. /147/

Aktuaattoreita on saatavilla useita eri tyyppejä ja kokoja. Niitä voidaan käyttää AC-, DC-, askel- tai servomoottoreilla ja niitä voidaan ohjata suoraan sisäänrakennetulla elektroniikalla tai "stand alone" ohjainyksiköllä.

Lineaarijohteita käytetään suoraviivaisen liikkeen tarkkaan ohjaamiseen. Johteet ovat lineaarisia liuku- tai vierintälaakereita. Johteista on saatavilla erilaisia valmiita moduuleita, jotka ovat valmiita asennettaviksi lineaarirakenteisiin. Lineaarilaitteissa käytetään yleisesti kuularuuveja, hammashihna- ja ketjuvaihderatkaisuja muuttamaan pyörivän liikkeen

suoraviivaiseksi liikkeeksi. Kuularuuvi on yleinen vaihtoehto lineaarilaitteissa sen tarkkuuden vuoksi. Kuva 21 näyttää erään valmistajan kuularuuveja. /144/

Kuva 21. Kuularuuveja /144/

Lineaariyksikkö on kokonainen paketti, missä on johde, kuularuuvi- tai hammashihnasysteemi ja monesti myös sähkömoottori. Näiden etuna on pieni koko ja se, että kaikki osat tulevat samassa paketissa, hinnaltaan nämä voivat olla todella kalliita, pituudesta riippuen. Kuvassa 22 ja 23 on esitelty erilaisia ratkaisuja lineaariyksiköille.

Kuva 22. Erilaisia lineaariyksiköitä Tollo-sarjasta: kuularuuvi- ja hihnakäyttöisiä ratkaisuja /28/

Kuva 23. Thomson tarkkuuslineaariaktuaattori /147/

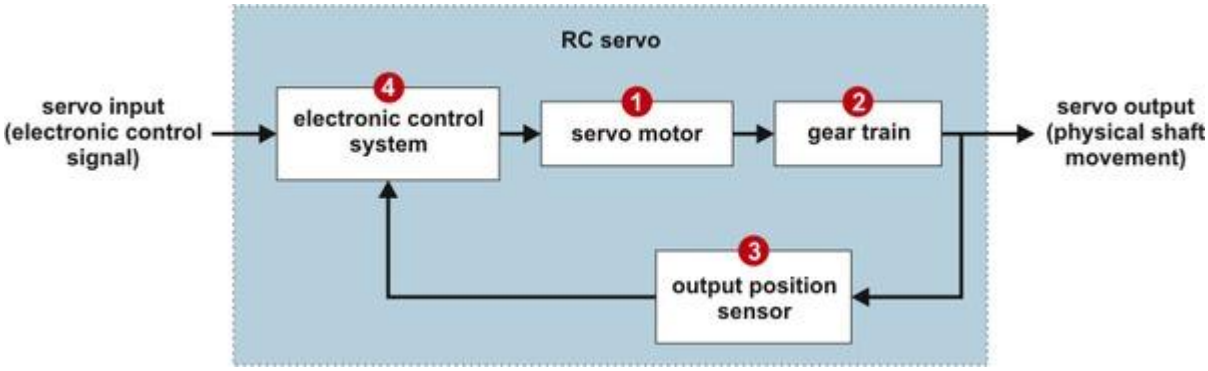
3.1.3. Servot

Servo ei ole sama asia kuin servomoottori. Se on lyhennys servomekanismista. Servomoottorit ovat taas moottoreita, jotka on tarkoitettu servokäyttöön. Servo kertoo, että systeemissä on takaisinkytkentä, jolla pystytään saavuttamaan tai ylläpitämään tiettyä parametria. Itse servosysteemeissä ei ole välttämättä ollenkaan moottoria tai edes mitään liikkuvia osia. Lämmitysjärjestelmät ovat tästä hyvä esimerkki: automaattisella tehonsäädöllä ylläpidetään haluttua lämpötilaa. Tärkein ominaisuus servoissa on takaisinkytkentä eli lähtötila (output) vaikuttaa seuraavaan tulotilaan (input). Tätä kutsutaan suljetun silmukan järjestelmäksi. Takaisinkytkentä saadaan tehtyä mm. potentiometrin avulla. Potentiometri ja servomoottori ovat yhteydessä toisiinsa. Servomoottorin pyöriessä, pyörii myös potentiometri ja tästä saadaan tieto servo-ohjaimelle missä asennossa servomoottorin akseli on. Ohjain vertaa tämän jälkeen moottorille lähetettyä signaalia ja potentiometriltä saatua signaalia keskenään, josta nähdään onko servomoottori suorittanut käskyn halutusti. /130/

Servomoottori

Servomoottori on sähköinen laite, jossa on kiinni akseli, jota voidaan pyörittää lähettämällä servomoottorille signaali. Akselin paikka voidaan määritellä servosta riippuen erittäin tarkasti. Liike annetaan servomoottorille asteina. Signaali lähetetään servomoottorille ja akseli pyörähtää tietyn asteen verran. Joissain servomoottoreissa liike on rajattu esim. 90 astetta tai 180 astetta. Erittäin suurta tarkkuutta vaativiin laitteisiin on olemassa myös 360 asteen servomoottoreita mm. robotiikkaan, mutta näihin pitää paikantunnistusta varten lisätä erillinen anturi. Servomoottoreita käytetään yleensä laitteissa, missä vaaditaan suurta tarkkuutta. Servot koostuvat moottorista, vaihteistosta, ohjauspiiristä ja kotelosta. /130/

Servoon menee kolme johtoa: virta, maa ja ohjaus. Tarvitaan myös erillinen servo-ohjain, jolla servomoottoria voidaan ohjata tarkasti ja jolla saadaan tietoa akselin asennosta. Toisin kuin normaaleissa askelmoottoreissa, jotka eivät lähetä tietoa siitä, missä asennossa ne ovat tai siitä, että saatiinko kaikki askelmat käytyä läpi. Servomoottorista saadaan heti tieto ohjaimelle, jos jostakin syystä komentoa ei saatu suoritettua. Kuva 24 näyttää tyypillisen harrastelijaservolaitteen rakenteen. /130/



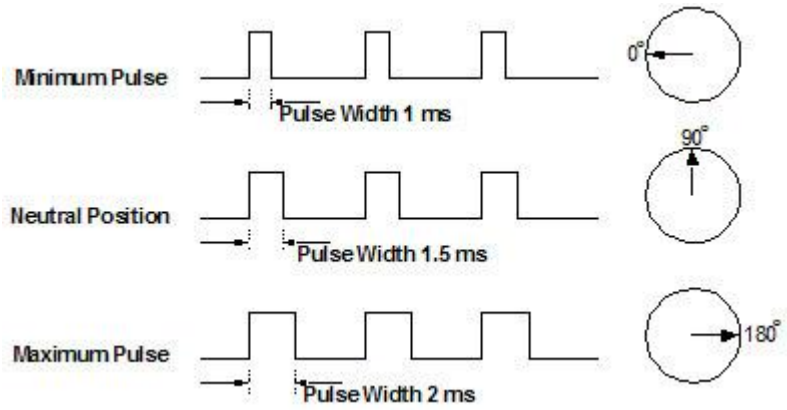
Kuva 24. Perinteinen servo /130/

Servomoottoreita ohjataan Pulse-Width Modulation (PWM, pulssinleveysmodulaatio) signaaleilla, jotka tulevat ulkoiselta servo-ohjaimelta. Ulkoisella ohjaimella lähetetään signaali servomoottorille, joka muuttaa sen liikkeeksi. Servomoottori ja potentiometri ovat yhteydessä toisiinsa. Servomoottorin pyöriessä, siihen kytketty potentiometri pyörii. Potentiometristä saadaan servomoottorin asento servo-ohjaimelle. Takaisinkytkennästä saatua arvoa verrataan servo-ohjaimelta lähetettyyn signaaliin ja näiden erotuksen ollessa nolla, servomoottori pysähtyy. /130/

Servomoottoria ohjataan kanttiaaltopulsseilla. Servomoottorin asento nollataan lähettämällä sinne 1,5 ms pulssi. Tämä voi poiketa isoissa teollisuuskäyttöön suunnitelluissa servomoottoreissa, mutta harrastelijoille suunnatut servomoottorit nollataan 1,5 ms pulssilla. Servomoottori pitää paikkaansa niin kauan kuin sille lähetettävä signaali pysyy

muuttumattomana. Servomoottori pyrkii pitämään asentonsa jonkin aikaa silloinkin, kun ohjaussignaali katkeaa. Ohjaussignaali pitää lähettää vähintään 20 ms - 30 ms välein. /130/

Servo-ohjaimia käytetään, jotta voidaan ohjata servomoottoria helposti ja tarkasti.



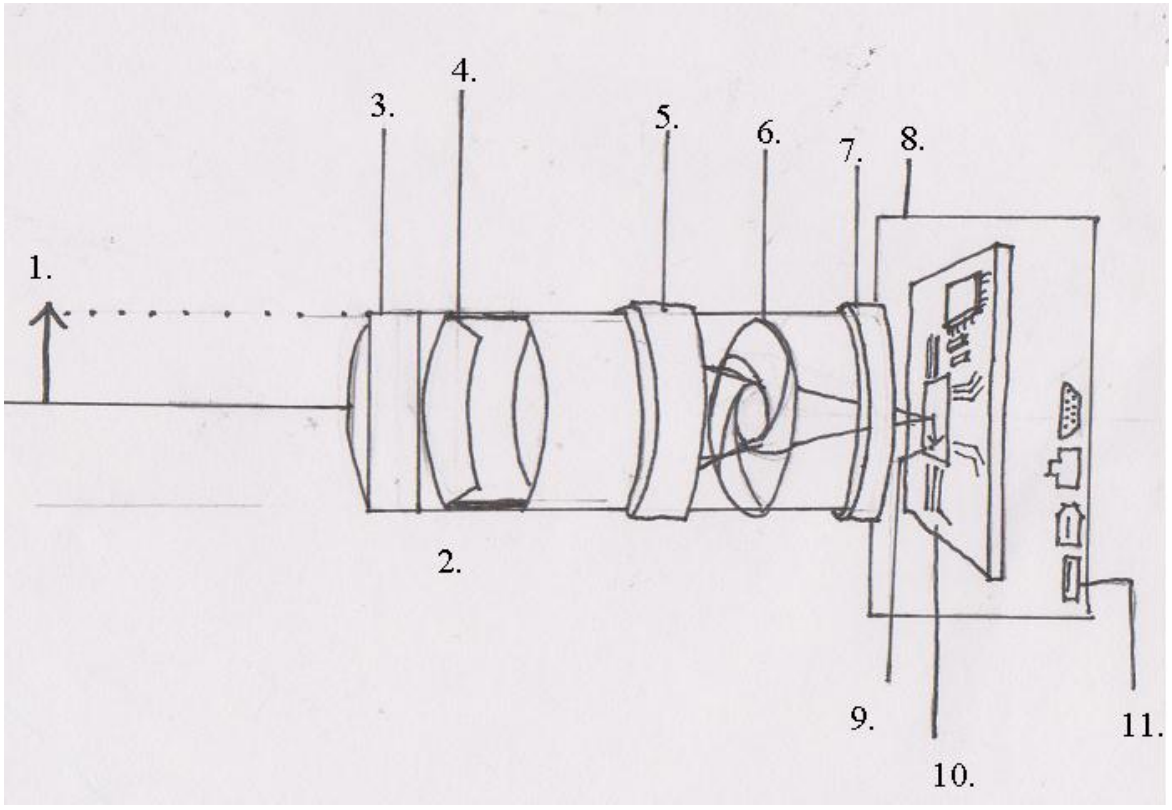
Kuva 25. Servomoottorin ohjaus /213/

Kuvassa 25 kerrotaan servomoottorin ohjaamisesta. Kun pulssileveys on 1 ms, niin moottori kääntyy vastapäivään ääriasentoon. 1,5 ms pulssilla päästään neutraaliin asentoon, tämä voi heitellä hieman, mutta yleisesti se on noin 1,5 ms. 2 ms pulssi kääntää moottoria myötäpäivään ääriasentoon.

3.2. Kamerat

Kamera on konenäön sydän ja se tulee valita aina tapauskohtaisesti. Ne ovat yksi suuri kokonaisuus, osiensa summa. Kaikki liittyvät toisiinsa puhuttaessa kuvan muodostuksesta, eikä yhdestä asiasta voi puhua mainitsematta toista.

Mikäli kamerassa pikseleitä on liian vähän, voidaan saadaan virheellisiä tuloksia. Jos pikseleitä on liikaa tehtävään nähden, kuvankäsittelyyn kuluu enemmän aikaa ja prosessi hidastuu. Myös värikamerat vaikuttavat nopeuteen sekä tarkkuuteen. Kameroissa yleisesti käytetyllä RBG:llä (Red, Blue, Green) on kolminkertaisesti kanavia harmaasävykameraan nähden, jolloin jo pelkästään tiedonsiirtoon menee kolminkertainen aika. /117/



Kuva 26. Kamerajärjestelmä

Kuvassa 26 on esitetty eräs kamerajärjestelmä selventämään, kuinka kamerat ovat osiensa summa. Jokainen järjestelmän osan laatu vaikuttaa kuvaan. Kuvassa:

1. kuvattava kohde
2. objektiivi, linssijärjestelmä
3. suodattimet ja suojat
4. linssi objektiivin sisällä
5. kuvanvakain
6. säädettävä aukko
7. liitin, jolla objektiivi sekä mahdolliset johtimet liitetään kameraan
8. kamera
9. kenno, jolle kuva muodostuu
10. ohjainelektroniikka
11. tietoliikenneyhteyksien liittimet: USB, FireWire, Ethernet sekä sarjaportti.

Tässä kappaleessa käsitellään kameroita, niiden sisältöä ja koetetaan selvittää, miten eri asiat vaikuttavat kuvanlaatuun, mitä valmistajan antamat tiedot tarkoittavat, sekä minkälaista kameraa konenäkösovellukseen kannattaa harkita.

3.2.1. Kennot

Digitaalinen kamera on laite, joka koostuu valoherkästä kennosta, ohjauselektroniikasta, objektiiveista sekä liittimistä ulkoista tiedonsiirtoa varten.

Kennon toiminta perustuu samaan periaatteeseen, millä auringonvalosta tuotetaan aurinkopaneeleissa sähköä. Fotonin törmätessä kennoon se synnyttää sähkövirran, joka ensin muunnetaan elektroneista jännitteeksi, analogiseksi signaaliksi ja tästä jännitetasoksi eli digitaaliseksi signaaliksi A/D-muuntimessa. CMOS- sekä CCD-kennot eivät tarvitse mekaanista suljinta toimiakseen.

Pikseli

Pikselillä (pixel, picture element) on kaksi eri merkitystä, jotka on hyvä käydä aluksi läpi väärinkäsitysten ehkäisemiseksi. Kennoista puhuttaessa sillä tarkoitetaan fotodiodia, pientä puolijohdetta, joka muuntaa osan osuneesta valosta elektroneiksi siis sähköksi. Kuvista puhuttaessa pikseleillä tarkoitetaan kuvan pienimpiä osasia, joilla ei ole fyysistä muotoa, vaan jokin värisävyn arvo. Näitä pikseleitä voidaan lisätä tai poistaa ja näin muuttaa kuvan kokoa. Kuvan pikseli ei aina vastaa kennon pikseliä.

Resoluutio

Resoluutiolla tarkoitetaan tarkkuutta, kuinka tarkasti jotain voidaan mitata tai ilmaista. Kameroissa resoluutioilla tarkoitetaan pikselitiheyttä kennolla. Esimerkiksi standardoidussa VGA:ssa suurin resoluutio on 640 * 480, eli 640 pikseliä leveyssuunnassa ja 480 pikseliä pystysuunnassa, yhteensä 307 200 pikseliä. Pikseleiden suuren määrän vuoksi usein puhutaankin kameroissa megapikseleistä (10^6). Tällöin 307 200 pikselin sijaan sanotaan 0,3 megapikseliä. /126/

Kuvissa resoluutiolla tarkoitetaan periaatteessa samaa asiaa kuin kameroissa, pikseleiden määrää pysty- ja vaakasuunnassa. Pikseleiden kokonaismäärällä harvemmin on merkitystä kuvista puhuessa.

Mitä suurempi kennon resoluutio on, sitä pienempiä yksityiskohtia voidaan havaita eli kuva on sitä tarkempi, mitä enemmän pikseleitä. Nyquistin teoreeman mukaan resoluution on oltava vähintään kaksi kertaa niin suuri kuin vaadittava tarkkuus, mutta osapikselimenetelmällä voidaan päästä pikseliäkin pienempään tarkkuuteen. /83, /121/

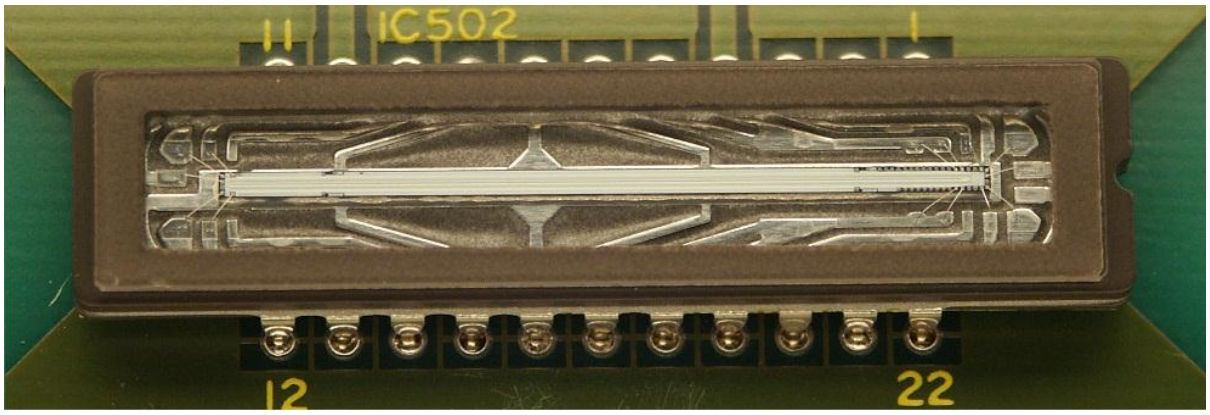
Kuvasuhde

Kuvasuhteesta puhuessa puhutaan pikseleiden suhteessa leveys- ja pituussuunnissa. Esimerkiksi VGA:n 640 * 480 kuvasuhde on 4:3.

Viivakamera ja matriisikamera

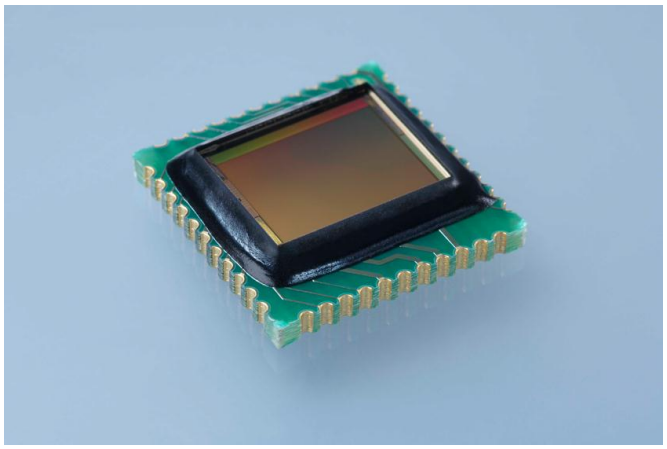
Kameroista lukiessa saattaa törmätä viiva- ja matriisikameroihin tai kennoihin. Tässä kappaleessa käydään lyhyesti läpi näitä käsitteitä.

Viivakamerassa, toiselta nimeltään viivasensorissa, kennon pikselit ovat yhdessä rivissä. Resoluutio määräytyy suoraan rivin pituuden mukaan. Viivakameroita käytetään yleensä kuvaamaan liikkuvia kohteita sekä rakennettaessa 3D-kuvia. Viivakameroita löytyy mm. skannereista sekä pullonpalautusautomaateista. Kuvassa 27 on tyypillinen viivakamera. /29/



Kuva 27. CCD-viivakamera /181/

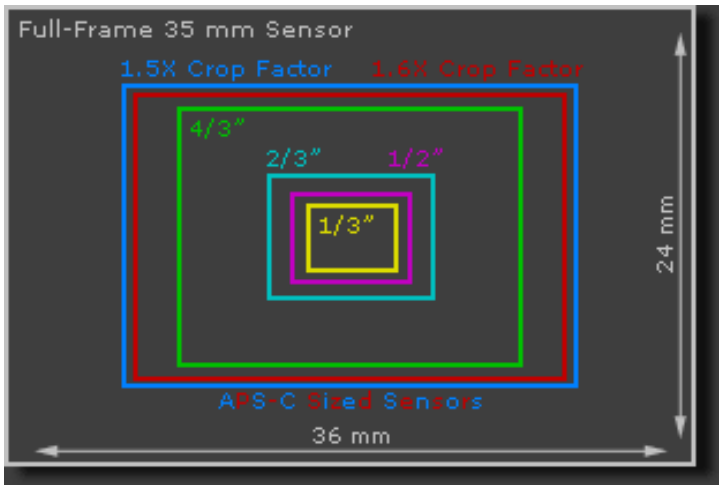
Matriisikameroissa pikselit kennolla on jaettu pysty- ja leveyssuunnassa suorakaiteen muotoisesti. Resoluutio määräytyy pikseleiden määränä näissä suunnissa. Kuvassa 28 on tyypillinen matriisikameran kenno. /29/



Kuva 28. Matriisikameran kenno /14/

Kennon koko

Digitaalikameran kennon kokoa verrataan kameran 35 mm:n filmiin, jonka mitat ovat 36 * 24 mm ja kuvasuhde 3:2. Digitaalikameroissa yleensä kuvasuhteet ovat 4:3 tai 16:9 johtuen videoformaateista. Kuvassa 29 on kennojen koot suhteutettuna toisiinsa. /19/, /62/, /91/



Kuva 29. Kennojen koko verrattuna filmiin /19/

Kennon fyysisesti suurempi koko ei automaattisesti tarkoita suurempia pikseleitä. 4/3” ja 1/3” kennoilla voi olla sama resoluutio, mutta pikselit ovat fyysisesti isompia. Suurempi ala kerää enemmän valoa, mutta pienet kennot ovat suositumpia kamerapuhelimissa ja web-kameroissa. /19/

Pienellä kennolla on suurempi terävyysalue ja usein näiden taustaa onkin hankalampi sumentaa. /19/, /80/

3.2.2. Kennotekniikat

Koska toteutettavat tehtävät ovat erilaisia, ovat myös kuvaamisessa syntyvät haasteet erilaisia. Kameroissa on käytössä kaksi eri kennotekniikkaa: CCD ja CMOS. Molemmilla on omat hyvät ja huonot puolensa, mitkä tekevät niistä toistaan paremman eri tehtäviin. Markkinoilla on tarjolla samanlaisia kameroita eri kennoilla. Tämän kappaleen tarkoituksena on tarkemmin tarkastella CCD- ja CMOS-tekniikkaa ja näiden eroja.

CCD

Charge Couplet Device, CCD, on ensimmäinen valokuvaukseen tarkoitettu digitaalinen kenno. Sen keksivät Willard S. Boyle ja George E. Smith vuonna 1969 Bellin laboratoriossa Amerikassa. Alunperin tarkoituksena oli kehittää tekniikkaa tallentamaan tietokoneiden dataa, mutta tuloksena olikin jotain paljon mullistavampaa. /158/

Kenno koostuu valoherkistä fotodiodeista ja yksi fotodiodi vastaa yhtä pikseliä. Valon osuessa pikseliin, osa valon energiasta muuttuu elektroneiksi, joka kerääntyy pikselin elektronikaivoon (well). Pikseli mittaa vain siihen osuneiden fotonien määrä. Valotuksen eli kuvanoton jälkeen pikseleiden varaukset kuljetetaan kohti “read-out” rekisteriä pikseli kerrallaan, kunnes kaikki on luettu. Rekisteristä jokainen rivi kerrallaan viedään E/V-muuntimen (Electrons-to-Voltage) läpi kohti kameran muuta elektroniikkaa, mitä havainnollistetaan kuvassa 30. Kuva otetaan

kerralla ja kaikki pikselit valottuvat yhtä aikaa. Tämä mahdollistaa liikkuvien kohteiden kuvaamisen ilman häiriöitä. /57/, /62/, /88/, /158/

Kennolle syntyy analoginen kuva ja jotta se saataisiin digitaaliseen muotoon, se kulkee kennolta kameran elektroniikalle, jossa analoginen signaali muutetaan digitaaliseen muotoon. Kenno tarvitsee siis toimiakseen ulkoista elektroniikkaa, josta se saa kaiken tarvitsemansa mm. kellotaajuudet. Kuvassa 30 ulkoinen elektroniikka on kuvassa oikealla. /57/, /88/, /158/

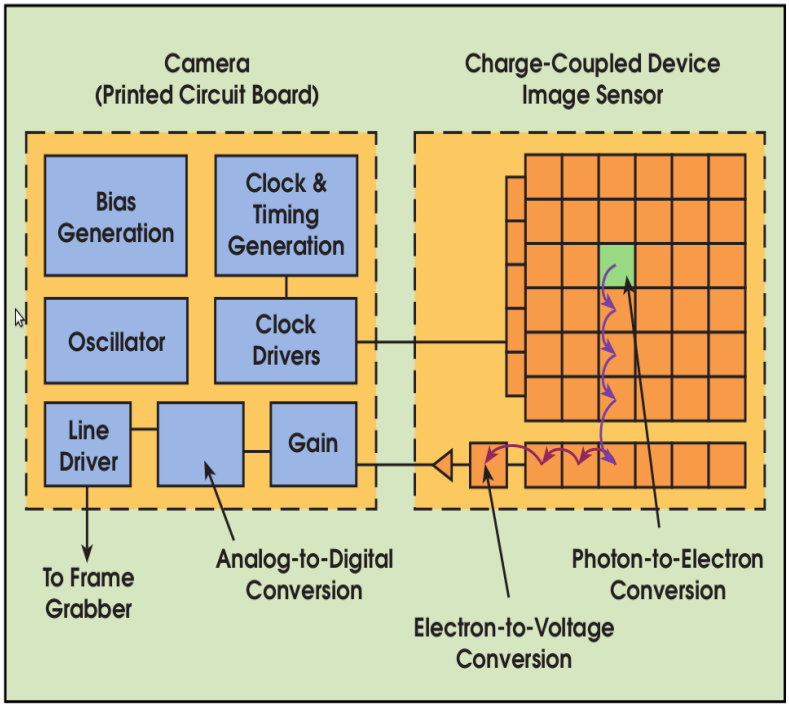


Figure 1. *On a CCD, most functions take place on the camera's printed circuit board. If the application's demands change, a designer can change the electronics without redesigning the imager.*

Kuva 30. CCD:n periaate /88/

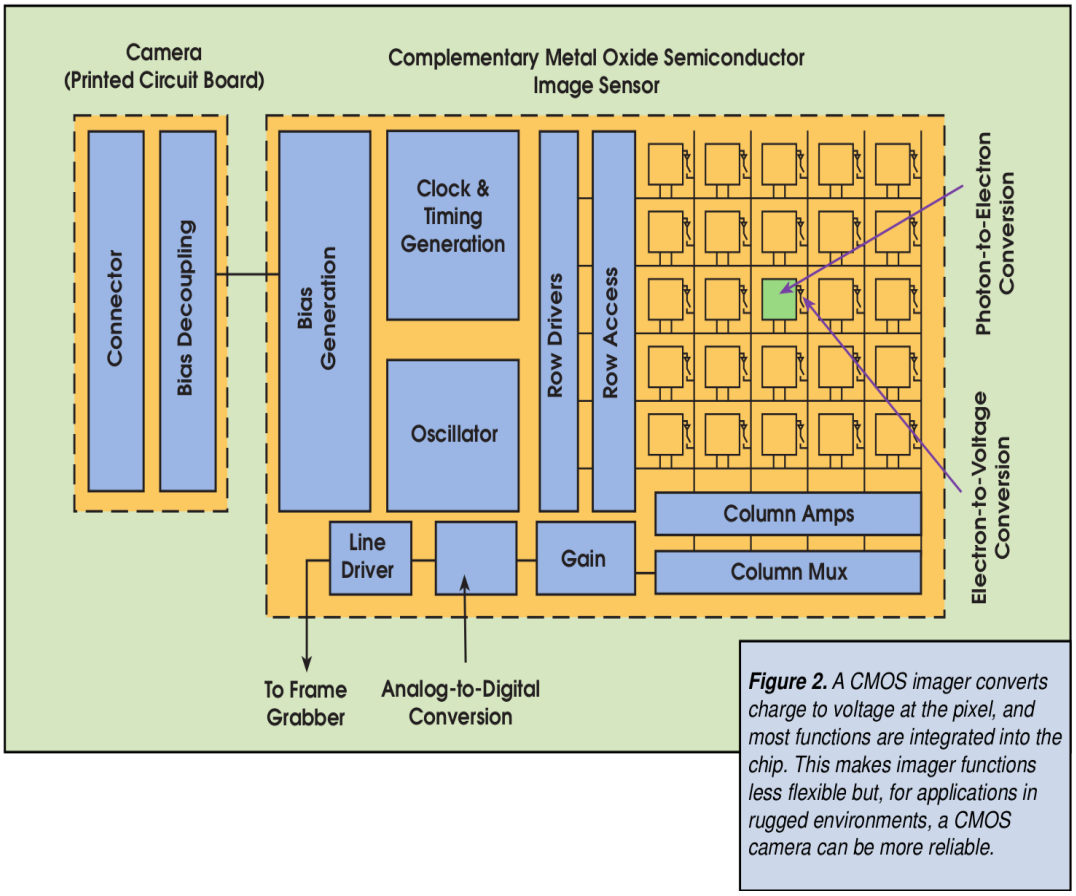
CCD-kennon toimintaa parantaakseen on kehitetty monenlaisia lisäominaisuuksia, kuten siirtorekistereitä, lisäkanavia ja vahvistimia, mutta toimintaperiaate on silti kaikissa samanlainen. /158/

CMOS APS

Complementary Metal Oxide Semiconductor, CMOS. Voidaan kutsua myös CMOS APS, Active Pixel Sensor. Se toimii samalla periaatteella kuin CCD, mutta CMOS-kennon jokainen pikseli itse muuntaa fotodiodiin osuvan valon tuottaman varauksen jännitteeksi ja varastoi sen fotodiodin vieressä olevaan apupiiriin tai muualle kennon elektroniikkaan. Jokaisessa pikselissä on myös nollaus (reset). CMOS-kennot ovat pitkälle integroituja eli niissä on valmiina paljon ominaisuuksia sisäänrakennettuina, kuten A/D-muuntimia. Integroinnista johtuen CMOS-kamerat mahtuvat pienempään tilaan. Kennolta poistuessa kuva on jo digitaalisessa muodossa. Kuvasta 31 voidaan havaita, kuinka suuri osa tarvittavasta tekniikasta on integroitu kennolle verrattuna CCD:n kuvaan 30. /88/, /89/, /158/

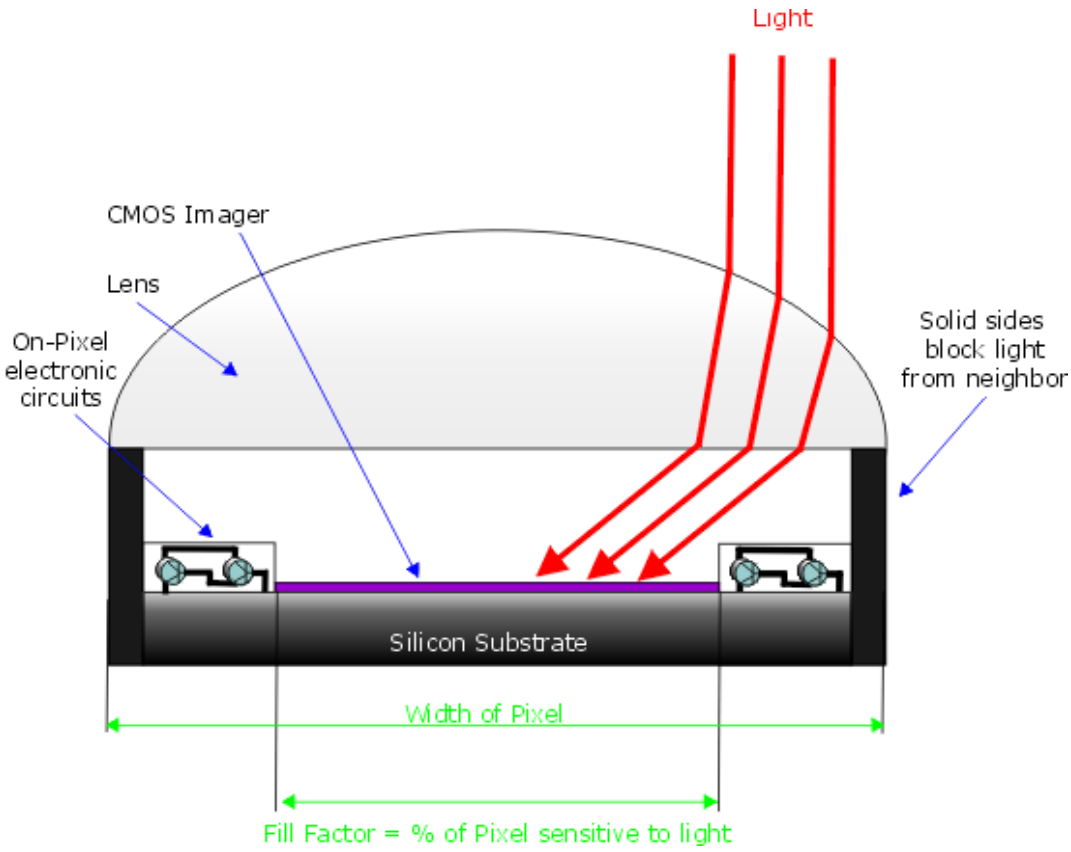
Pikseleistä jännitteiden haku A/D-muuntimeen tai -muuntimiin tapahtuu yleensä CMOS-kennossa rivi kerrallaan. Tätä tapaa kutsutaan myös nimellä “rolling shutter”. /88/, /89/, /158/

CMOS-kennot tehdään samalla periaatteella kuin tietokoneiden puolijohteet, kuten RAM-muistit ja prosessorit. Tässä on myös se etu, että niitä voidaan valmistaa samoilla tuotantolinjoilla, mikä huomattavasti laskee kehitys- ja tuotantokustannuksia. CMOS-pikseleiden rakenne on esitetty kuvassa 33. /88/, /89/, /158/



Kuva 31. CMOS-kenno ja kamera /88/

Koska jokainen pikseli itse suorittaa E/V-muunnoksen, voidaan tarvittaessa lukea vain yksittäisiä pikseleitä kennolta. Tällä apupiirillä on myös varjopuolensa, sillä osa fotoneista osuu näihin pikseleissä ja kenno tarvitseekin perinteisesti enemmän valoa verrattuna CCD-sensoriin. Tähän ratkaisuna on kehitetty pieniä linssejä jokaisen pikselin päälle ohjaamaan valo pois apupiiristä, mikä on esitetty kuvassa 32, sekä sijoittamalla apupiirit fotodiodien alle eri kerroksiin. /20/, /88/, /89/, /158/



Kuva 32. Mikrolinssin toimintaperiaate /67/

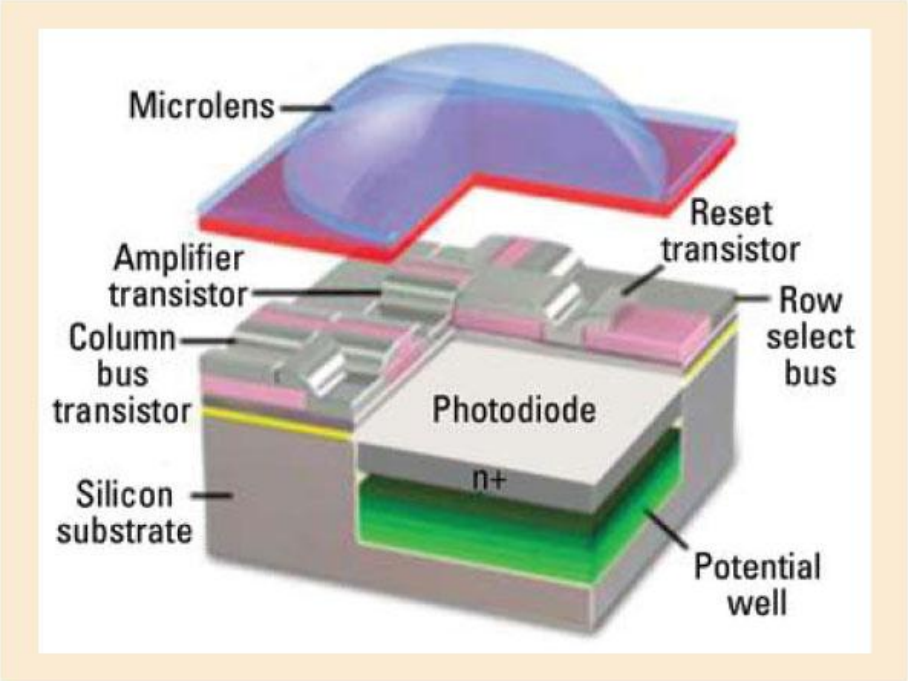


Figure 2. In the CMOS pixel structure, each pixel incorporates a photodiode (potential well) to collect light, a reset transistor, a row select (column bus) transistor and an amplifier transistor. The photodiode is covered by a microlens to help gather light, increasing sensitivity. The photodiode is typically 25 percent to 30 percent of a pixel's area. This is called the sensor's fill factor.

Kuva 33. Poikkileikkaus eräästä CMOS-kennon pikselistä /56/

Joskus tapaa 3T, 4T tai 5T liitteitä CMOS-kennoissa, missä T:n etuliite kertoo, kuinka monesta transistorista pikseli koostuu. 3T:ssä on siis kolme transistoria, mikä on pienin määrä, millä pikselin voi rakentaa. Transistorien vähäinen väärä aiheuttaa paljon hetkellistä lämmöstä johtuvaa häiriötä ja 4T:ssä hetkellistä häiriötä ei ole. Toisaalta mitä enemmän transistoreja, sitä enemmän yleensä tarvitaan valoa. /158/

3.2.3. CCD vs. CMOS

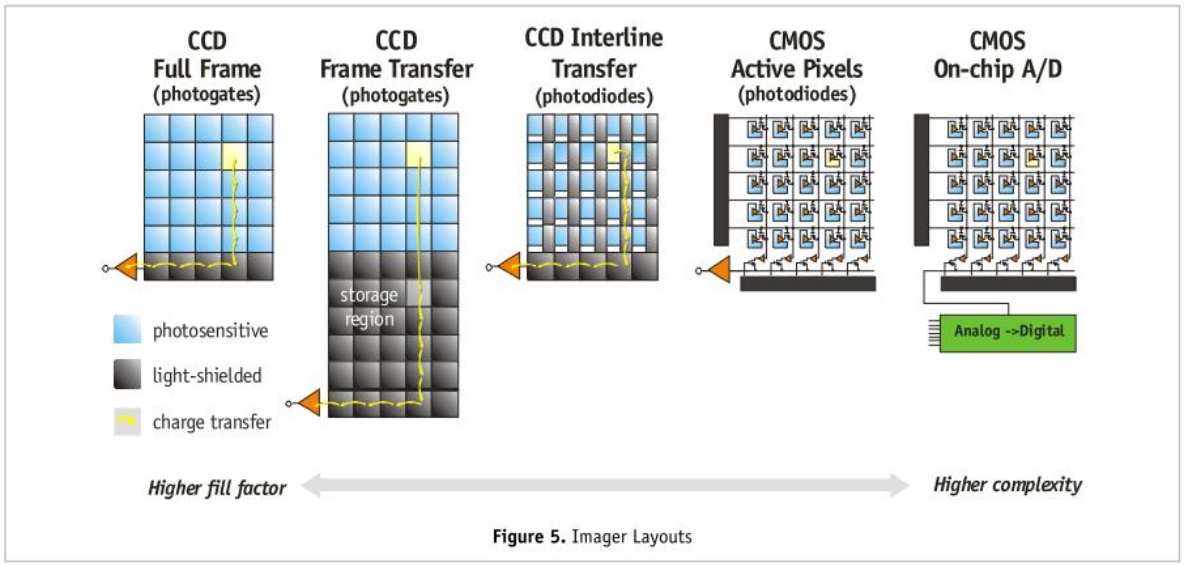
Tässä kappaleessa on lyhyesti esitelty suurimpia eroja CCD- ja CMOS-kennojen välillä. Molemmilla on omat vahvuutensa ja heikkoutensa tekniikan kehityksestä huolimatta. Taulukoissa 1 ja 2 on käydään vielä läpi CCD:n ja CMOS:n eroja ja kuvassa 34 kuvan informaation siirtotapoja CCD:llä ja CMOS:lla.

Taulukko 1. CCD vs. CMOS /157/

Feature	**CCD**	**CMOS**
Signal out of pixel	Electron packet	Voltage
Signal out of chip	Voltage (analog)	Bits (digital)
Signal out of camera	Bits (digital)	Bits (digital)
Fill factor	High	Moderate
Amplifier mismatch	N/A	Moderate
System Noise	Low	Moderate
System Complexity	High	Low
Sensor Complexity	Low	High
Camera components	Sensor + multiple support chips + lens	Sensor + lens possible, but additional support chips common
Relative R&D cost	Lower	Higher
Relative system cost	Depends on Application	Depends on Application
Performance	CCD	CMOS
Responsivity	Moderate	Slightly better
Dynamic Range	High	Moderate
Uniformity	High	Low to Moderate
Uniform Shuttering	Fast, common	Poor
Uniformity	High	Low to Moderate
Speed	Moderate to High	Higher
Windowing	Limited	Extensive
Antiblooming	High to none	High
Biasing and Clocking	Multiple, higher voltage	Single, low-voltage

Taulukko 2. CCD vs. CMOS /157/

Initial Prediction for CMOS	**Twist**	**Outcome**
Equivalence to CCD in imaging performance	Required much greater process adaptation and deeper submicron lithography than initially thought	High performance available in CMOS, but with higher development cost than CCD
On-chip circuit integration	Longer development cycles, increased cost, tradeoffs with noise, flexibility during operation	Greater integration in CMOS, but companion chips still required for both CMOS and CCD
Reduced power consumption	Steady improvement in CCDs	Advantage for CMOS, but margin diminished
Reduced imaging subsystem size	Optics, companion chips and packaging are often the dominant factors in imaging subsystem size	CCDs and CMOS comparable
Economies of scale from using mainstream logic and memory foundries	Extensive process development and optimization required	CMOS imagers use legacy production lines with highly adapted processes akin to CCD fabrication



Kuva 34. Erilaisia CCD- ja CMOS-siirtotekniikoita /158/

- CMOS-kamerat ovat immuuneja “blooming”- sekä “smear”-ilmiöille. Molemmat aiheuttavat CCD:n kuvaan valkoisia tai muun värisiä sarakkeita. /88/, /89/, /159/
- Rolling shutterilla toimiva CMOS kärsii “screw”-, “wobble”- sekä “partial exposure”-ilmiöistä. /88/, /89/, /159/
- Fotonit osuvat myös apupiiriin, minkä takia CMOS-sensori tarvitsee perinteisesti enemmän valoa verrattuna CCD-sensoriin. /88/, /89/, /159/
- CMOS-kennot voidaan tuottaa tehtaalla tavallisella puolijohdekomponenttilinjalla, mikä tekee niistä erittäin edullisia tuottaa. Samalla puolijohteiden sekä niiden tuotannon kehittyessä CMOS-kennot kehittyvät. /88/, /89/, /159/
- CMOS-kennot kuluttavat vähemmän virtaa kuvatessa, joidenkin lähteiden mukaan jopa satakertaisesti. Tämä ei kuitenkaan tarkoita, että itse kamera itsessään veisi vähemmän virtaa. /88/, /89/, /159/

3.2.4. Värit sekä värikuvien ottaminen

CCD ja CMOS ottavat vastaan fotoneita, joiden määrää mittaamalla saadaan aikaan pikselin arvo, **valoisuuden määrä**, mutta ne **eivät mittaa valon aallonpituutta**. Värien saamiseksi

kuvaan onkin kehitetty erilaisia tekniikoita. Tässä kappaleessa käydään läpi värikuvien ottamiseen ja käsittelyyn liittyviä asioita.

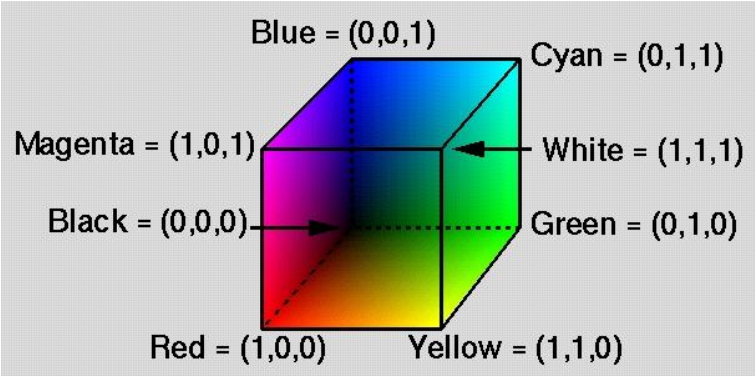
Yleistä väreistä

Suodattimissa ja väreissä tulee usein vastaan lyhenne RGB, mikä tarkoittaa punaista, vihreää ja sinistä väriä. Nämä tulevat ihmisen silmän soluista, jotka aistivat vain näitä kolmea pääväriä sekä harmaasävyjä. Useissa suodattimissa vihreää väriä onkin eniten, sillä ihmisen silmässä on eniten vihreän aallonpituuden havaitsevia soluja muihin nähden.

Päävärit

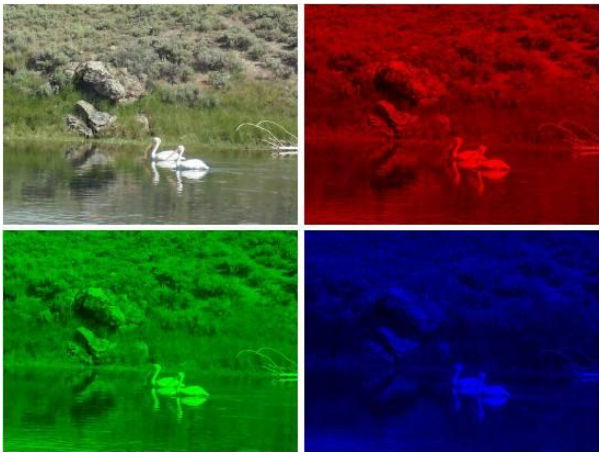
Pääväreillä tarkoitetaan värejä, joita ei saada yhdistämällä muita värejä, mutta niitä yhdistämällä saadaan aikaiseksi muut värit. Kuvassa 35 on kuvattu näiden välinen suhde värikuutiossa ja kuinka sävyt syntyvät lähestyessä kuution kulmia. Mikä tahansa väri tai sävy voidaan ilmoittaa värikuution avaruudessa.

Yleisesti taiteilijoiden suosimat päävärit ovat punainen, keltainen ja sininen, mitkä ovatkin enemmän tunnettuja pääväreinä RGB:n sijaan.



Kuva 35. Värikuutio /46/

Tietotekniikassa ja kuvissa pääväreillä (RGB) on usein 8-bittinen arvo, eli voimakkuus voidaan jakaa 256 eri tasoon, 0 - 255, ja sitä kutsutaan värikanavaksi. RGB:ssä on kolme väriä ja kolme värikanavaa. Näillä värillä saadaan 255 * 255 * 255 eri väriyhdistelmää, noin 16,7 miljoonaa eri väriä. Täysin mustassa kuvassa kaikkien kanavien arvo on 0 ja täysin valkeassa 255. Kuvassa 36 vasemmalla ylhäällä on kuva joutsenista. Kuva on jaettu kolmeen värikanavaan, jotka yhdistettynä saadaan ”alkuperäinen” kuva aikaiseksi.



Kuva 36. Kuva muodostuu kolmesta värikanavasta /12/

Värikuvasta saadaan harmaasävykuva laskemalla kaikkien kanavien yhteinen keskiarvo tai käyttämällä yhtä värikanavaa. Tämä kuitenkin riippuu täysin käytetystä interpoloinnista.

Konenäkökameroiden värit on usein ilmoitettu joko harmaasävyinä (monocolor) tai väreinä (color).

3.2.5. Värikuvien otto tapoja

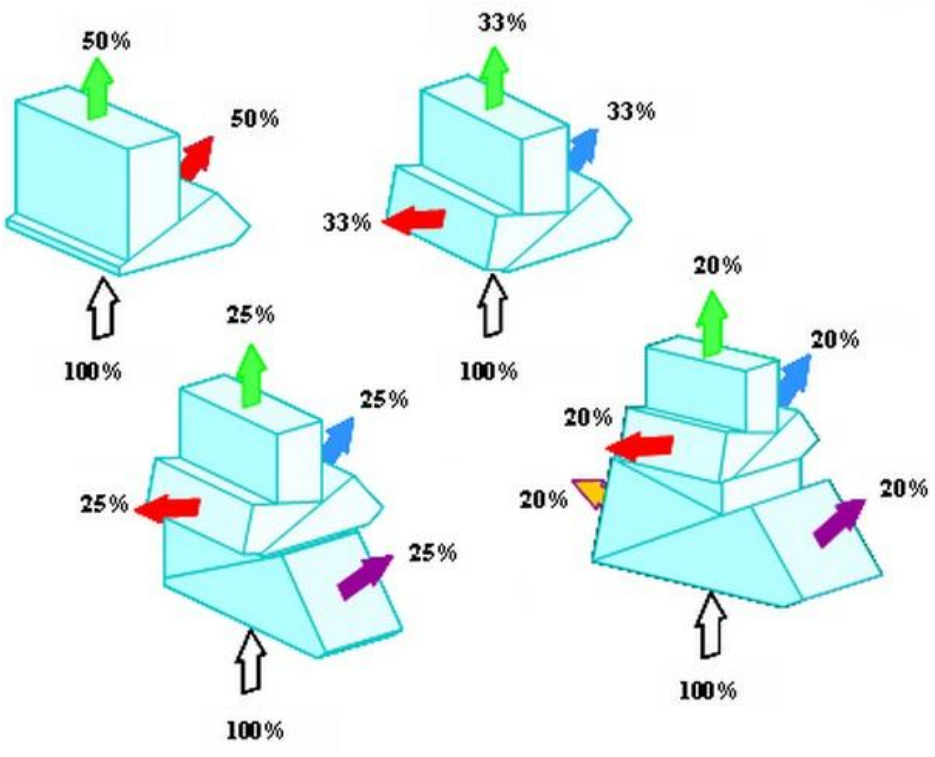
Eriväristen valojen käyttö valaistuksessa

Eräs tapa saada värit kuvaan on käyttää eri aallonpituuksia valaistuksessa. Kohde valaistaan ja kuvataan eri aallonpituuksilla, kuten punaisella, sinisellä ja vihreällä. Kohde heijastaa takaisin eri aallonpituuksia eritavalla, jolloin yhdistämällä nämä kaikki yhdeksi kuvaksi saadaan aikaan värikuva kolmesta eri harmaakuvasta. Laitteiston puolesta tämä voi olla edullisin

vaihtoehto, sillä hyvällä harmaasävykameralla saadaan tarkkoja värikuvia vain eri valoja käyttämällä. Huonona puolena tällaisessa järjestelmässä on sen hitaus, sillä kohde joudutaan kuvaamaan joka värillä ja tänä aikana kohteen tulisi olla liikkumatta. /27/

3CCD

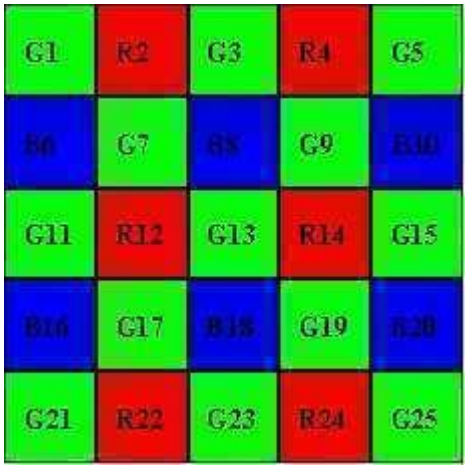
CCD:lle on avuksi sovelluksia, joissa tuleva valo jaetaan prismoilla esimerkiksi punaiseksi, siniseksi ja vihreäksi. Jokaiselle prisman jakamalle "värille" on omat kennonsa. Kohteesta tuleva valo voidaan jakaa kahdesta viiteen eri väriin, mutta silloin tarvitaan parempi valaistus, sillä sama valon määrä jaetaan näiden prismojen kesken. Yleisin tällainen järjestelmä tunnetaan nimellä 3CCD, jossa valo jaetaan kolmeen väriin: punaiseen, vihreään ja siniseen. Kuvassa 37 on esitetty 3CCD:n sekä kolmen muun prismatekniikalla toteutetun kameran toimintaa. /10/, /63/, /158/



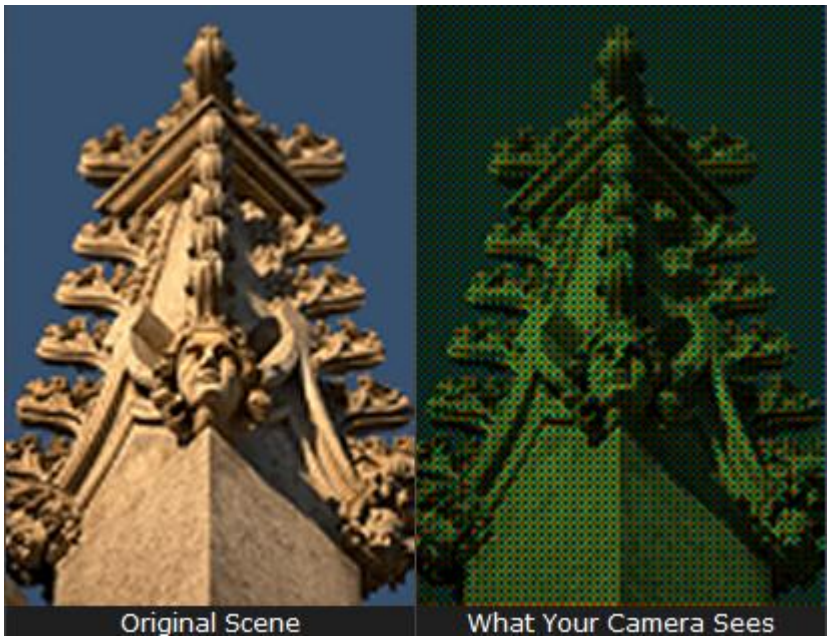
Kuva 37. Neljä erilaista prismatekniikkaa. Värit on merkattu nuolilla ja valonmäärä suhteessa tulevaan valoon on esitetty prosentteina. /10/

BAYER-SUODATIN

3CCD:n ongelmana on, että se on kallis ja vie prismojen ja kennojen takia paljon tilaa. Yhdellä kennolla värikuvia voidaan ottaa käyttämällä suodattimia (Color Filter Array, CFA), jolloin kukin pikseli vastaa tiettyä väriä. Tällaisista yleisin on Bayerin RGB-suodin, jossa värisuodattimet ovat mosaiikkimaisesti pikseleiden päällä, jolloin kukin pikseli vastaa määrättyä väriä. Kuvassa 38 on Bayer suodattimen perinteinen malli havainnollistettuna. /20/, /34/, /63/, /126/, /158/



Kuva 38. Bayer suotimen mosaiikkimainen RBG ruudukko /22/



Kuva 39. Vasemmalla on esitetty miten ihminen ja oikealla miten kamera näkevät saman patsaan /20/

Karkeasti ottaen pikselille pääsee vain yksi kolmasosa tulevasta valosta, 25 % sinisestä, 25 % punaisesta ja 50 % vihreästä. Koska suotimessa 50 % pikseleistä on vihreitä, tämän värin kanava on kaikista vähiten altis häiriöille. Kuvassa 39 on havainnollistettu, kuinka suodin suodattaa eri värejä kuvasta. Jokaisella pikselillä on vain oma väriarvonsa. 3-kanavainen värikuva saadaan tästä kuvasta interpoloimalla. /20/

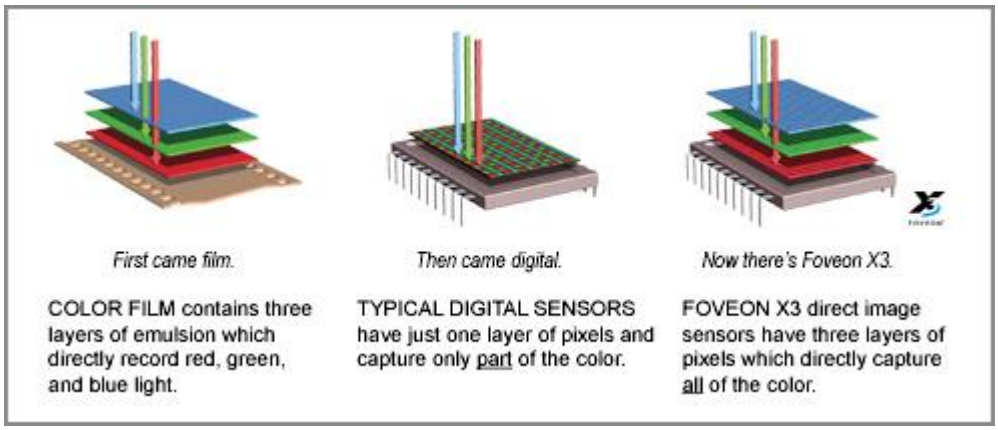
Bayerin RGB-suodin ei ole ainoa mosaiikkimainen värisuodin. Olemassa on suotimia kaikilla pääväreillä, sekä suotimia parantamaan hämärässä kuvausta, jossa on esimerkiksi omat suotimet saman värin “tummille” ja “vaaleille” sävyille tai suodin on poistettu joidenkin pikseleiden päältä (White pixels). /20/

FOVEON X3

FOVEON X3 on CMOS-kennotyyppi värikuvaukseen. Siinä on kolme eri kennoa päällekkäin, jokainen kerros kerää tiettyjä aallonpituuksia, sinisiä, vihreitä ja punaisia. Vihreät ja punaiset aallonpituudet läpäisevät sinisen kerroksen, punainen vihreän, joka lopulta törmää omaan

kennoonsa. Tämä on suhteellisen uusi ja vähemmän käytetty kennotyyppi digitaalisessa kuvauksessa. Periaate Foveonissa on sama kuin kamerafilmeissä. Foveon X3:sen suodatusta kameran filmiin sekä Bayerin suotimeen on verrattu kuvassa 40. /34/, /50/

Kameroiden tiedoissa kokonaispikselimäärä ei aina kerro kuvan todellista kokoa, sillä valmistajat saattavat lisätä pikselit yhteen, kun kuvan koko määräytyy yhden kennon pikseleiden mukaan. /34/, /50/

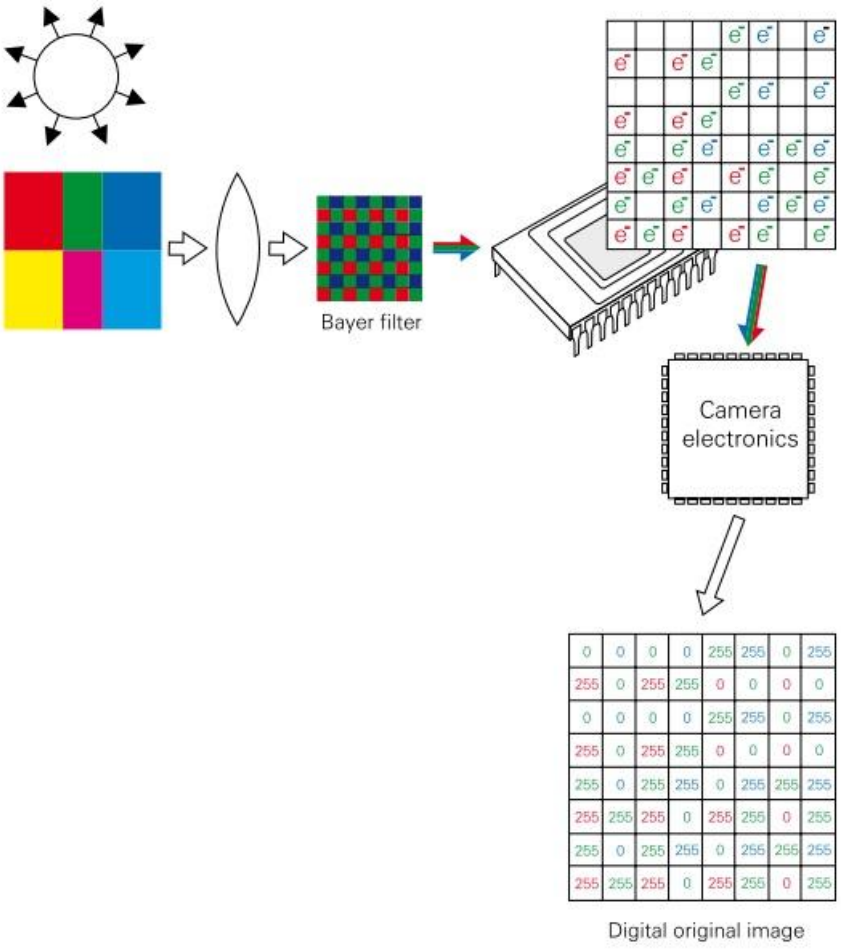


Kuva 40. Kuva esittää Foveon X3:n toiminnan verrattuna perinteiseen kameran filmiin ja Bayer-värisuodattimeen /50/

Värien laskenta ja interpolointi

”Interpolointi: Jonkin suureen tunnettujen tai taulukoitujen arvojen välissä olevien arvojen laskeminen.” /165/

Vaikka normaalissa värikuvassa on punainen, sininen ja vihreä väriarvo joka pikselille, todellisuudessa kameran kennolle suotimen läpi päätyy vain yksi, harmaasävyinen arvo, joka vastaa jotain kolmesta väristä. Tämä tulee ottaa huomioon värikuvien tarkkuutta tarkastellessa. Kappaleessa on asia esitetty suuntaa antavasti. /21/, /22/, /63/

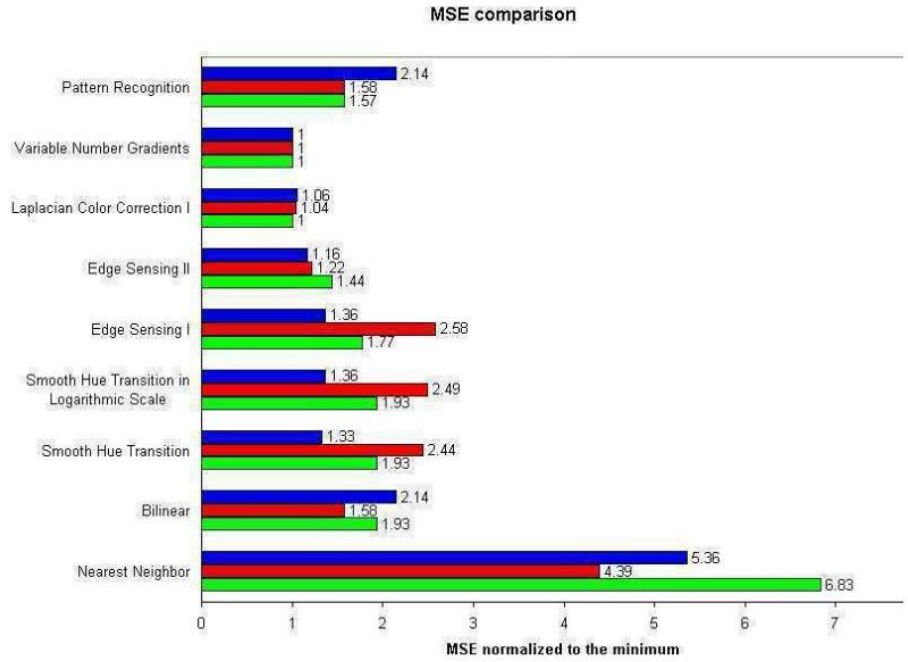


Kuva 41. Värikuva kennolla varauksina ja kameralta lähtevä kuva /63/

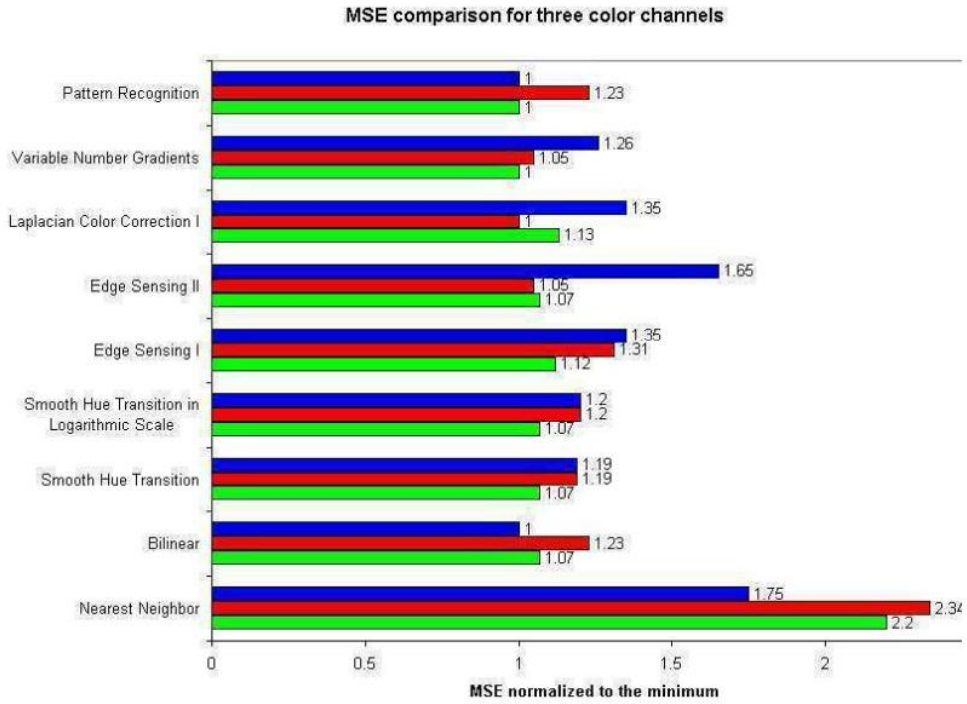
Kuvassa 41 on esitetty alkuperäinen värikuva sekä raakakuva kameralta. Raakakuvassa pikselillä on vain yksi arvo, riippuen siinä käytetystä suotimesta. Kahden muun värin arvot puuttuvat ja nämä lasketaan viereisistä pikseleistä. Samanlainen tilanne on kuvassa 39. Tätä tapaa kutsutaan CFA (Color Filter Array) interpoloinniksi tai “demosaicing”. Tapa muistuttaa etäisesti miinaharava-peliä. /21/

Kuvissa 43, 44 ja 45 on vertailtu eri algoritmien tarkkuutta kuvaan 42 sekä algoritmien laskentaan kulutettua aikaa verrattuna “billinear”-algoritmiin. Tarkoitus on esittää, että mitä todenmukaisempi kuva on, sitä suurempi on laskentaan käytetty aika. Itse algoritmeihin tässä ei oteta kantaa. /22/

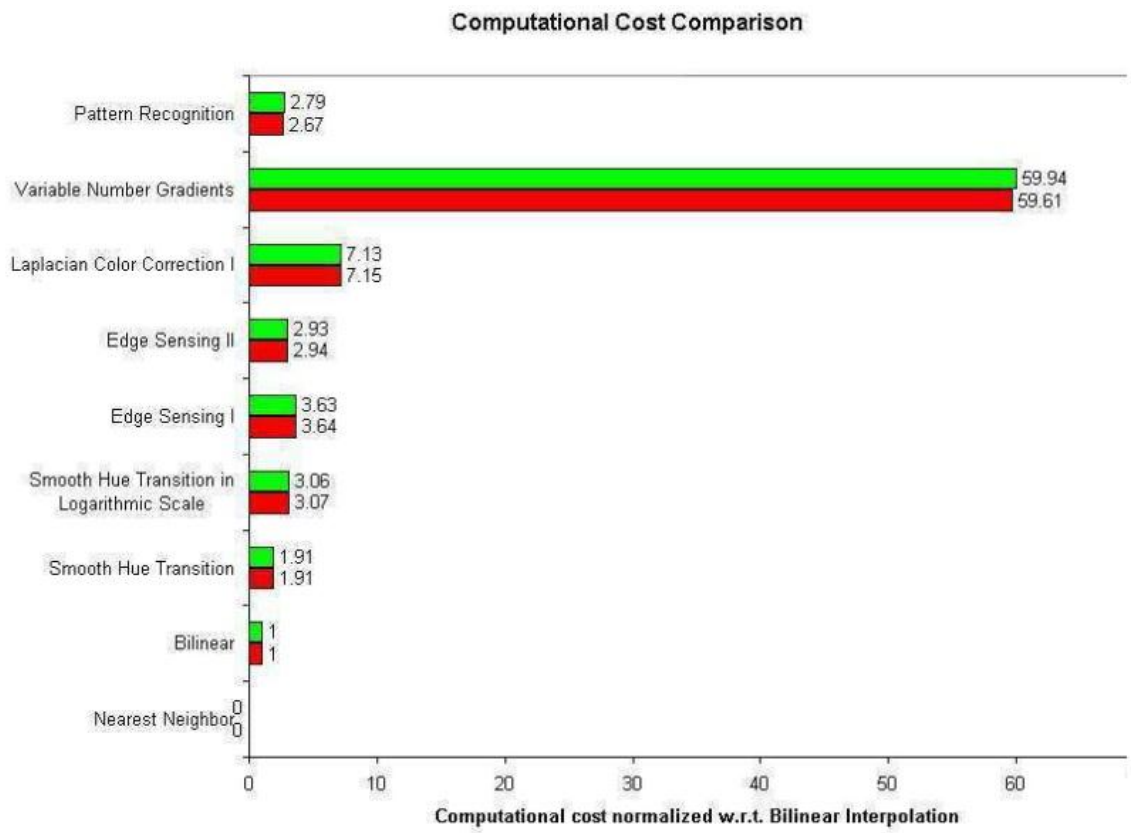
Kuva 42. Alkuperäinen kuva /22/



Kuva 43. /22/



Kuva 44. /22/



Kuva 45. /22/

MSE, “Mean Square Error”, keskineliövirhe eli virheen neliön keskiarvo. Vertailussa on laskettu alkuperäisen ja interpoloidun kuvan jokaisen värikanavan pikseleiden arvojen ero, jotka on korotettu potenssiin ja jaettu pikseleiden kokonaismäärällä.

3.3. Optiikka

Jotta kuva saataisiin muodostettua kennolle, tarvitaan avuksi optiikkaa. Linssien ja aukkojen avulla kennolle saadaan tarkka kuva kohteesta. Oikeanlaisella optiikalla saadaan rajattua oikea kohde kuvattavaksi. Tällöin voimme käyttää pienempiresoluutioisia kameroita ja täten nopeuttaa operaatiota. Aluksi onkin hyvä käydä läpi tärkeitä termejä ja periaatteita asian ymmärtämiseksi.

Polttopiste

Polttopiste on piste, jonka kautta kaikki linssin läpi kulkeva valo kulkee. Linssin sanotaan kokoavan valonsäteet, jotka kohtaavat polttopisteessä.

Polttoväli, focal length

Polttovälillä mitataan kennon ja objektiivin optisen keskipisteen, polttopisteen välimatkasta. Objektiivissa polttovälillä määrätään kuvan suurennos, kennolle syntyvän kuvakulman. Polttoväli ilmoitetaan objektiiveissa millimetreinä. /82/, /102/, /127/

Mitä lyhyempi polttoväli on, sitä laajemman kuvan kameralla voi ottaa ja sitä pyöreämmältä kuva vaikuttaa, eli reunoille muodostuu vääristymiä. /82/, /102/

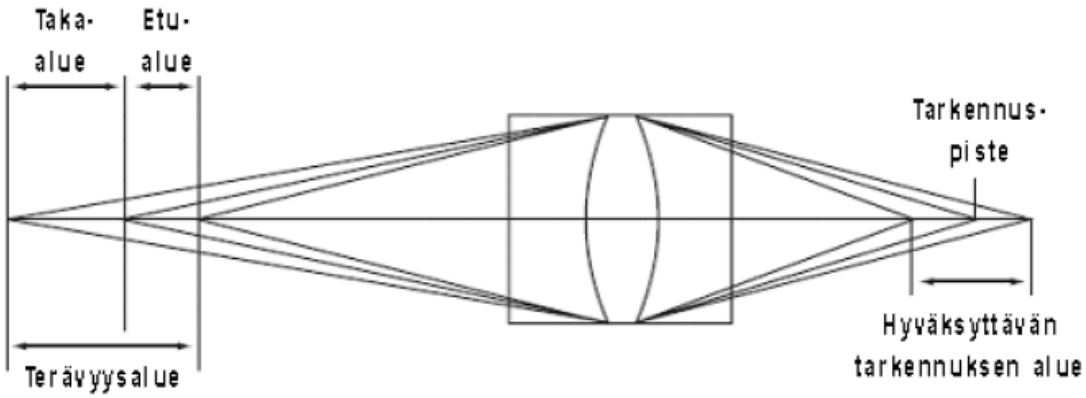
40 - 55 mm polttoväli vastaa suurin piirtein ihmisen silmän näkemää perspektiiviä. /82/, /102/

Terävyysalue, depth of field (DOF)

Kameran kuva tarkennetaan teräväksi ja tarkaksi. Terävyys on määriteltävä alue, joka jakaantuu terävimmän pisteen molemmille puolille. Pisteestä mentäessä jatkuvasti kauemmas, kuva alkaa sumentua ja kun reunoja ei enää erota selvästi, ollaan terävyysalueen ulkopuolella. Tätä on kuvattu kuvassa 46. Terävyysalue on siis hyvin vapaasti määriteltävissä tarkoituksesta tai tarkastelijasta riippuen. /29/, /80/, /126/, /128/

Terävyysalue pienenee, kun: /80/, /126/

- etäisyys kohteeseen pienenee
- aukon koko kasvaa
- polttoväli kasvaa
- kennon koko kasvaa.



Kuva 46. Terävyysalue /29/

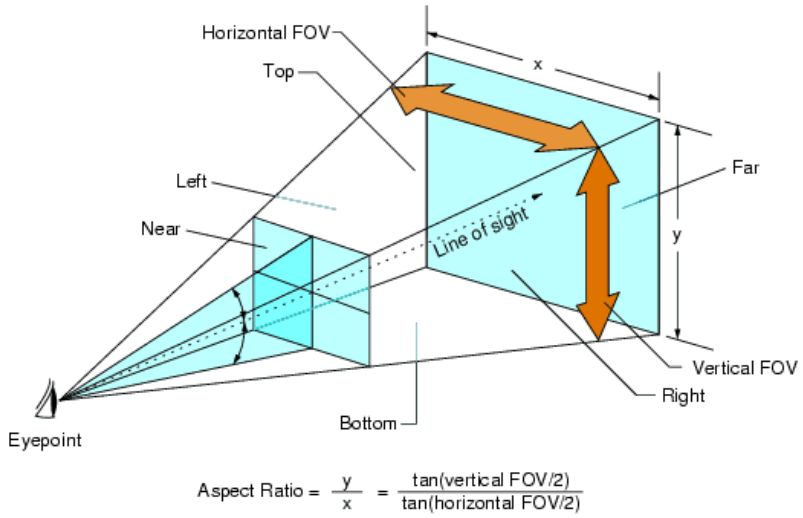
Näistä yllä mainituista syistä johtuen esimerkiksi kännykkäkameroissa, joissa on pieni kamera ja kenno, terävyysalue on suuri ja vaikutusmahdollisuudet niihin pienet.

Objektiivin kuvakulma, field of view (FOV):

Objektiivin kuvakulmaa ei pidä sekoittaa suomenkielessä kulmaan, josta kuva otetaan. Objektiivin kuvakulmalla tarkoitetaan polttopisteestä katsottuna objektiivista lähtevää kulmaa, josta kuva muodostuu. Kuvaa kuinka suuri osa edessä olevasta kohteesta tulee kuvaan ja se ilmoitetaan asteina. Mitä lyhyempi polttoväli, sitä suurempi on kuvakulma. Kuvakulman vaikutusta esitetään kuvassa 47.

Kuvakulma voidaan antaa pysty- sekä vaakasuoraan.

Kuvakulman avulla voidaan laskea esimerkiksi kuinka kauas kamera on sijoitettava, jotta kuva vastaa todellisuudessa tiettyjä mittoja, kuten korkeutta. /102/



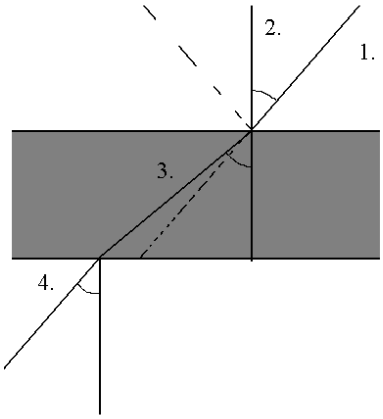
Kuva 47. Kuvakulman vaikutus /47/

3.3.1. Linssi ja kuva

Valon siirtyessä väliaineesta toiseen se taittuu näiden rajapinnassa. Jokaisen aineparin rajapinnalle on oma taitesuhde, joka voidaan laskea Snellin lailla. Tätä lakia ei tässä käydä läpi, vaan tärkeintä on ymmärtää, että valon edetessä aineesta toiseen, kuten vaikka ilmasta vesilasiin, valonsäteen etenemiskulma muuttuu tulokulmaan nähden. Asia on esitetty kuvassa 48, missä:

1. tuleva valon säde
2. rajapintojen normaali
3. taittunut valon säde väliaineessa
4. valo taittuu jälleen rajapinnassa.

"Normaali on vektori, suora tai jana, joka on kohtisuorassa toista pintaa, janaa tai suoraa vasten." /194/

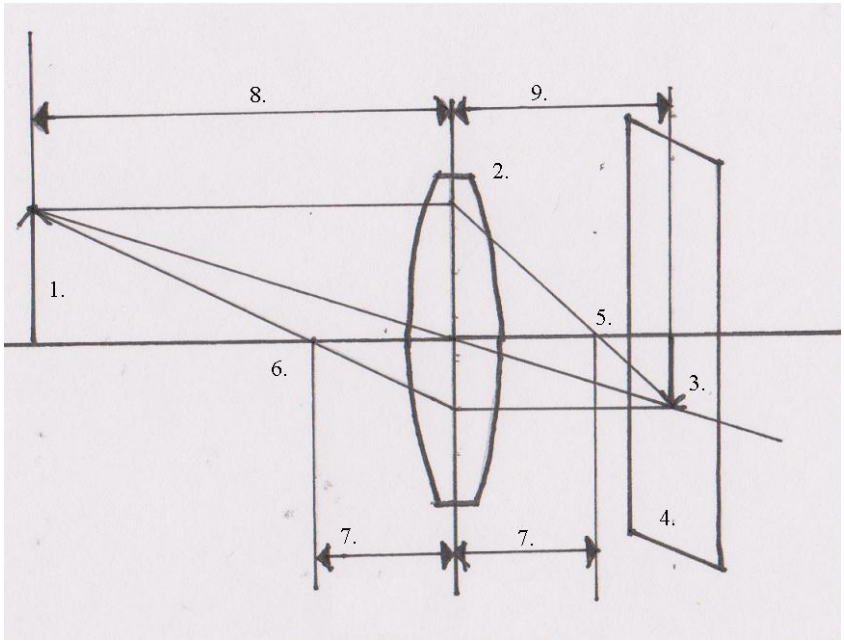


Kuva 48. Valon kulku kahdessa eri aineessa on erilainen.

Valo saattaa myös heijastua takaisin samassa tulokulmassa pintojen normaaliin nähden. Tätä ilmiötä kutsutaan kokonaisheijastukseksi ja se tapahtuu, kun valo tulee optisesti tiheämmästä (suurempi taitekerroin) aineesta harvempaan laskettavan kriittisen taitekulman jälkeen.

Kupera ja kovera linssi

Linssien muodot voidaan jakaa karkeasti kahteen muotoon, valoa kerääviin kuperiin sekä valoa hajottaviin koveriin linsseihin. Näitä kahta yhdistämällä saadaan aikaan linssistöjä, objektiiveja. Kuvassa 49 on esitetty kuperan linssin toiminta, mistä nousee kameran optiikan kannalta tärkeitä asioita. /78/, /91/



Kuva 49. Kuvassa on esitetty yksinkertainen kupera linssi

Kuvassa 49:

1. kohde
2. kupera, identtisesti kaksipuolinen linssi
3. väärinpäin heijastuva kuva kohteesta
4. kameran kenno
5. taaempi polttopiste
6. etupuolen polttopiste
7. polttoväli
8. välimatka kohteen ja linssin välillä
9. välimatka linssin ja kennon välillä.

Kupera linssi kokoaa siihen tulevat valonsäteet, jotka leikkaavat polttopisteessä. Vaikka kameran objektiivien linssistöt eivät näytä yhtään samalta kuin kuvassa 49, periaate on silti sama ja niihin pätee ohuen linssin yhtälö: /78/

$$1 / l_{li} + 1 / l_{ke} = 1 / f \quad (1)$$

missä

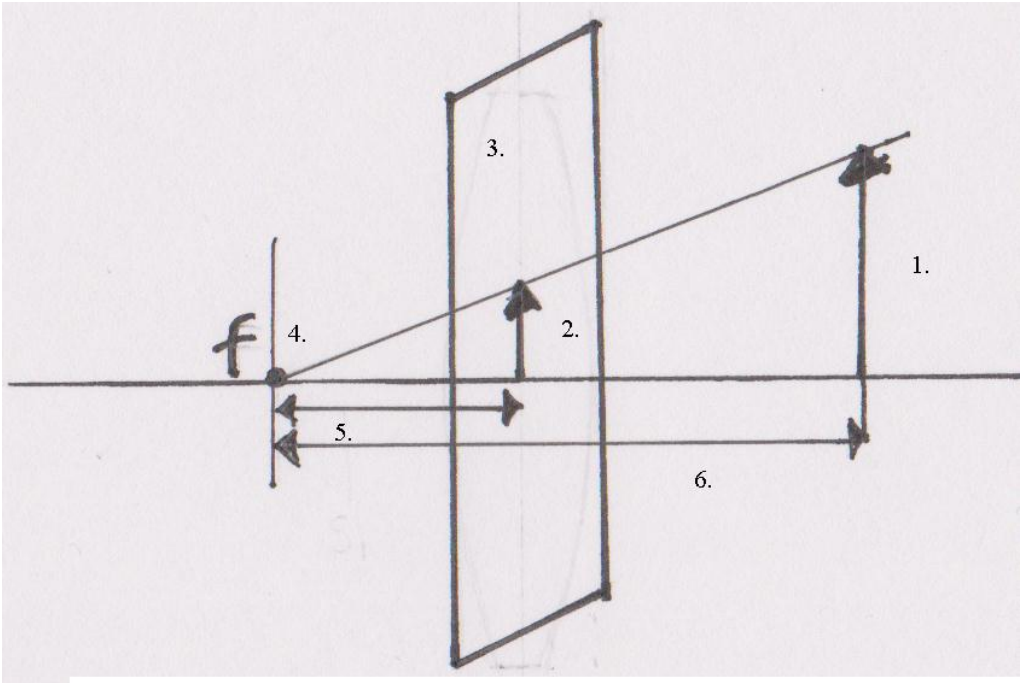
l_{li} on välimatka kohteesta linssin keskikohtaan

l_{ke} on välimatka linssin keskikohdasta kameran kennolle

f on polttoväli

Tietämällä kaksi etäisyyttä voidaan laskea kolmas etäisyys. Esimerkiksi kun olemme valinneet kameran ja tiedämme etäisyyden kuvattavaan kohteeseen, voidaan laskea polttoväli ja täten tiedämme millainen objektiivi tulee hankkia, tai vastaavasti kun kamera ja objektiivit ovat jo olemassa, voimme laskea, kuinka kauas kamera tulee sijoittaa. /78/

Koska välimatka linssistä kohteeseen on huomattavasti suurempi kuin matka kennolle (8.>>9.), voimme olettaa, että kenno on terävyysalueella. Kuva 49 voidaan esittää kenties selvemmin neulanreikäkameran kuvan muodostumisen avulla. /78/



Kuva 50. Neulanreikäkamera

Kuvassa 50:

1. kuvattava kohde
2. kuva kennolla
3. kenno
4. polttopiste F
5. polttoväli
6. välimatka kohteen ja linssin välillä.

Kuvasta 50 voidaan laskea pituuksia kaavalla: /78/

$$f * h_t / l_{ka} = h_i \quad (2)$$

missä
f on polttoväli
h_t on kohteen oikea korkeus
l_{ka} on kohteen etäisyys kamerasta
h_i on kohteen korkeus kuvassa

Kaavalla 2 voidaan laskea muodostuvan kuvan korkeus kuvassa, tarvittava objektiivi, jos etäisyydet ja korkeudet tiedetään tai etäisyyksiä. Esimerkiksi määrittäessä kohteen mittaamiseen tarvittavia pikselimääriä, tästä mallista on paljon enemmän hyötyä kuin linssiversiosta.

Objektiivi

Suomen kielessä objektiivi on linssi tai usean linssin muodostama järjestelmä, joka muodostaa optisen kuvan. Kameroihin liitettäviä objektiiveja kutsutaan kuvausobjektiiveiksi. Englannin

kielessä usein puhutaan myös linsseistä (camera lens) tai kameroiden optiikoista (camera optics). Yleensä yksi linssi ei riitä antamaan hyvää ja tarkkaa kuvaa kohteesta. /182/

Objektiivien yhteydessä törmää usein monenlaisiin kirjainlyhenteisiin, kuten Di, Digitally integrated. Näitä lyhenteitä ei ole standardoitu, vaan ne ovat usein valmistajakohtaisia ja kertovat valmistajan ilmoittamista ominaisuuksista. /182/

Objektiivien laatu riippuu usein linssien laadusta, mikä taas näkyy hinnassa. Se saattaakin olla kamerajärjestelmän kallein osa.

Aukko

Kennon saamaa valomäärää voidaan säätää aukolla. Mitä suurempi aukko, sen enemmän valoa pääsee kennolle. Aukon kokoa kuvataan aukkosuhteella, f-luvulla. Suurella aukolla f-luku on pienempi, eli mitä pienempi f-luku, sen enemmän valoa pääsee kennolle. Valomäärän kaksinkertaistamiseen aukon pinta-alan on kasvettava 1,4-kertaiseksi. /126/, /184/

Aukkosuhde lasketaan jakamalla polttoväli aukon halkaisijalla. Tästä syystä aukon koko voi olla ilmoitettu esimerkiksi f/2,8 tai f2,8, mitkä tarkoittavat samaa asiaa. /126/, /184/

Aukon koko vaikuttaa myös terävyysalueeseen. Suurentamalla aukkoa terävyysalue pienenee. /126/, /184/

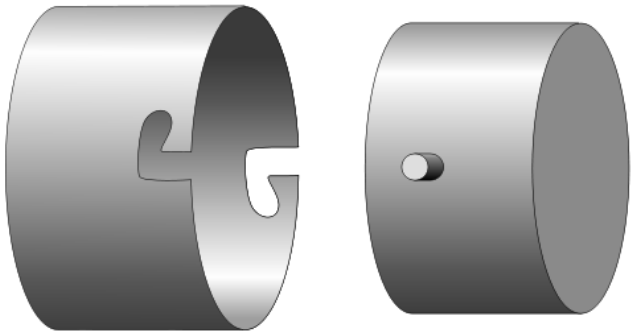
Kinovastaavuus ja rajauskerroin

Polttovälit eivät ole suoraan vertailukelpoisia, sillä kennojen koko vaihtelee. Tästä syystä se suhteutetaan 35 mm filmin tai ns. kokokennon (full-frame) kokoon. Käyttäessä kinovastaavuuksia voidaan siis verrata eri kameroiden polttovälejä keskenään. Tämä vertaaminen voidaan tehdä käyttämällä ns. rajauskerrointa (Crop Factor), jolla kertomalla saadaan vastaavuus kinokokoon. /82/

Kiinnitysmekanismit

Mekanismeja on kolmen laisia: ruuvimainen (screw-threaded), klipsi-lukko (friction lock) sekä bajonetti (bayonet). Bajonetti on yleisin tapa, koska sillä saadaan parhaiten objektiivien sähköistämiseen. Bajonetin yksinkertaistettu rakenne on esitetty kuvassa 51. Englannin kielessä mekanismien nimille ei löydy yhtä yhteistä sanaa, vaan ne vaihtelevat mount, system sekä jo mainittujen mekanismien kanssa. /126/

Kiinnitysmekanismeista on tehty standardeja, joita on mm. C- ja T-mount.



Kuva 51. Bajonetin tai Bayonetin periaate /193/

Objektiivien tyyppejä

Objektiivien, joiden avulla kuva näyttää “luonnolliselta”, polttoväli on yleensä 45 - 55 mm, kutsutaan normaaliobjektiiveiksi. Kennon läpimittaa suurempia polttovälejä (suhteutettuna 35 mm filmiin) kutsutaan **teleobjektiiveiksi**, joissa on yli 55 mm polttoväli ja pienempiä **laajakuvaobjektiiveja**, joissa 10 - 45 mm polttoväli. Lisäksi on olemassa mm. **makro-objektiiveja**, joilla suurennetaan pieniä kohteita tarkasti, kun kuva otetaan läheltä, sekä ns. **kalansilmäobjektiivi**, jonka kuva-ala on 180 astetta. Laajan kuva-alan vuoksi kuva on hyvin vääristynyt. /29/, /126/, /189/

Makro-objektiiveille on yleensä ilmoitettu suurennus suhde, esimerkiksi 1:1. /127/, /189/, /191/, /192/

Makro, makrokuvaus ja objektiivit

Makrokuvauksessa pyritään kuvaamaan kohde suurempana kuin se on luonnostaan. Digitaalisten järjestelmien myötä makrokuvauksen käsite on muuttunut ja sillä voidaankin tarkoittaa kaikkea lähikuvaamista, johtuen kennojen koosta. Makro-objektiivien avulla voidaan kuvata kohdetta läheltä suurimman suurennoksen aikaansaamiseksi. /29/, /82/, /126/

Sisätarkenteinen objektiivi

Kuvan tarkennus tapahtuu objektiivin sisällä olevia linssejä siirtämällä, eikä pidentämällä objektiivia tai linssejä pyörittämällä. /126/

Valovoima

Objektiiveissa ilmoitettu valovoima on samalla suurin aukkoluku. Se kuvaa objektiivin kykyä kerätä valoa ja se on täysin riippuvainen objektiivin linsseistä, niiden laadusta ja erityisesti koosta.

Zoom

Zoom-termillä tarkoitetaan polttovälin muuttamista, jolloin kuva voidaan tarkentaa eri etäisyyksille. Digikameroissa törmää kahteen eri termiin, optinen zoom (optical zoom) sekä digitaalinen zoom (digital zoom). /29/, /55/, /82/

Digitaalinen zoom ei oikeastaan ole mikään tarkennus, vaan kuvan rajaus. Siinä rajataan kuvasta pienempi alue ja mahdollisesti interpoloidaan suuremmaksi, jolloin ei tapahdu polttovälin muuntumista. Kun pikseleitä on todella paljon, esimerkiksi 4000 x 4000, voidaan kuvasta ottaa VGA-tasoinen kuva (640 x 480) tarkasteltavaksi, kuin se olisi otettu VGA-tasoisella kameralla. /55/

Esimerkkejä objektiiveista

AF 70–200/2.8–4.0 1:1 Ø72 mm

Kyseessä Auto Focus zoom-objektiivi, joka muuttaa polttoväliä 70 - 200 mm välillä, ja jonka aukkoluku vaihtelee f/2.8:n ja 4.0:n välillä polttovälin mukaan. Sillä saavutetaan 1:1 suurennossuhde, mikä kertoo, että se on makro-objektiivi, jossa on 72 mm läpimittainen suodinkierre. /127/, /189/

Canon EF-S 15-85mm f/3.5-5.6 IS USM

Canonin zoom-objektiivi, jonka polttoväli voidaan muuttaa 15 - 85 mm väliltä. Samalla aukkoluku vaihtelee 3.5 - 5.6 väliltä. EF (Electro-Focus) on Canonin oma objektiivien kiinnitysjärjestelmä, IS (image stabilization) tarkoittaa kuvanvakaajaa ja USM on lyhenne ultraäänimoottorista, jolla liikutetaan zoom-objektiivin linssejä. /126/, /127/, /189/

3.3.2. Kameratermistöä

Tässä kappaleessa käydään läpi erilaisia termejä ja asioita, joihin törmää kameroista luettaessa ja ominaisuuksia tutkiessa.

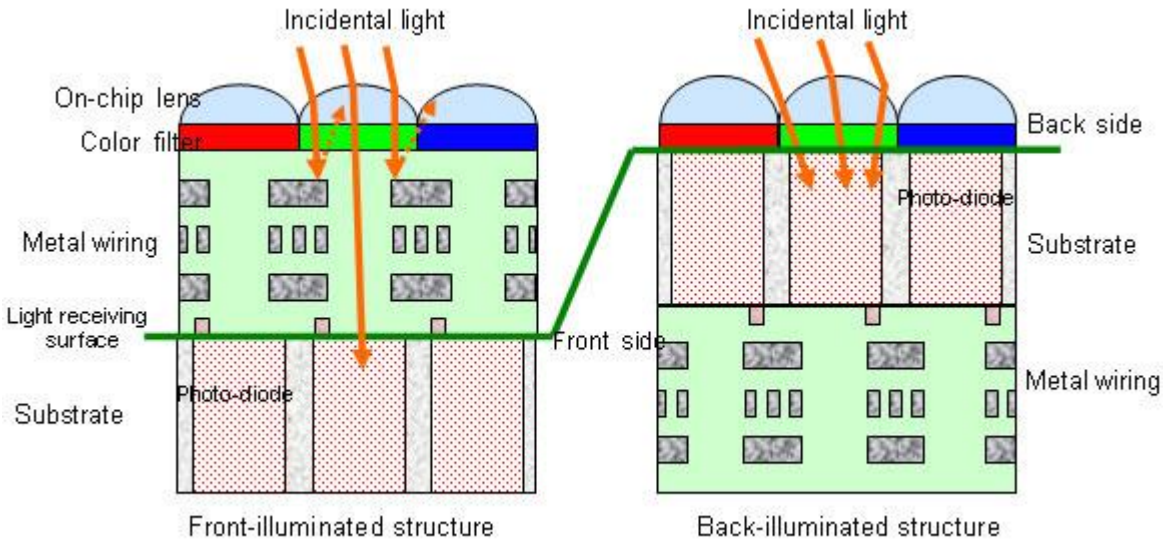
A/D-muunnin

ADC, Analog-to-Digital Converter, on elektroninen piiri, joka muuntaa analogisen signaalin, eli jatkuvan, portaattoman signaalin, digitaaliseen, mitattavaan ja porrastettavaan muotoon. A/D-muuntimen tarkkuus eli resoluutio ilmoitetaan bitteinä. Esimerkiksi 8-bittisellä muuntimella mitattava arvo voidaan jakaa (2^8-1) 255 osaan 0 - 255 välille. /126/

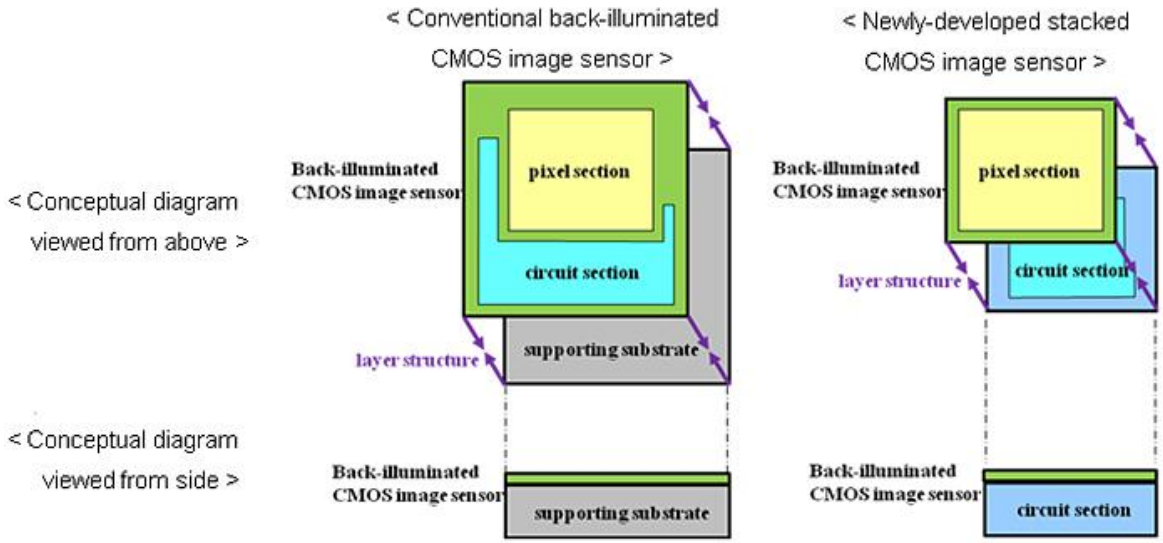
ADC:n laatu ja nopeus korostuvat eritoten CMOS-kennoilla, joissa maksimi kuvanottotaajuus määräytyy käytännössä ADC:n nopeuden mukaan. /126/

Back-illuminated sensor

Tunnetaan myös nimellä “Backside illumination sensor”. CMOS-kennotyyppejä, joissa ongelma transistoreiden tiellä oloon pikselissä on ratkaistu siirtämällä ne fotodiodien alle eri kerroksiin. Näin saavutetaan parempi valoherkkyys, suurempi valon tulokulma sekä pienennetään kohinaa. Kuvassa 52 on poikkileikkaus perinteisestä pikselistä sekä ”Backside” pikselistä, josta nähdään kuinka huomattavasti suurempi osa valosta osuu fotodiodiin. Kuvassa 53 on vielä pidemmälle viety CMOS-kenno, missä kennoon integroidut lisäpiirit on siirretty pikseleiden alle kokonaan, jolloin koko pinta-ala saadaan fotodiodeiden käyttöön. /152/



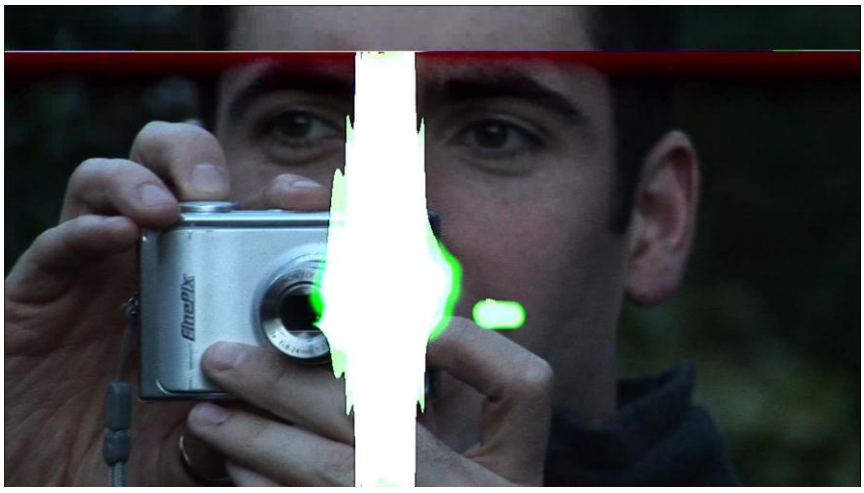
Kuva 52. Vasemmalla perinteinen CMOS-sensorin ratkaisu, oikealla back-illuminated /150/



Kuva 53. Oikealla back-illuminated kenno, jossa apupiirit on siirretty pikseleiden alle /151/

Blooming ja Smear (CCD)

Molemmat tarkoittavat pikseleiden ylivalotusta. Bloomingissa CCD:n pikseli ylikuormittuu kuvattaessa ja ylimääräiset elektronit vuotavat pikseli pikseliltä samalla sarakkeella. Smearissa luettaessa pikseleitä tapahtuu samanlainen ilmiö. Kuvassa 54 on äärimmäinen esimerkki bloomingin aiheuttamasta häiriöstä kuvassa. /30/



Kuva 54. Blooming ilmiö /36/

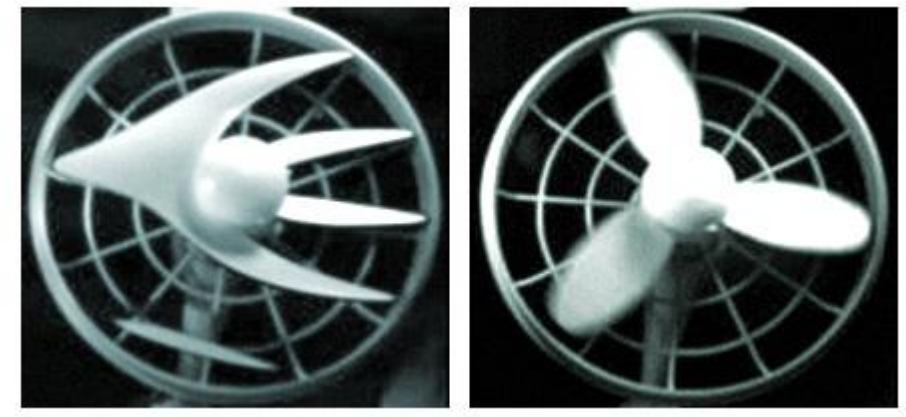
Dark current - pimeä virta

Pimeä virta aiheutuu kamerassa syntyvän lämmön vaikutuksesta. Atomien lämpöliike aiheuttaa sähköisiä muutoksia pikselin arvoissa, mikä taas aiheuttaa kohinaa.

Global shutter vs. Rolling shutter

Global shutter on tekniikka, jossa koko kenno luetaan kerralla, jolloin liikkuvia kohteita kuvattaessa ei synny vääristymiä. CMOS-kennoilla, varsinkin edullisimmilla, on yleensä käytössä ns. rolling shutter. Kuvatessa rolling shutterilla kennon ensimmäinen rivi nollataan, jonka jälkeen se luetaan. Tämä sama operaatio toistetaan, kunnes viimeinenkin rivi on luettu. Tästä johtuen ensimmäisen ja viimeisen rivin välillä on viivettä, joka pahimmillaan aiheuttaa pahoja virheitä varsinkin kuvattaessa liikkuvaa kohdetta. Näitä virheitä tai häiriöitä kutsutaan termeillä “skew”, “wobble” sekä “partial exposure”. /13/

CMOS-kennoja on saatavilla myös global shutterilla. Globalin ja rollingin suurin ero on havainnollistettu kuvassa 55.



Kuva 55. Vasemmalla rolling ja oikealla global shutter /13/

Skew, wobble sekä partial exposure

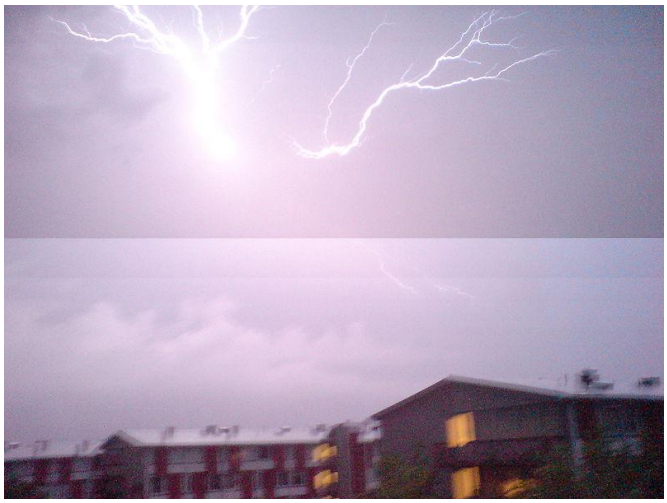
“Skew”, “wobble” sekä “partial exposure” ovat kaikki rolling shutterin aiheuttamia virheitä kuvassa. Skew syntyy kun kohde tai kamera liikkuu rivien luvun aikana. Tämä aiheuttaa kohteen venymisen tai taipumisen kuvassa. Esimerkiksi kuvattaessa kohdetta “|”, sen liikkuessa oikeasta laidasta vasempaan valotuksen aikana, kuvasta muistuttaa “/”. Tätä on havainnollistettu kuvassa 56. Häiriötä voidaan käyttää myös erikoistehosteena, kuten kuvassa 58 on tehty. /37/

Kuva 56. Skew ilmiö /187/

Wobble syntyy samasta syystä kuin skew. Wobble syntyy yleensä kameran värähdellessä tai tärähtäessä ja aiheuttaa kuvaan aaltomaisen häiriön. Yhdessä skew’n kanssa kuvasta tulee erittäin vääristynyt eikä kohteesta saa välttämättä selvää ollenkaan. Ongelma tulee eteen varsinkin videota kuvatessa. /37/

Partial exposure eli osittainen valottuminen ilmenee voimakkaiden valojen, kuten salamavalon kanssa. Ongelma syntyy, kun kesken kuvan valotuksen taso muuttuu. Tällöin osa kuvasta on

valoisa ja loput selvästi pimeämpiä. Kuvassa 57 on otettu kuva salaman välähdyksessä. Kuva on osittain /37/



Kuva 57. Partial exposure. Kuvan luku on edennyt puoleenväliin kuvaa, kun salama on jo kadonnut /188/

Kuva 58. Häiriöitä voidaan myös käyttää korostuskeinoina. Kuvassa kitaran kielet /166/

Nämä kolme häiriötä ovat aina läsnä kuvatessa rolling shutteria käyttävällä kameralla. Näitä voidaan vähentää parantamalla kennon A/D-muunninta, nostamalla näyteenottotaajuutta tai vaikka lisäämällä mekaaninen suljin. /37/

ISO-herkkyys

Kuvaa kennon herkkyyttä valolle. ISO-luku on kahden asteikon yhdistelmä: eurooppalaisen DIN:n (Deutsches Institut für Normung) ja amerikkalaisen ASA:n (American Standard Association). ASA:n kaksinkertaistuminen tarkoittaa herkkyyden kaksinkertaistumista ja DIN:ssa kaksinkertaistuminen on ilmoitettu kolmella asteella. DIN 21 astetta tarkoittaa samaa kuin ASA 100. Yleensä ISO-luvussa on ilmoitettu vain ASA:n arvo tilan säästämiseksi. /126/

Kun herkkyyttä nostetaan, tarvitaan vähemmän valotusaikaa, mistä taas syntyy kohinaa, pimeää virtaa ja häiriöitä kuvaan. Pidempää valotusaikaa käytetään kuvatessa hämärässä tai korostaessa liikettä. /183/

Interline Transfer

Interline Transfer on CCD-kennotekniikka, jossa pikseleille on tehty valolta suojatut siirtorekisterit, jonne pikseleiden valon tuottama varaus siirretään. Siirtorekisteristä varaukset siirretään eteenpäin kuten normaalissa CCD:ssä. Rekisterit ovat joko aina pikselisarakkeen vieressä omana sarakkeenaan tai matriisimaisena pikseleiden alla. /126/

Kontrasti

Kontrastilla voidaan tarkoittaa mustan ja valkean erottumista toisistaan, mutta myös kohteen erottumista taustasta. Mitä suurempi kontrasti, sitä selkeämmin kohde ja tausta ovat erillään. /126/

Lomitettu (interlaced scanning) sekä lomittamaton (progressive scan)

Kameroissa yleensä törmää termiin ”interlaced” tai ”progressive scanning”. Nämä ovat eri videoiden esitystekniikkaa. Interlaced on historiallinen jäänne ajoilta, jolloin käytettiin lähinnä kuvaputkitelevisioita ja kaistanleveydet olivat pieniä. Interlaced tarkoittaa, että videon kuva on jaettu parillisiin ja parittomiin riveihin. Lomitettu 25 fps:n video sisältää oikeasti 50 kuvaa virkistystaajuuden nostamiseksi. Lomitetut kuvat vievät vähemmän kaistaa, mutta liikkuvissa kuvissa syntyy sumeita reunoja. Esimerkiksi LCD-televisioihin tuodessa kuva voidaan joutua interpoloimaan. /126/, /185/, /186/, /205/

Jotkin CCD-videokamerat kuvaavat vieläkin lomitettua kuvaa. Näissä kennon pikselit on jaettu valmiiksi parillisiin ja parittomiin riveihin.

Lomittamattomassa (progressive) videossa esitetään koko kuva kerralla. Tällöin ei tapahdu liikkuvissa kuvissa minkäänlaista sumentumista. Se on suositumpi formaatti tietotekniikassa, sillä lomitettu kuva on hankalampi pakata ja esittää esimerkiksi LCD-näytöissä. Lomittamattomasta kuvasta saa tarvittaessa helposti lomitetun ilman häviötä. /185/, /186/, /205/

Karkeasti otettuna interlaced tarkoittaa analogista, progressive digitaalista.

Videoissa saattaa olla esimerkiksi 720p tai 1080i, missä ”p” tarkoittaa progressive scan ja ”i” interlaced scan. /185/, /186/, /205/

Still-kuva

Still-termiä käytetään erottamaan kuva digikuvauksessa videokuvasta sekä valokuvakameraa videokamerasta.

Suljinaika, valotusaika

Suuressa osassa kameroista käytetään joko elektronista tai mekaanista suljinta. Mekaanisilla sulkijoilla suljinajalla tarkoitetaan sulkimen aukioloaikaa, jolloin valo pääsee kennolle. Kuvaus tapahtuu, jonka jälkeen suljin estää valon pääsemisen kennolle. CCD- ja CMOS-kennoissa on jo itsessään elektroninen suljin, eli kenno lopettaa valottumisen itsestään luvun yhteydessä. CCD:ssä saattaa tapahtua vuotoa, mikäli voimakas valonlähde jatkaa kennon valottamista luvun yhteydessä. /126/, /129/

Todella pitkiä valotusaikoja käytetään mm. tähtitieteessä sekä pimeäkuvauksessa ja lyhyitä valotusaikoja kuvatessa liikkuvia kohteita. Mitä kirkkaampi valo ja suurempi ISO-herkkyys, sitä lyhyempi valotusaika. /126/, /129/

3.3.3. Tietoliikenneyhteydet

Konenäköjärjestelmissä käytetään nykyisin tietoliikenneyhteyksissä USB 2.0-, FireWire-, Camera Link- tai GigE-protokollia. Varsinkin tiedonsiirtonopeus nousee tärkeäksi mikäli kuva pitää siirtää jonnekkin muualle kamerasta. Tässä kappaleessa käydään läpi näiden ominaisuuksia sekä tiedonsiirtonopeuksia. /118/

USB 2.0

Universal Serial Bus, sarjaliikennearkkitehtuuri tiedonsiirtoon. Käytetään tiedonsiirron lisäksi laitteiden ohjaukseen.

USB:ssä on käytössä +5 voltin käyttöjännite ja 100 mA:n virta. Standardin mukaan USB:ssä pystytään maksimissaan antamaan 500 mA, mutta joissain laitteissa, kuten kannettavissa tietokoneissa virran määrä on rajoitettu 100 mA. /118/, /153/, /204/

Teoreettinen maksiminopeus USB versio 2.0:ssa on 480 megabittiä sekunnissa, eli 60 Mt/s ja minimissään 24 Mt/s. Todellisuudessa maksiminopeus on noin puolet teoreettisesta maksiminopeudesta. Tämä tarkoittaa, että jos kuvien koko on esimerkiksi 2 Mt, teoreettisesti voidaan saada 30 fps, mutta käytännössä fps on 12 - 24 kuvan väliltä. /118/, /153/, /204/

Samaan laitteeseen voidaan yhdistää 127 USB-laitetta. Kaapelin maksimipituus on viisi metriä. Keskittymiä käyttämällä matka voidaan nostaa korkeintaan 25 metriin asti. /118/, /153/, /204/

USB-rajapinta löytyy jokaisesta tietokoneesta. Siinä on hyvät standardit laitteille, mutta ei ajureille. Tietoliikenne USB:n kautta kuormittaa suoritinta. /118/, /153/, /204/

FireWire

FireWire eli toiselta nimeltään IEEE 1394a tai b on alunperin suunniteltu audiovisuaalisen datan siirtoon, missä se on parempi kuin USB. FireWire on luotettava ja siinä on mm. ennakoiva tiedonsiirron kesto, minkä ominaisuudet korostuvat ajallisesti tärkeissä tehtävissä. Nykyään FireWireä käytetään lähinnä teollisuudessa tai Applen valmistamissa tuotteissa. /118/, /204/

Standardi jaetaan 1394a:han ja 1394b:han. A:ssa teoreettinen maksimi nopeus 50 Mt/s, vähintään 37,5 Mt/s ja maksimissaan 32 Mt/s per laite. B:ssä teoreettinen maksimi on 100 Mt/s, minimissään 75 Mt/s ja maksimissaan 64 Mt/s per laite. Kaikki kytketyt laitteet jakavat saman rajapinnan. Tämän takia 400 Mb/s on harhaanjohtava. /118/, /204/

FireWiressä maksimi laitemäärä on 63. Kaapelin enimmäispituus a:ssa on 4,5 metriä, mutta onnistuneita siirtoja voidaan tehdä jopa kahdeksan metrisillä kaapeleilla ja b:ssä sata metriä. Kaapelin kokonaispituutta ei voida kasvattaa kuten USB:llä. Pystyy siirtämään enemmän virtaa laitteelle ja kuormittaa vähemmän suoritinta kuin USB 2.0. /118/, /204/, /118/

Camera Link

Camera Link on tietokonenäköä varten kehitetty standardoitu sarjaliikenneprotokolla. Etuna muihin tiedonsiirtoihin on se, että tiedonsiirrossa ei ole viivettä, se on turvallinen, varma ja hyvin vakiinnuttanut asemansa teollisessa kuvansiirrossa. Jokaisen standardia käyttävän laitteen on toimittava muiden standardia käyttävien laitteiden kanssa. /7/, /118/, /204/

Camera Linkin liittimiä, kaapeleita tai kuvankaappauskortteja ei käytetä muualla kuin kuvankäsittelylaitteissa. Jokainen standardia käyttävä laite tulee sertifioida valmistajan toimesta, jolloin hinnatkin ovat korkeammat. /7/, /118/

Standardissa on määritelty myös PoCL, ”Power over Camera Link”, jossa dataliikenteen lisäksi päätelaite saa virtansa saman kaapelin kautta. /7/, /118/

Nopeus kasvaa käytettävien 8-tavuisten kanavien mukaan kolmella kanavalla 255 Mt/s, kuudella 510 Mt/s, kahdeksalla 680 Mt/s sekä kymmenellä 850 Mt/s. Nopeutensa takia Camera Linkillä voidaan kuvata ja siirtää todella nopeasti, jopa useita satoja kuvia sekunnissa. Kaapelin maksimipituus 85 MHz:n taajuudella on kymmenen metriä, pienemmillä taajuuksilla kaapelin maksimipituus kasvaa. /7/, /118/

GigE Vision

GigE Vision on 2006 julkaistu standardi kameroiden rajapinnasta, jossa käytetään 1000 Mbit/s Ethernet-lähiverkkoa. Takana on ryhmä yrityksiä, jotka tahtoivat kehittää avoimen Ethernet-lähiverkkoon perustuvan standardin korkealaatuisille kameroille. /7/, /118/, /106/, /204/

GigE Vision jakaantuu neljään pääosaan: /7/, /52/, /106/, /204/

- GigE Vision Control Protocol GVCP: Ohjausprotokolla, joka toimii Universal Datagram Protocol (UDP) IPv4:lla. Määrittelee kuinka laitteita ohjataan ja säätää asetuksia, määrittelee streamauksen kanavat ja antaa kameroille mekanismin kuinka lähettää kuvaa ja ohjaustietoa isäntäkoneelle.

- GigE Vision Stream Protocol GVSP: Määrittelee käytetyt datatyypit ja kuinka kuvat siirtyvät verkon yli.
- GigE Device Discovery Mechanism: Määrittelee kuinka kamerat ja muut laitteet saavat IP-osoitteen.
- XML-pohjainen kuvaustiedosto, datalehti, jonka avulla pääsee käsiksi kameran ohjaukseen ja kuviin. Perustuu GenICam ohjelmisto rajapintaan, jolla voi ohjata kaikkia kameroiden rajapintoja tekniikasta riippuen.

Ethernet-protokollan käyttö tuo mukanaan monia hyviä lisäominaisuuksia konenäköjärjestelmiin: /7/, /204/

- Kaapelien pituus voi olla maksimissaan 100 metriä. Toistimilla, kytkimillä, keskittimillä, ja kuituoptiikalla saadaan matka kasvamaan tarvittaessa moninkertaiseksi.
- Nopea tiedonsiirto, 1 000 tai 10 000 Mb/s.
- Standardoidut, yleisessä käytössä olevat kaapelit sekä liittimet alentavat kustannuksia.
- Tiedon lähettäminen tavallista lähiverkkoa tai internettiä käyttäen.
- “Power over Ethernet” -tekniikalla voidaan samalla kaapelilla syöttää kameraan käyttöjännite.

Standardin kehitystä ja hallintaa valvoo tällä hetkellä The Automated Imaging Association AIA. /7/, /44/

3.3.4. Kameratyyppejä

Compact-kamera

Kompaktikamera, digipokkari, on kamera, jossa kaikki on kiinteää ja osia ei voida vaihtaa. Se on myös kuluttajille tarkoitetuista kameroista edullisin. Siinä on kiinteä optiikka, mahdollisesti mallista riippuen digitaalinen tai optinen zoom, sisäinen ja ulkoinen muisti, ja yleensä USB-liitin kuvien siirtoa varten. Fyysiseltä kooltaan pieni kokoinen, kuten on myös kennon koko.

Tästä johtuen valotusajat ovat pitkät, terävyysalue on hyvin iso ja sen pienentäminen on usein melko mahdotonta. Koska optiikkaa on mahdotonta vaihtaa, kuvan laatua parannetaan kasvattamalla pikseleiden määrää, mitkä nykykameroissa ovatkin 12 megapikselin luokkaa. /55/, /81/

Kompaktista kamerasta on tehty mahdollisimman automaattinen ja ominaisuuksien, kuten valotusajan muuttaminen voi olla mahdotonta. Kameroissa on myös LCD-näyttö, joka näyttää tarkasti saman kuvan kuin otettavista kuvista. Kuvien lisäksi kameralla voi ottaa videokuvaa ja ääntä. /81/

Kompaktikameroiden suosio johtuu niiden edullisuudesta, helppokäyttöisyydestä ja pienestä koosta. Kilpailu tuo myös jatkuvasti parempia kameroita edullisemmin uusilla ominaisuuksilla, kuten kosketusnäytöllä tai HD-tekniikalla. /81/

Järjestelmäkamerat

Järjestelmäkameroiden englanninkieliset nimet SLR, ”Single Lens Reflex” ja DSLR, ”Digital SLR” tulevat etsimissä käytetystä tekniikasta. Etsimestä käyttäjä näkee saman kuvan kuin otettava kuva. Toisin kuin LCD-näytössä, näkyvässä kuvassa ei ole viivettä. Kuvassa 59 on esitetty etsin. /81/

Kuva 59. Etsin /45/

Järjestelmäkameran ideana on, että kamera itsessään on runko, johon liitetään objektiivit, valot ja muut lisävarusteet ja niitä voi vaihtaa tilanteen mukaan, luoden näin järjestelmän. Tämä on suurin ero kompakteihin, missä kaikki on integroitua. Kameran ominaisuuksia voidaan helposti muuttaa ja säätää, kuten valotusaikaa. Siinä on suurempi kenno kuin kompaktikamerassa, jolloin aukon kokokin on suurempi. Tällöin voidaan vaikuttaa syväterävyyteen eritasolla mitä kompakteissa, valotusajat ovat huomattavasti pienempiä ja kuvia voi ottaa monta sarjassa kuvien laatua heikentämättä. /81/, /125/

Vaikka kennojen koko on fyysisesti isompi, ei järjestelmäkameroissa ole sen enempää pikseleitä kuin parhaimmissa kompakteissakaan. Kuvan paremmuuteen voidaan vaikuttaa muilla keinoilla, tilanteesta riippuen. /81/, /125/

Markkinoilla on myös järjestelmäkameroita ilman etsintä, jolloin kameran koko pienenee merkittävästi. /81/, /125/

Järjestelmäkamerat ovatkin huomattavasti kalliimpia kuin kompaktikamerat, mutta verrattuna konenäössä käytettäviin kameroihin, kuten älykameroihin, ne ovat edullisia. Suurimman yksittäisen menoerän tuovat erikseen ostettavat objektiivit. /81/, /125/

Web-kamerat

Web-kamerat ovat kameroita, jotka on suunniteltu nimenomaan pikaviestintäohjelmiin tai videoneuvotteluun. Ensimmäiset web-kamerat kehitettiin vuonna 1991 Cambridgen yliopistossa. Web-kameroita käytetään videopuheluissa, videoneuvotteluissa, valvontakameroina ja on hyvin suosittu tähtiharrastajien keskuudessa. Kamera on suunniteltu lähinnä videokuvan ottoon. Perineisin web-kamera on esitetty kuvassa 60, jollaista käytettiin myös työssä. /173/, /176/

Kameran rakenne on yleensä linssi, kenno, mikrofoni, elektroniikka ja yhteys tietokoneeseen. Linssi voi olla kiinteä, manuaalisesti säädettävä tai kuten uusimmissa malleissa automaattinen, kennoina CCD tai CMOS. Web-kamerat ovat edullisia, mistä johtuen kennojen ominaisuudet

eivät ole kovin hyvät ja tarvitsevat yleensä paljon valoa toimiakseen. Useissa kameroissa on myös IR-suodin. Suotimen voi useimmista kameroista poistaa helposti, jolloin saadaan IR-taajuudet käyttöön. /173/, /176/

Kuva 60. Logitechin QuickCam- sarja on vanhimpia ja suosituimpia web-kamera merkkejä /174/

Yleensä kameroiden resoluutio rajoittuu yleensä VGA-tasoiseksi, sillä videopuhelusovellukset kuten Skype tai Microsoftin Messenger eivät tue korkeampaa resoluutiota. Olemassa on kuitenkin HD-tasoisia web-kameroita, joita käytetään mm. YouTube-videoiden kuvaamiseen. /173/, /176/

Suurimmalla osalla kameroista yhteys tietokoneeseen tapahtuu USB:llä. Saatavilla on myös FireWirellä kameroita, mutta nämä ovat kalliimpia vaikkakin ominaisuuksiltaan identtisiä USB- versioiden kanssa. Kameroita valvontatarkoituksiin on saatavilla myös verkkoliittimillä. /173/, /176/

Web-kameroiden suosio johtuu niiden helppokäyttöisyydestä ja edullisuudesta. /173/, /176/

Konenäkökamera

Konenäkökameroiksi (Machine vision camera) kutsutaan kameroita, joita käytetään konenäkösovelluksissa teollisuudessa. Kuvan parantamiseksi kameraan voidaan liittää objektiiveja ja suodattimia. Kuvassa 61 on Foculuksen konenäkökamera, johon on liitetty optiikkaa. Samanlaista kameraa käytettiin työssä. /83/

Konenäkökamerat eivät prosessoi kuvaa, vaan tähän tarvitaan jokin ulkoinen laite tai järjestelmä prosessoimaan kameran tuottamaa kuvaa sekä ohjaamaan mahdollisia I/O-laitteita. Kameroita on saatavilla USB, FireWire, Camera Link:llä sekä GigE Vision yhteyksillä. /83/

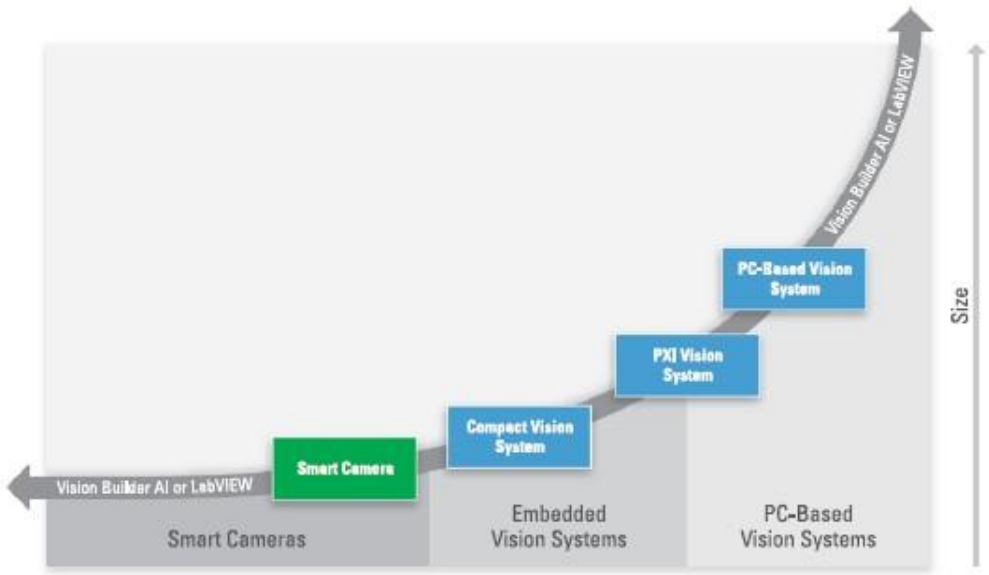
Usein konenäkökameran matkassa tulee kuvankäsittely kirjasto avuksi sovellusten kehittämiseen. /83/

Kuva 61. Foculuksen konenäkökamera, johon on kiinnitetty optiikkaa /5/

Älykamerat

Älykamerat (“smart” tai “intelligent”) ovat konenäköön suunniteltuja kameroita. Kameroihin on sisäänrakennettuna sulautettu järjestelmä tai FPGA kuvan prosessointia varten, sekä valmiit viestintäyhteydet, joilla voidaan suoraan ohjata esimerkiksi robottia asennuslinjalla ilman erillisiä ohjainlaitteita, kuten tietokonetta. Ulkonäöllisesti ne eivät juuri eroa perinteisistä konenäkökameroista. /83/, /171/

Älykameroihin voidaan liittää objektiiveja, valoja, I/O-laitteita, sekä muita lisälaitteita parantamaan kuvauksen laatua ja korostamaan haluttuja ominaisuuksia. Joissain malleissa on sisäänrakennettuna sulautettu Windows ja kameraan voidaan liittää hiiri, näppäimistö ja näyttö ja käyttää kameraa PC:n tavoin. /171/



Kuva 62. Älykameroiden koon suhde muihin järjestelmiin /120/

Pienessä koossa on myös huonot puolensa. Vaikka kameroiden tekniikassa, laskentatehossa ja muistissa on menty eteenpäin, jää se silti jälkeen PC:n suorituskyvystä sekä mukautuvuudestaan. Älykameroissa tapahtuvat kuvankäsittelyt usein jäävätkin muutamiin algoritmeihin. Kuvassa 62 on järjestetty konenäköjärjestelmiä pienimmistä suurimpaan. /171/

Älykamera tarjoaa usein useita valmiita toimintoja sekä kirjastoja kuvankäsittelyyn, mutta ohjelmoitavuutensa ansioista käyttäjä itse voi ohjelmoida laitteen haluamallaan tavalla ja näin saada optimoidun tuloksen. /171/

Älykamerat käyttävät konenäköön luotuja tiedonsiirtostandardeja (Camera Link sekä GigE Vision), mutta myös USB:tä sekä FireWireä. /171/

Älykamerat ovat todella kalliita verrattuna järjestelmäkameraan. Niiden etuna on pieni koko, korkea laatu sekä erikoistuneet ominaisuudet. /171/

3.4. Valaistus

Valaistus on eräs tärkeimmistä työkaluista konenäössä. Sillä voidaan korostaa haluttuja ominaisuuksia, poistaa kuvattavan kohteen materiaaleista ja muodoista syntyviä heijasteita sekä varjoja ja minimoida ympäristön tai liikkeen vaikutus kuvattavaan kohteeseen. /83/

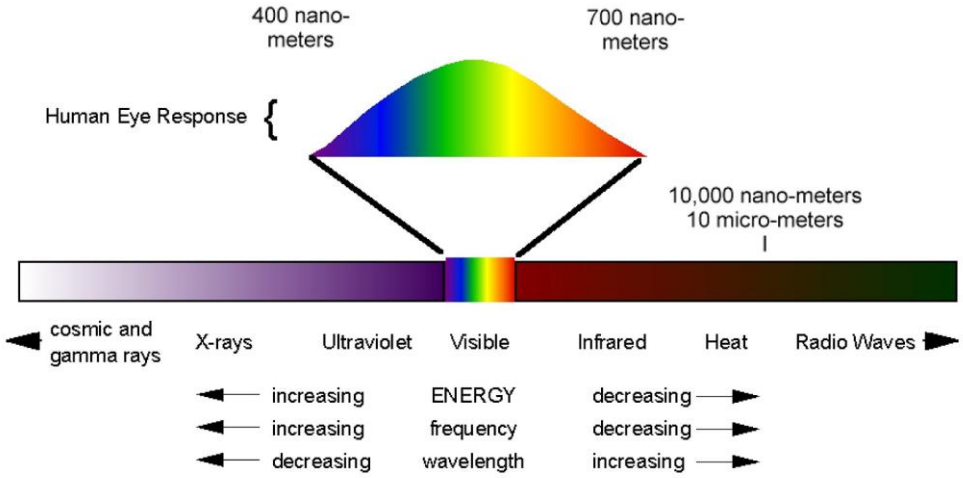
Valaistuksessa on hyvä huomioida mm:

- valon käyttäytyminen kappaleen pinnassa
- valonlähteen sijainti kameraan nähden
- käytettävä valonlähde
- valon suutaus
- valonlähteen aallonpituudet
- valon polarisaatio
- korostettavat kohteet.

Mitä valo on

Ihmisen silmien havaitsemia valon aallonpituuksia kutsutaan väreiksi. Valoksi voidaan kutsua myös juuri havaitsemisen ulkopuolelle jääviä aallonpituuksia: lyhyempänä ultravioletti sekä pidempänä infrapuna. Infrapunan ihminen aistii myös lämpönä. /195/

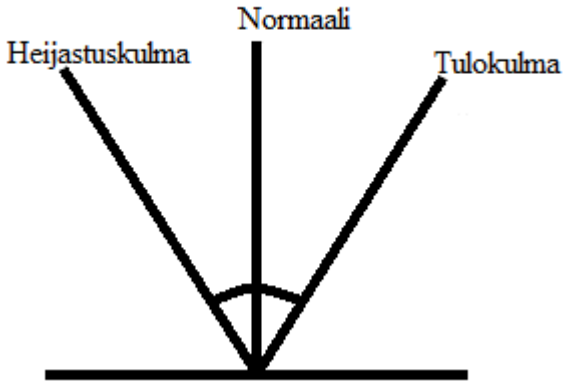
Niin sanottu väritön tai valkea valo sisältää kaikki havaitsemamme aallonpituudet, kaikki näkemämme värit. Valon osuessa kohteeseen osa valon aallonpituuksista imeytyy tai suodattuu kappaleeseen, absorboituu, ja kaikki muut aallonpituudet heijastuvat kohteesta. Nämä heijastavat aallonpituudet tuottavat kohteelle värin, jonka ihminen havaitsee. Mistä aallonpituudesta värit alkavat on aika hankala määrittää, sillä jokainen ihminen havaitsee värejä erilailla. Yleisesti violetti on 380 - 450 nm ja punainen 630 - 760 nm. Valon spektri on esitetty kuvassa 63. /195/



Kuva 63. Näkyvä valo /42/

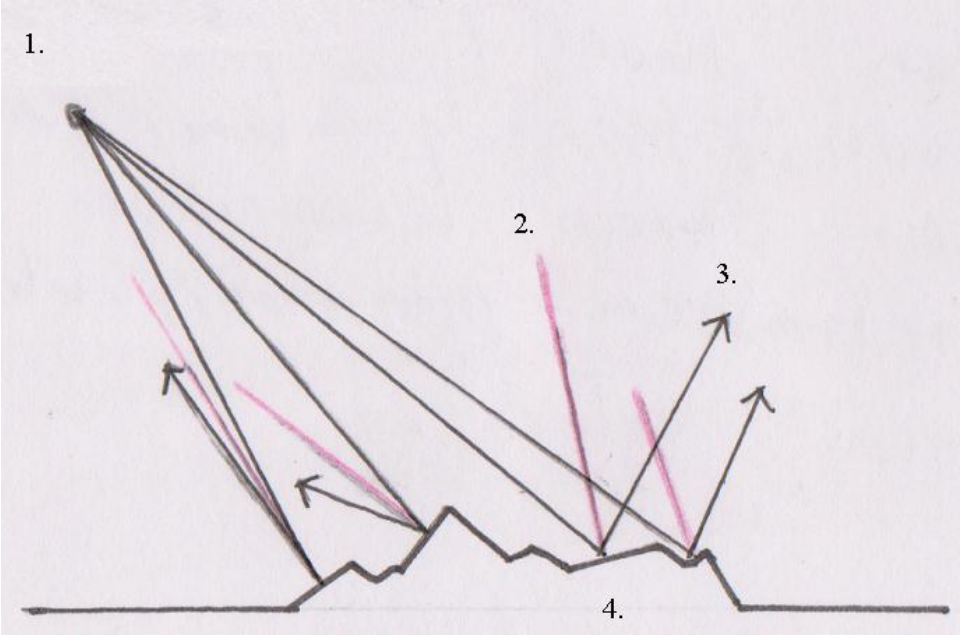
Valolla on aallonpituus (väri), amplitudi (kirkkaus) sekä värähtelykulma (polarisaatio).

Jotta kohde tulee näkyväksi, sen on heijastettava valoa. Valon voidaan ajatella kimpoavan törmätessään esteeseen. Valo heijastumisessa puhutaan tulo- ja heijastuskulmista. Kun tasaista pintaa kohtisuoraan tehdään suorakulma, puhutaan normaalista, heijastuskulma on samanasteinen kuin tulokulma, eri puolin normaalia. Tämä on havainnollistettu kuvassa 64. /195/



Kuva 64. Heijastuskulma on sama kuin tulokulma normaaliin nähden

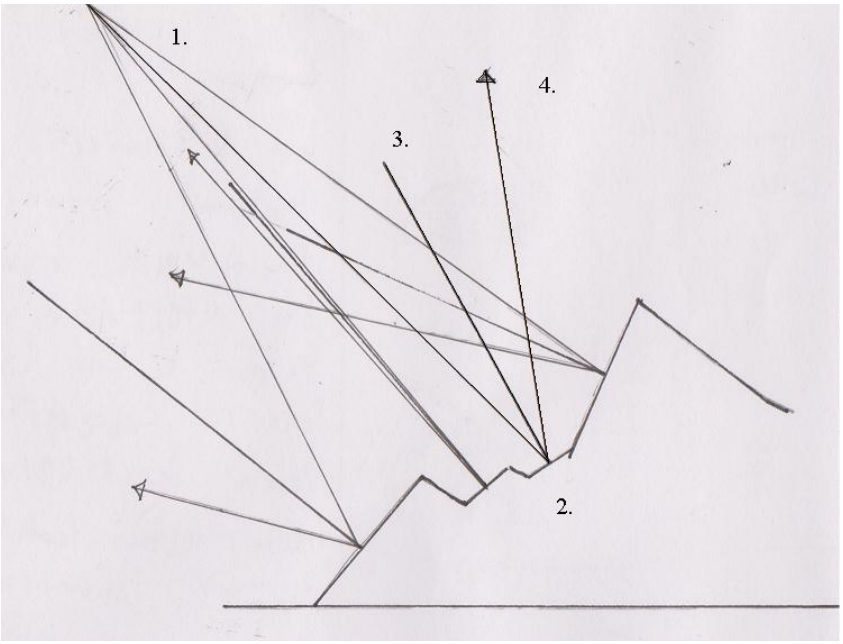
Tasaiselta pinnalta heijastuessa valo ei hajoa, vaan kaikki heijastuvat samassa kulmassa. Valon heijastuminen tapahtuu rosoiselta pinnalta samalla periaatteella kuin tasaiselta pinnalta, vaikka valo hajoaa erisuuntiin ja puhutaankin diffuusista (hajavalo). Valon osuessa pintaan se heijastuu samassa kulmassa tason normaaliin nähden. Valon heijastuminen diffuusoidusti on esitetty kuvissa 65 ja 66.



Kuva 65. Valon heijastuminen epätasaiselta pinnalta

Kuvassa 65 numeroidut kohdat ovat:

1. pistemäinen valonlähde
2. normaali pintaan nähden
3. heijastunut valo.



Kuva 66. Valon heijastuminen epätasaiselta pinnalta

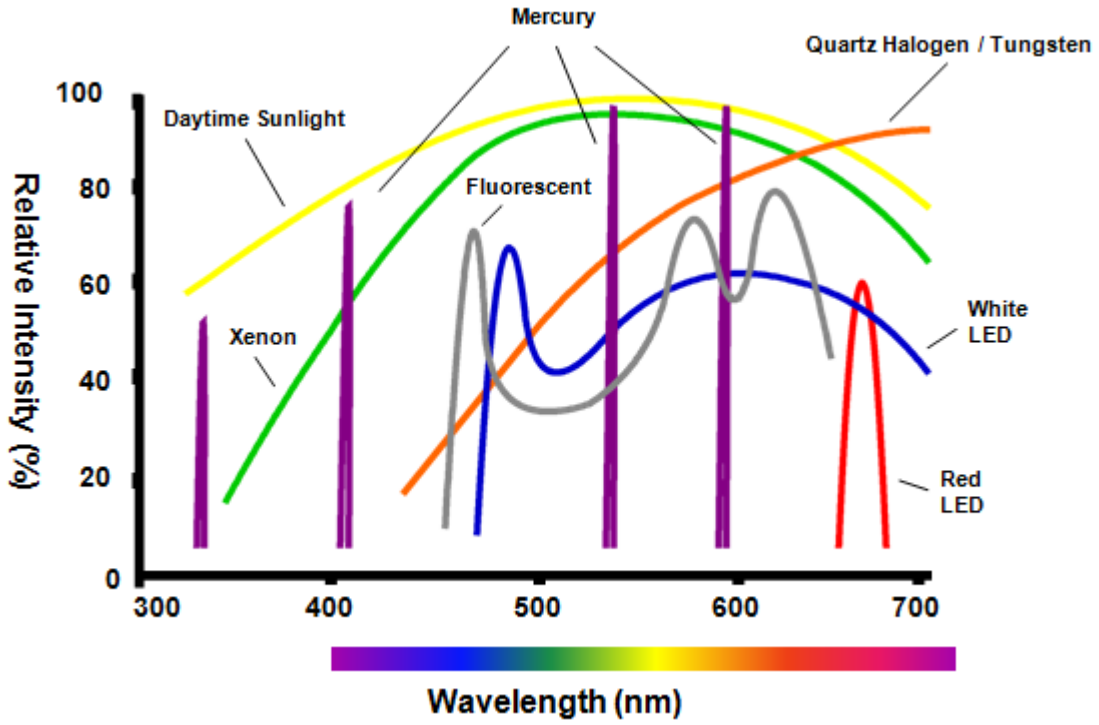
Kuvassa 66 numeroidut kohdat ovat:

1. pistemäinen valonlähde
2. kalteva pinta
3. normaali pintaan nähden
4. heijastunut valo.

Voisikin ajatella, että suunnattu valaistus on kuin aurinkoinen päivä, jolloin aurinko synnyttää varjoja ja pinnat kiiltävät. Diffuusoitu valo on kuin pilvinen päivä, jolloin varjot ja heijastukset katoavat auringonvalon hajotessa.

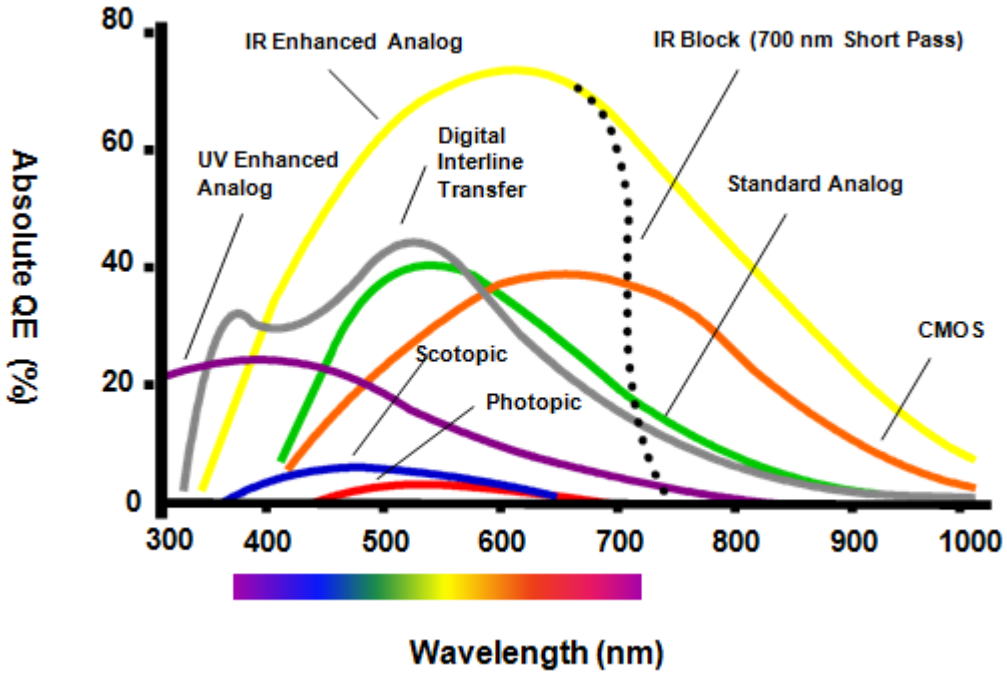
3.4.1. Valaisimet

Valon tuottamiseen käytetään erilaisia valaisimia, kuten laserdiodeja, halogeenilamppuja, ksenon- tai loistevalaisimia sekä LED-valoja. Jokainen näistä valoista koostuu eri määrästä eri aallonpituuksia. Kuvassa 67 on esitetty näiden valonlähteiden koostumusta (aallonpituuksia) suhteessa valon määrään. /103/



Kuva 67. Eri valaisimien valon aallonpituuksien määriä suhteutettuna toisiinsa /103/

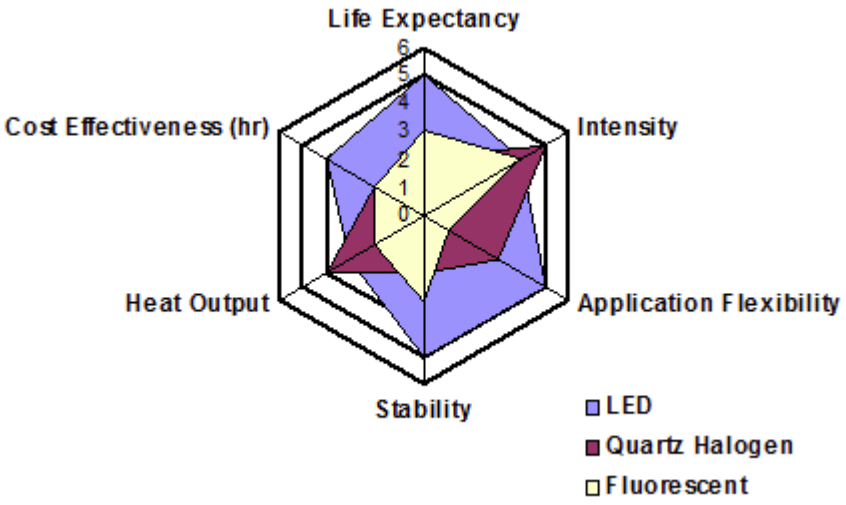
Kuvassa 68 on esitetty mitkä aallonpituudet havaitaan mitenkin hyvin eri kennoilla. Kuvassa aivan alhaalla on esitetty ihmisen silmän hämäränäkö (Scotopic) ja näkeminen päivänvalossa (Photopic), CMOS sekä CCD:n eri muotoja.



Kuva 68. Eri kameroiden aallonpituuksien havaitseminen /103/

Kuvista 67 ja 68 voidaan päätellä:

- Valittaessa valonlähdettä on hyvä tietää, miten hyvin kyseinen kenno kykenee sen valoa havaitsemaan.
- Kennojen infrapunan havaitseminen jatkuu hyvin pitkälle ihmisen havaintokyvyn jälkeen, mikä tulee myös ottaa huomioon ja käyttää tarvittaessa suodattimia. Infrapunaa voidaan myös käyttää hyödyksi.
- Eri suodattimia käyttäen voidaan häiritseviä taustavaloja, paitsi auringonvaloa, joka koostuu kaikista taajuuksista. Ainoa tapa estää se on estää fyysisesti valon pääsy kohteeseen.

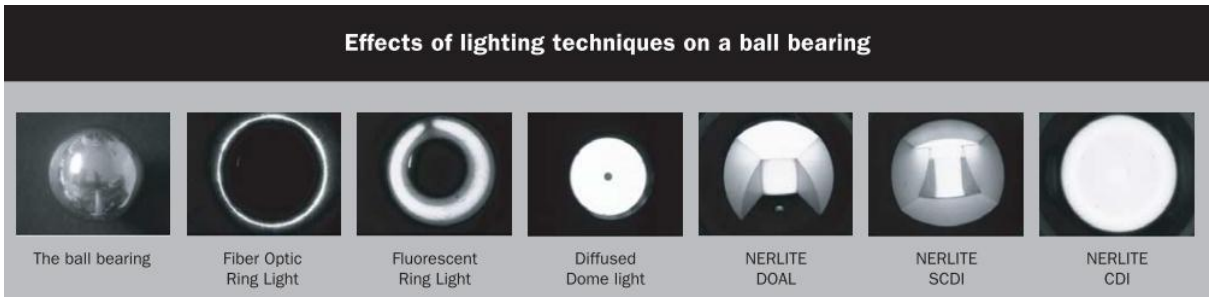


Kuva 69. Eri valonlähteiden vertailua /103/

Kuvasta 69 on esitetty LED, halogeeni ja loisteputken eri ominaisuuksia toisiinsa verrattuna. Halogeenit tuottavat kirkkaimman valon, mutta häviävät muilla aloilla paremmuudessa LED-valoille sekä loisteputkille. LED-valojen kehitys sekä laskeva hinta ovatkin viimeaikoina tehneet siitä johtavan konenäkövalaisussa käytettävän tekniikan ja korvaa hiljalleen muut tekniikat.

3.4.2. Valaistuksen tyypit

Kuvassa 70 on otettu samasta kuulalaakerista kuvia eri valotustekniikoilla. Eri tavoilla voidaan korostaa aivan erilaisia asioita ja näin edesauttaa tuloksen kannalta parhaan kuvan ja tuloksen syntymisen. /93/, /105/, /172/

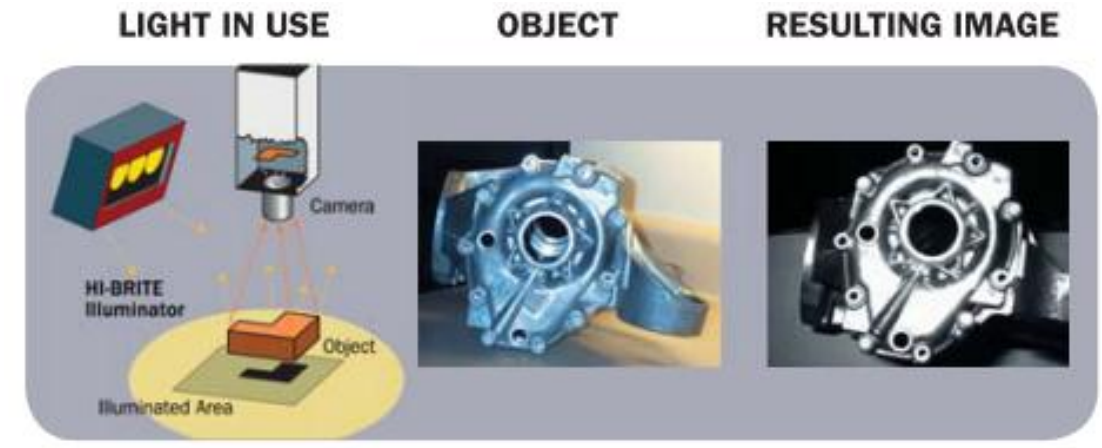


Kuva 70. Eri valotekniikoiden vaikutukset laakerin kuulaan /93/

Tässä kappaleessa käydään läpi näiden valaisutekniikoiden ominaisuuksia sekä mahdollisia käyttötarkoituksia.

Suunnattu valaistus

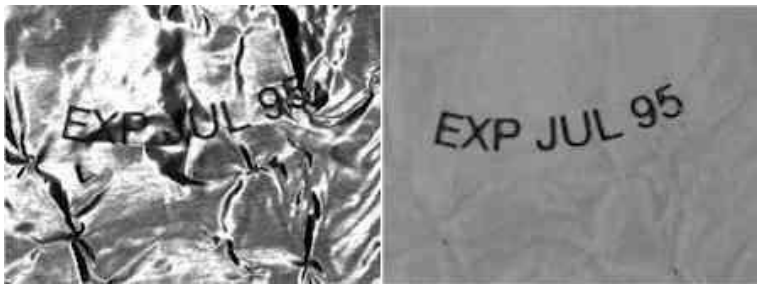
Suunnattu valaistus tarkoittaa valaistusta, jossa valonlähteen tuottama valo tuodaan kuvattavaan kappaleeseen kulmassa. Tämä aiheuttaa sen, että korkeuserot näkyvät varjostuksina. Suunnattua valaistusta voidaan käyttää epätasaisuuksien korostamiseen, mitä havainnollistetaan kuvassa 71. /27/, /54/, /95/



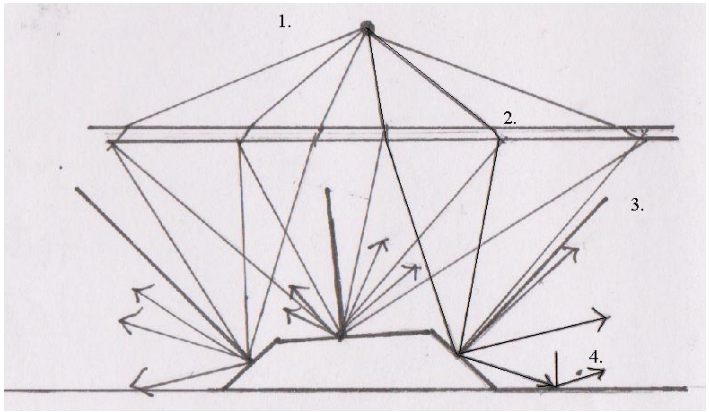
Kuva 71. Suunnattu valaistus /95/

Diffuusi valaistus

Diffuusoidulla valolla saadaan kohteesta tasaisesti valaistu, jolloin varjot ja voimakkaat heijastukset häviävät, kuten kuvassa 72 on käynyt. Valon hajottamiseen muutamia keinoja, joista tässä käsitellään diffuusointi levyllä, dome sekä on-axis. Kaikkien näiden on oltava suhteellisen lähellä valaistavaa kohdetta toimiakseen kunnolla. /27/, /54/



Kuva 72. Rypistynyt metallifolio diffuusoidussa valossa oikealla /29/



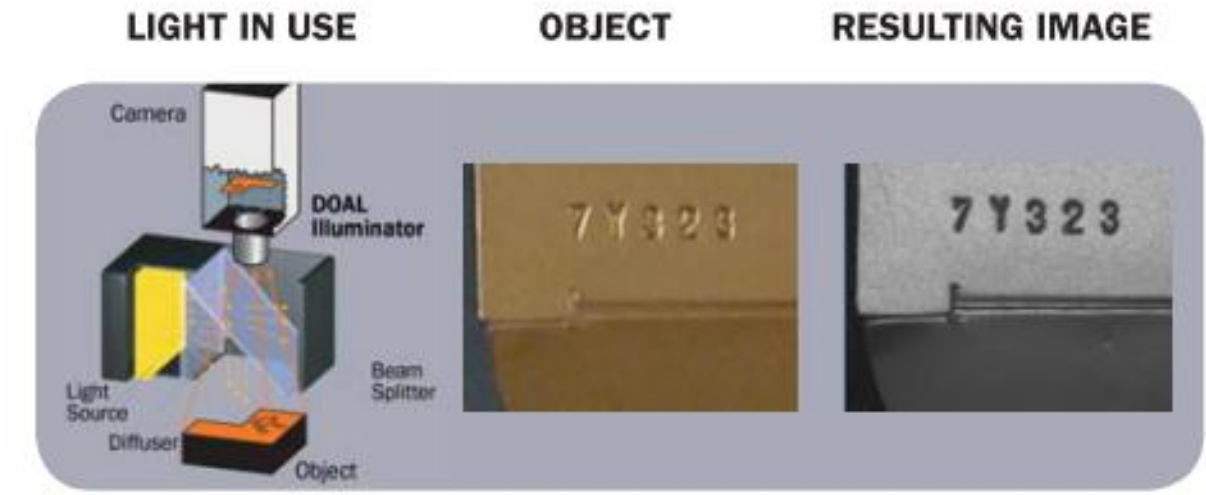
Kuva 73. Valo diffuusoidaan asettamalla kohteen ja valonlähteen välille valoa hajottava pinta

Kuvassa 73 on esitetty pistemäisen valon hajottaminen valoa läpäisevän tason avulla. Pistemäinen valonlähde säteilee joka suuntaan. Säteet, jotka osuvat levyyn kulkevat sen läpi. Poistuessa levystä valonsäde hajoaa säteilemällä joka suuntaan, jolloin syntyy tasainen valaistus. Kuvassa 73 numerot ovat:

1. ei diffuusoitu valonlähde
2. diffuusoiva levy
3. normaali tasoon nähden
4. valo heijastuu pinnasta joka suuntaan.

Olemassa on myös levyjä, joissa valo heijastetaan levyn sisälle, josta se valaisee tasaisesti. Tällaisilla levyillä saadaan tilaa säästäviä ja edullisia ratkaisuja. /172/

On-axis diffuse

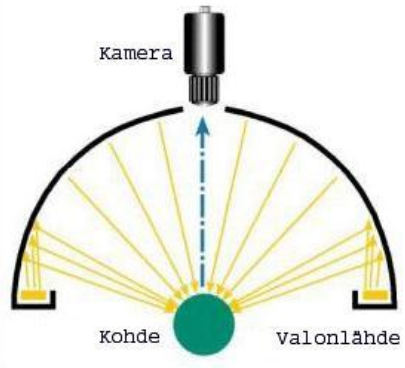


Kuva 74. on-axis diffuusoiva valo /95/

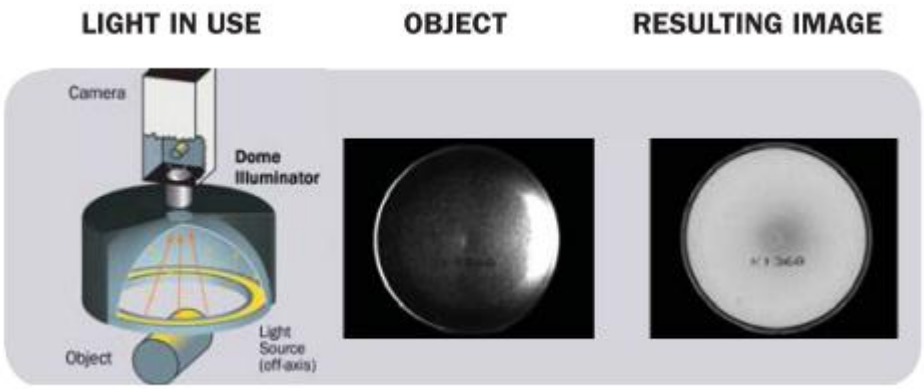
Kuva 74 esittää “Diffuse on-axis” valaistusjärjestelmä, joka on kehitetty nimenomaan konenäköjärjestelmiä varten. Siinä valo diffuusoidaan levyllä, hajotetaan kahtia prismoilla (beam splitter) ja näin saadaan aikaan tasainen valaistus ilman häiritseviä heijastuksia. Käytetään mm. tekstien, kuvien ja kulmien tarkastelussa. /27/, /54/, /94/, /95/, /105/, /175/

Kupolivalaistus (Dome)

Kupolivalaistuksessa valonlähteet tai lähde, käytettävästä tekniikasta riippuen, on sijoitettu kupolimaisen pinnan sisäpuolelle niin, että valo heijastuu kupolin sisäpinnan kautta kohteeseen. Kupolin pinnan ansiosta valo leviää tasaisesti vähäisin heijastuksin. Toimintaperiaate on esitetty kuvassa 75 ja vaikutuksia on 76. /27/, /29/, /54/, /95/



Kuva 75. Kupolivalaistuksen periaate /29/

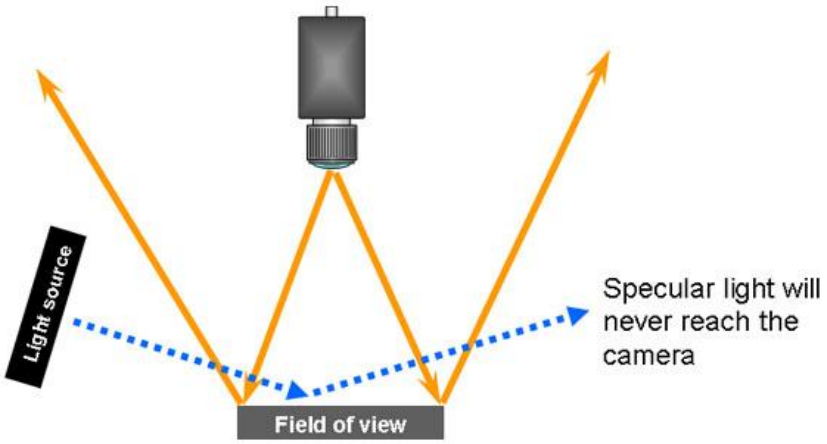


Kuva 76. Kupolivalaistus /95/

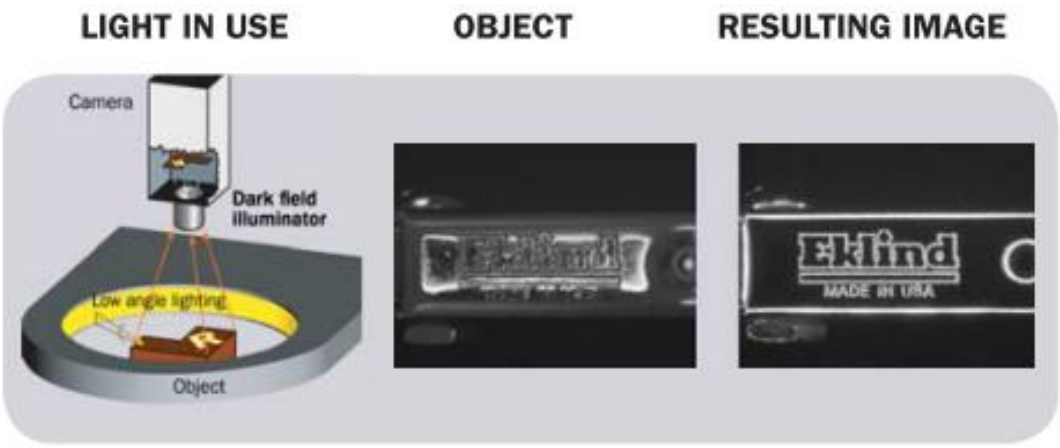
Olemassa on myös puolikupoleja sekä yhdistelmiä, joissa diffuse on-axis valaisin on yhdistetty domen kupolirakenteen kanssa. Tällöin saadaan molempien parhaimpia puolia käyttöön. /27/, /95/

Dark field-valaistus

Dark field-valaistuksessa kohdetta valaistaan sivuilta hyvin pienessä kulmassa, jolloin tasaisilta pinnoilta heijastuva valo heijastuu poispäin. Tämä aiheuttaa sen, että tasaisista pinnoista valo heijastuu pois kamerasta ja kohokuviot ja epätasaisuudet heijastuvat kameralle. Toimintaperiaate on esitetty kuvassa 77, valaistuksen vaikutus 78 ja 79. /27/, /29/, /54/, /95/, /96/



Kuva 77. Dark fieldin periaate /96/

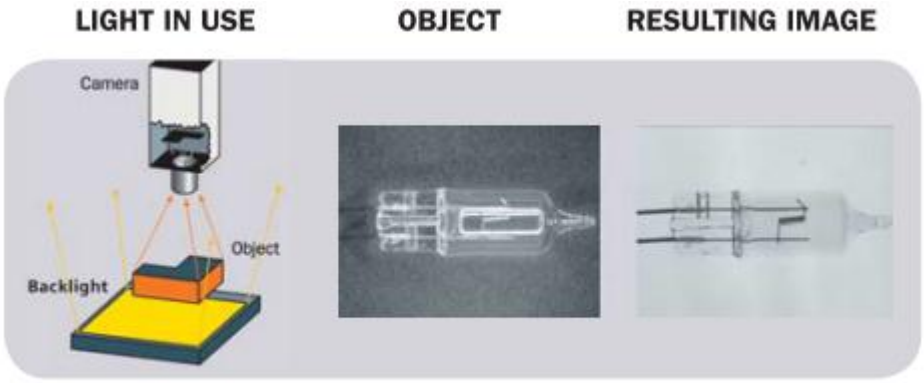


Kuva 78. Dark field valaisin /95/

Kuva 79. Kolikko dark field valaistuksessa /124/

Taustavalaisu

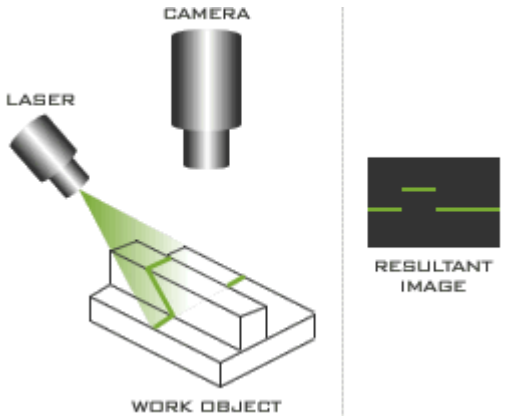
Taustavalaisussa valonlähde on kuvattavan kohteen takana, toisinsanoen kohde on kameran ja valonlähteen välissä, jolloin kappaleen reunat tulevat tarkasti näkyviin. Yleensä taustavalaisussa käytetään diffuusoitua valoa, sillä voimakkaan valon kohdatessa kappaleen se ei jatka matkaansa suoraan, vaan se voi ns. taipua, kiertää kappaleen reunan. Tällöin esimerkiksi mikropiirin reunoilla sijaitsevat liitosnastat saattavat kadota tai niiden reunat voivat olla sumeat tai olla näkymättä kuvassa kokonaan. Taustavalaistuksen vaikutuksia ja periaatteita on kuvattu kuvassa 80. /27/, /54/, /95/



Kuva 80. Taustavalaistus /95/

Rakenteellinen valaisu

Rakenteellisessa valaisussa (structured lighting) kohde valaistaan viivamaisella, viuhkamaisella valolla kulmassa kameraan nähden, kuten laserilla. Rakenteellisella valaistuksella voidaan kohteen korkeuseroja mitata helposti. Kun kuvia otettaan tarpeeksi monesta kohdasta kappaletta, voidaan kuvasta rakentaa kolmiulotteinen kuva. Toimintaperiaate on esitetty kuvassa 81. /29/



Kuva 81. Rakenteellinen valaisu /155/

Salamavalaistus, strobo

Salamavalaistusta käytetään kuvatessa liikkuvia kohteita ehkäisemään liikkeen aiheuttamaa epätarkkuutta kuvassa, kuin pysäyttämään liike kuvassa. Käyttäessä voimakasta valoa kameran kennolle valottuu sen hetken kuva. /31/

Strobo on salamavalo, joka välähtää nopeasti ja useasti peräkkäin. /31/

Kameroissa salamavalo voi välähtää kahdesti, joista ensimmäistä kutsutaan esisalamaksi. Sen tarkoitus on mitata valon heijastusta kohteesta, jotta voidaan laskea oikea “oikean” salavalon pituus. Salamavalon pituus on sekunnin sadas-, tuhannes- tai kymmenestuhannesosa. /31/

Tarvittava valotuksen pituus voidaan laskea kaavalla: /31/, /94/

$$(\alpha / n_{pikseli}) / v = s_{pulssi} \quad (3)$$

missä
α on kuvakulma (FOV), cm
$n_{pikseli}$ on pikseleiden määrä
v on kohteen nopeus (cm/s)
s_{pulssi} on tarvittavan valopulssin pituus

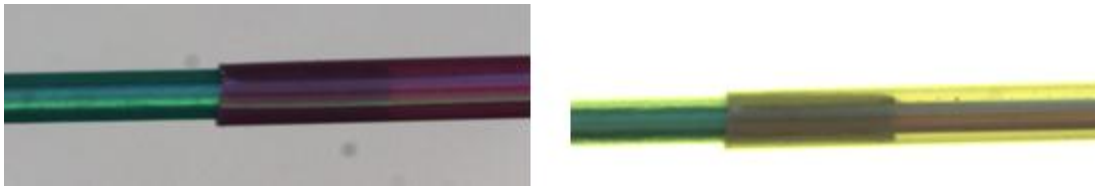
Ultravioletti- ja infrapunavalo

Ultravioletti- tai infrapunavalon käyttö konenäössä perustuu samaan kuin värien havaitseminen. Eri materiaalit heijastavat ja absorboivat eri aallonpituuksia erilailla. Valaistessa kohde toisella näistä valoista ja käyttämällä suodattamia poistamaan "näkyvä" valo, voidaan nähdä "näkymättömiä" kohteita, asioita jotka eivät näy normaalivalossa.

Ultraviolettivaloa käytettäessä voidaan puhua fluerosoivista aineista tai fluorenssistä sekä fosforoivista aineista sekä fosforenssista, joissa valo imeytyy aineeseen ja se alkaa emissoimaan valoa. Suodattimien avulla pinnasta voidaan erottaa eri aineita, kuten kuvassa 82 auton osia on maalattu eri maalilla. /196/, /197/

Kuva 82. UV-valolla voimme nähdä, että auton osia on vaihdettu/ 97/

Infrapunavalo tunkeutuu paremmin eri materiaalien läpi mitä näkyvä valo pitemmän aallonpituuden takia. Sen avulla voidaan nähdä joidenkin pintojen läpi, joita näkyvä valo ei läpäise, poistaa kuvattavista kohteista painettuja tekstejä, kuvia ja värejä paljastaen kohteen pinnan, kuten kuvassa 83 ja 84 on käynyt. /104/



Kuva 83. Oikealla taustavalona on käytetty infrapunavaloa, joka tunkeutuu muovin läpi ja paljastaa johdon /104/

Kuva 84. Vasemmalla infrapunavalolla valaistussa kuvassa teksti on kadonnut /148/

Polarisoitu valo

Valo on aaltomaista säteilyä. Polarisoitunut valo värähtelee vain yhdessä tasossa lineaarisesti tai ympyrämäisesti ja poistaa heijastuksia peilimäisiltä pinnoilta sekä näyttää kohteesta mekaanisia vikoja. /27/, /32/

Valonlähteen eteen laitetaan suodin, joka suodattaa kaikki muut paitsi suodattimen läpäisemän tason ja kameran eteen asetetaan vastakkaisesti polarisoitu suodin. Valon heijastuessa kohteesta se diffuusoituu ja polarisointi hajoaa, jolloin se läpäisee kameran edessä olevan

suotimen. Peilimäisestä pinnasta heijastuessa polarisaatio säilyy eikä valonsäde läpäise kameran suodinta. /27/, /32/

Pyöröpolarisaatiosuodinta käytetään poistamaan auringonvalon heijastukset, sillä osa maahan tulevasta valosta on polarisoitunut ilmakehässä. /32/

Valaistuksessa huomioitavia asioita

Valaistuksessa on hyvä valita sopiva tausta ja taustavalaisu, mikäli siihen voidaan vaikuttaa. Taustan avulla saadaan lisää kontrastia, jolloin voidaan selvästi erottaa kuvattavaan kappaleen reunat ja yksityiskohdat. /40/, /90/, /91/, /93/, /103/, /104/, /105/

Myös värien käyttäytyminen on hyvä muistaa valaistuksessa. Värit syntyvät, kun ne imevät muut aallonpituudet itseensä, jolloin vääränlaisilla aallonpituuksilla valaistaessa saadaan vääristyneitä tuloksia ja jotkin kohteet voivat olla näkymättä. Tätä voi tietenkin käyttää hyödyksikin joissain sovelluksissa. /40/, /90/, /91/, /93/, /94/, /103/, /104/, /105/

4. OHJELMOINTI

4.1. Ohjelmointi

Tietokoneiden prosessori lukee ja kirjoittaa käskyjä joita kone suorittaa. Käskyt koostuvat sarjoista ykkösiä ja nollia, ja tätä kutsutaan konekieleksi. Konekielellä ohjelmointi on vaikeaa. Työtä helpottamaan on kehitetty ohjelmointikieliä, kuten C, C++, Java ja Perl. Ohjelmointikielillä on omat merkkinsä, sanansa ja kielioppinsa ja ne on helpompi oppia kuin konekieli. Jokainen kieli on erilainen ja sopii eri tarkoituksiin.

Ohjelmointikielet toimivat ihmisen ja koneen välisenä siltana. Koodi "käännetään" konekielelle kääntäjällä tai tulkilla tilanteesta riippuen.

Tässä kappaleessa esitellään käytetyt ohjelmointikielet, näiden ominaisuuksia sekä eroja, yleisiä asioita ohjelmistosuunnittelusta sekä käyttöliittymistä.

4.1.1. Python

Python on nimetty komediaryhmän Monty Pythonin mukaan. Pythonin ensimmäinen virallinen versio julkaistiin 1995, toinen 2000 ja kolmas 2008. Toinen versio sisälsi automaattisen muistinkäsittelytoiminnot sekä merkkijonojen käsittelyä helpottavat ominaisuudet. Kolmannessa versiossa kielioppia (syntaksia) suoraviivaistettiin ja siitä poistettiin vanhentuneita toimintoja. Suurin osa Pythonin kirjastoista tukee edelleenkin vain toista versiota, vaikka kolmannen version julkaisusta on jo aikaa. /198/

Pythonin lisenssi on avoimen lähdekoodin lisenssi, se on ilmainen käyttää ja vapaasti jaettavissa myös kaupallisena.

Python on korkean tason ohjelmointikieli, jossa on panostettu erityisesti koodin luettavuuteen, jonka takia se sopiikin aloittelijalle hyvin. Muista ohjelmointikielistä poiketen Pythonissa käytetään sisennyksiä ja välilyöntejä koodilohkojen merkintään eikä sulkeita kuten monissa muissa kielissä on tapana. Pythonissa on automaattinen muistinhallinta, joten manuaalista muistin tyhjennystä tai muistin varauksia ei tarvitse tehdä. Se on alustariippumaton, joten sitä voidaan käyttää ongelmitta Windows-, Mac- ja useimpien Unix-järjestelmien kanssa. Pythonissa ei tarvitse määritellä muuttujan tietotyyppiä etukäteen, sitä voidaan muuttaa sopivaksi kesken ohjelman suorituksen. /68/, /86/, /134/, /135/

4.1.2. C++

C++, increased C, on ohjelmointikieli, jonka kehitti Bjarne Stroustrup 1980-luvulla C-kielen pohjalle lisäämällä uusia ominaisuuksia. C++ on yleiskäyttöinen ohjelmointikieli, jota voidaan käyttää lähes kaikkialla. C++ sai ensimmäisen standardinsa vuonna 1998 ja uusin standardi on tullut vuonna 2011: /190/

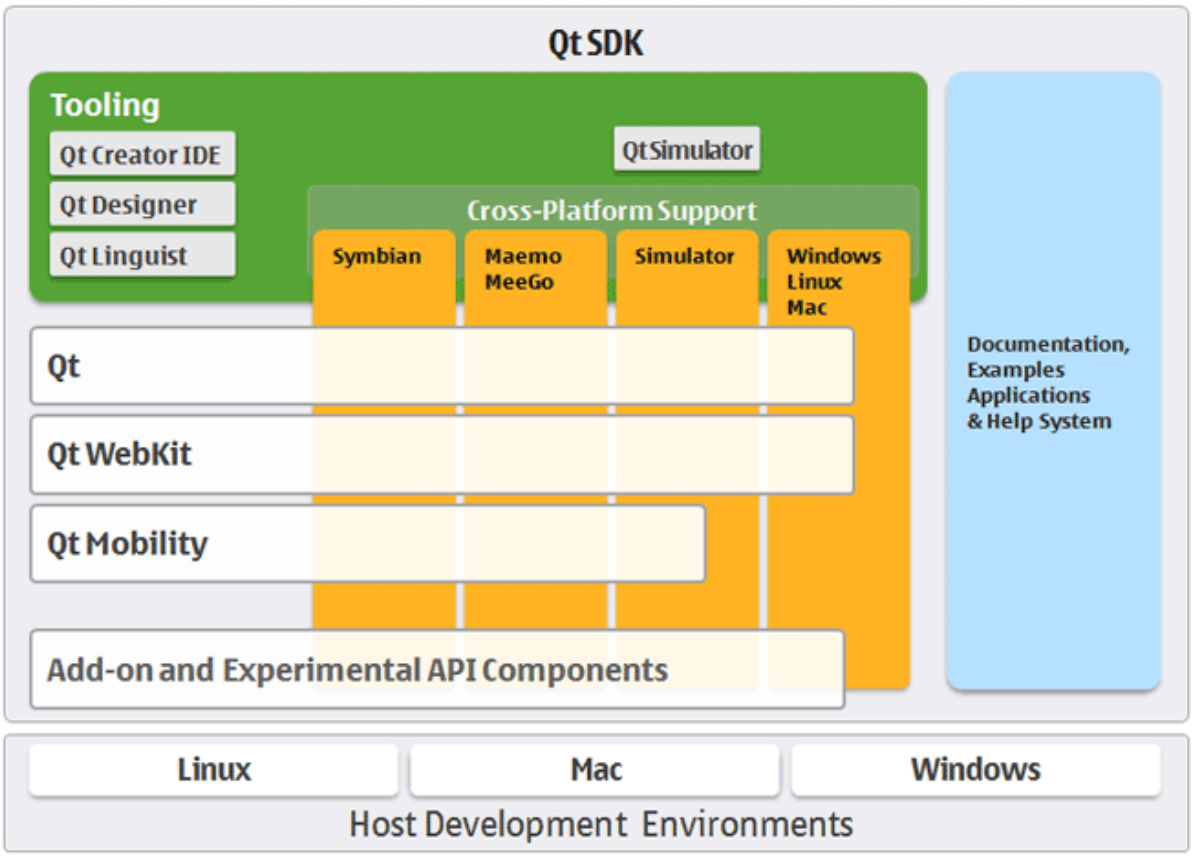
- ISO/IEC 14882:1998
- ISO/IEC 14882:2003
- ISO/IEC TR 19768:2007
- ISO/IEC 14882:2011

Karkeasti ajateltuna tärkeimmät lisätyt ominaisuudet ovat luokat ja oliot. Luokat koostuvat datan lisäksi toiminnoista. Näistä luokista luodaan ilmentymiä eli olioita. Jokaisen luokan tarkoitus on muodostaa toiminnallinen osakokonaisuus. Olio-ohjelmoinnissa ohjelmassa on joukko olioita, jotka ovat kytköksissä toisiinsa: kun yksi olio tekee jotain, niin seuraavan pitää tehdä näin. Oliot voivat vastaanottaa tietoa, käsitellä tiedon itse ja lähettää tietoa muille olioille. C++-kielessä ei ole automaattista muistinhallintaa kuten Pythonissa ja se on erittäin tarkka tietotyypeistään. /71/, /109/, /190/

4.1.3. Qt

Qt tai “cute” on alunperin norjalaisen Trolltechin kehittämä alustariippumaton ohjelmistojen sekä käyttöliittymien ohjelmointiympäristö, jolla voidaan ohjelmoida PC sekä sulautettuihin ympäristöihin ohjelmia muuttamatta lähdekoodia vain kääntämällä ohjelma eri ympäristössä. Nokia osti Qt:n vuonna 2008 ja teki siitä avoimemman, mm. lisenssit muuttuivat avoimeksi LGPL:ksi (Lesser General Public License) sekä kaupallisiksi lisensseiksi. /122/, /136/, /140/, /202/

Qt:llä on tehty mm. Google Earth, Skype, VideoLAN-mediasoitin sekä Opera-selain. SDK:n sisältöä ja tukea on havainnollistettu kuvassa 85.



Kuva 85. Nokia Qt SDK /136/

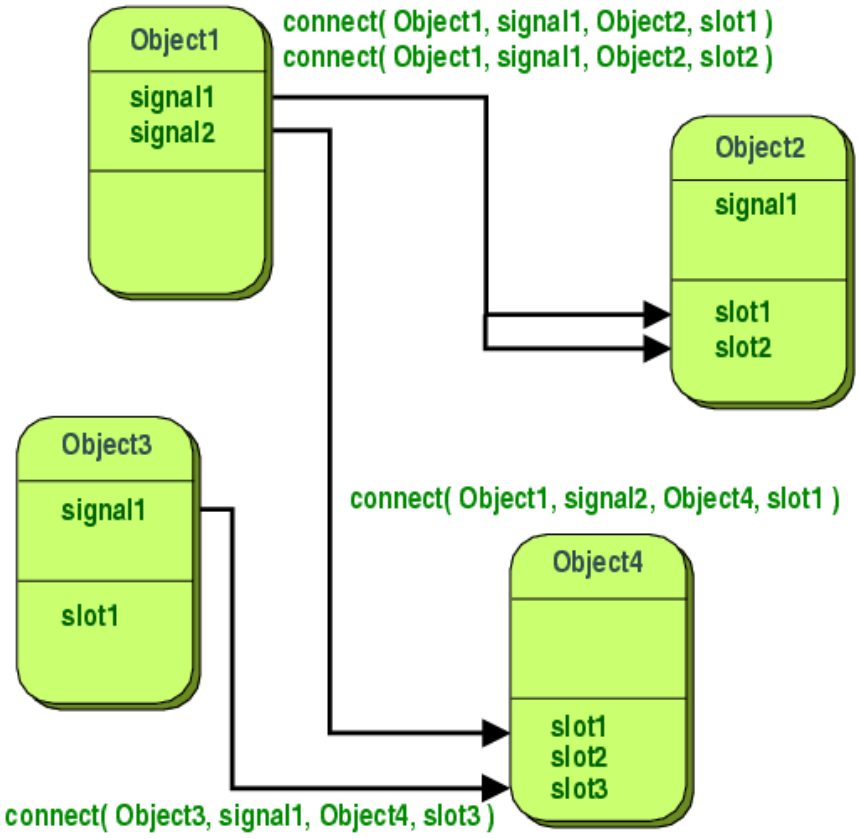
Qt:n ominaisuuksia

- Qt:n ohjelmat kirjoitetaan C++:lla ja käännetään suoraan konekielelle ilman virtuaalikoneita. /16/, /122/, /136/, /140/
- Qt:n ohjelmien lähdekoodit voidaan kääntää millä tahansa C++:aa ja käyttöjärjestelmää tukevalla kääntäjällä. /16/, /122/, /136/, /140/
- Kääntäjän tarvitsevat kuvaustiedostot (makefile) voidaan tehdä Qt:n omalla qmake-ohjelmalla. /16/, /122/, /136/, /140/
- Automaattinen muistinhallinta QObject:sta periytyville luokille. Esimerkiksi QWidget-luokat (graafiset käyttöliittymä luokat) periytyvät QObject:sta. /16/, /122/, /136/, /140/
- QString tukee Unicode-merkistöä 16-bittisinä QChar-merkkeinä. Tuettuja merkkejä ovat mm. heprea, kiina, arabia, suomi... Lisäksi on olemassa Qt Linguist, jonka avulla voidaan käyttäjälle muutettavat tekstit muuntaa helposti kieleltä toiselle. /16/, /122/, /136/, /140/
- Hyvä ja selkeä dokumentointi. /16/, /122/, /136/, /140/

Tapahtumapohjainen ohjelmointi

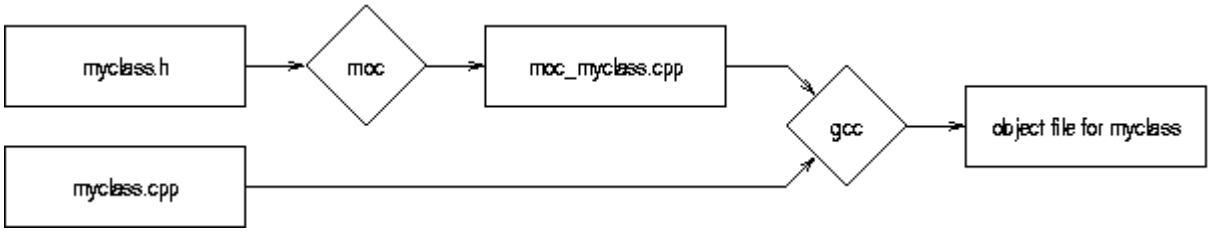
Suurin yksittäinen ominaisuus Qt:ssä on olioiden välinen viestintä, missä käytetään ns. “Signals & Slots” -mekanismia. Jonkin sisäisen tapahtuman johdosta lähetetään signaali (signal), jonka tapahtumankäsittelijä olio (slot) käsittelee. Tätä on kuvattu kuvassa 86. Yhden signaalin voi yhdistää moneen slottiin ja toisin päin. Nämä parit määritellään jo luokkamäärittelyn yhteydessä. /16/, /33/, /122/, /136/, /140/

Signaalin voi lähettää myös toiseen signaaliin, jolloin toinen signaali lähetetään välittömästi. /33/, /122/, /136/, /140/



Kuva 86. Signals & Slots /137/

"Signals & Slots" toimii vain, kun luokka toteuttaa Q_OBJECT-makron, joka kertoo esikääntäjälle, että luokka käyttää mekanismia ja käy läpi moc:n (meta-object code), joka muuntaa mekanismin C++:ksi. Tapahtuman vuokaavio on esitetty kuvassa 87. /33/, /122/, /136/, /137/, /140/



Kuva 87. Moc-työkalu muuntaa myclass.h:n C++:ksi /33/

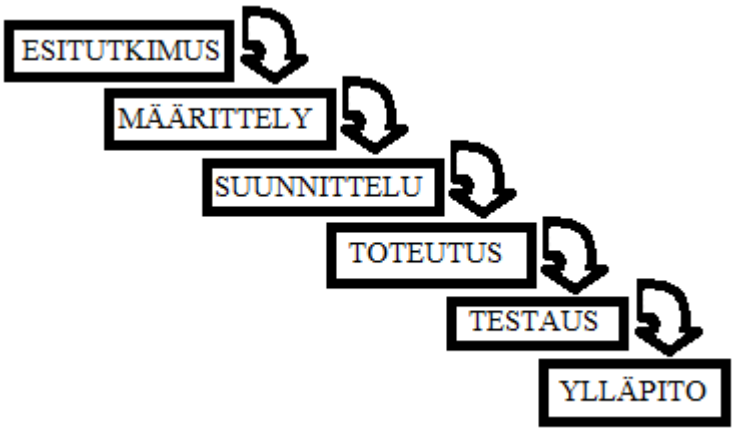
Qt Creator

Qt:lle on oma kehitysympäristö Qt Creator, jolla pystyy Qt:n lisäksi ohjelmoimaan normaalia C++. Qt Creator on alustariippumaton integroitu kehitysympäristö (IDE), joka toimii mm. Windows, Linux, Mac OS X käyttöjärjestelmissä. /16/, /122/, /136/, /140/

4.2. Ohjelmistosuunnittelu

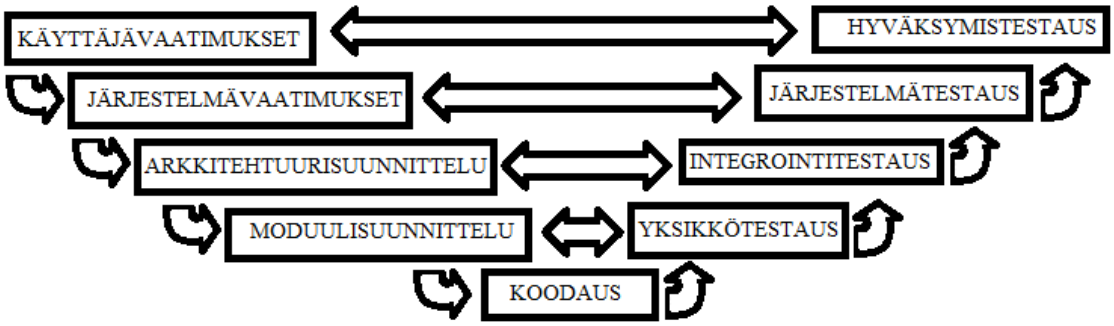
Aloittaessa ohjelman tekemisen, ohjelmistotuotannon, on hyvä suunnitella kuinka edetä loogisesti ja järjestelmällisesti projektin aikana. Tällaisia asioita ohjaamaan on useita malleja, jotka kaikki on pyritty esittämään ohjelman tuotannon etenemisen mukaan. Kaikille malleille löytyy yhteisiä vaiheita: Esitutkimus, määrittely, suunnittelu, toteutus ja testaus. /199/

Perinteisin näistä malleista on ns. vesiputousmalli, johon kaikkia muita malleja verrataan. Malli on esitetty kuvassa 88. Siinä toteutetaan prosessin yksi osa kerrallaan ja edetään loogisessa järjestyksessä. Edellisten vaiheiden tuloksia, dokumentteja käytetään hyödyksi tulevissa vaiheissa. /69/, /169/



Kuva 88. Vesiputousmalli

Vesiputousmallin heikkoutena on sen joustamattomuus ja mukautumattomuus. Hyvässä ohjelmistotuotannossa prosessissa palataan takaisin aikaisempiin vaiheisiin (iteraatio) ongelmatilanteissa korjaamaan tilannetta. Tunnetuin tällainen malli on V-malli, joka on esitetty kuvassa 89.



Kuva 89. V-malli

Ketterät menetelmät

Ketterien menetelmien periaatteita ovat nopeus ja yksinkertaisuus, sekä vahva vuorovaikutus asiakkaan kanssa. Yleensä ohjelmasta tehdään "runkoversio", joka sisältää vain kaikista oleellisimmat toiminnot. Tätä versiota päivitetään lisäten aina ominaisuuksia ja korjaten vikoja. Menetelmä on yleistynyt internetin sekä matkapuhelinten myötä. Tässä työssä ei käydä läpi, millaisia ketteriä menetelmiä on olemassa. /139/, /156/, /161/

Kaikille menetelmille yhteisiä piirteitä ovat: /139/, /156/, /161/

- iteratiivisuus (tiheät syklit)
- inkrementaalisuus (jatkuvasti arvioitavaa tulosta)
- testauksen automatisointi (test first)
- pyritään sallimaan ja hallitsemaan muutoksia
- kommunikoinnin, yhteistyön ja vuorovaikutuksen korostaminen

- suoraviivaisuus.

Ketteryys vähenee järjestelmän kasvaessa ja monimutkaistuessa. Tätä koetetaan ehkäistä suuremmissa projekteissa jakamalla se pienempiin, itsenäisiin osiin, jolloin sitä voidaan hallita paremmin. /139/, /156/, /161/

4.2.1. Scrum

Yleensä käytetään, usein tietämättä "Scrum" menetelmää. Siinä projekti on jaettu lyhyisiin samanmittaisiin aikajaksoihin (sprintit), joiden aikana toteutetaan määrätyt tehtävät. Scrum-mallisessa menetelmässä henkilöt on jaettu 5-9 hengen ryhmiin, ja jokaisessa ryhmässä on: /69/, /100/, /201/

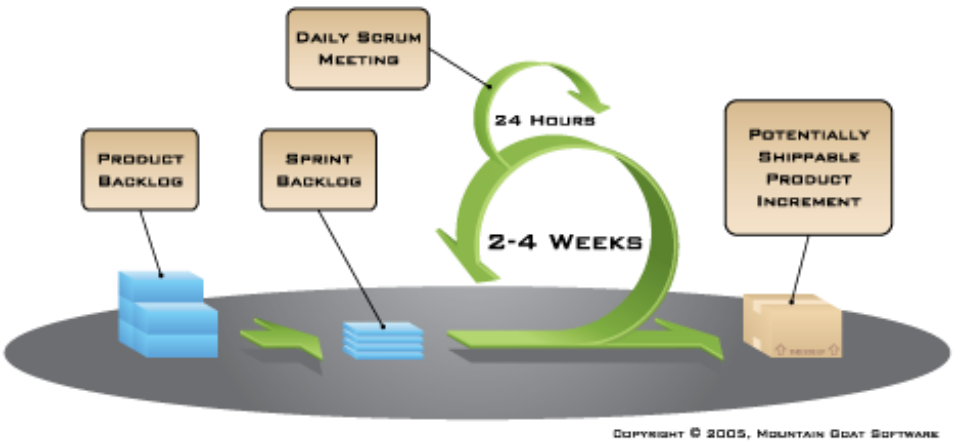
- **Tuoteomistaja**, joka vastaa siitä, että rahalle saadaan vastinetta sprintin aikana.
- **Kehitystiimi**, joka suorittaa ohjelmiston kehittämisen. Tiimin jäsenillä voi olla omat tehtävät, mutta vastaavat yhdessä ohjelmiston kehittämisestä.
- **Scrummaster**, joka seuraa sprintin kehitystä sekä katsoo, että pelisääntöjä noudatetaan. Mikäli hän pelkää aikataulun pettävän, pidetään kokous jossa katsotaan, poistetaanko sprintistä tehtäviä.

Scrumin keskeiset dokumentit: /69/, /100/, /201/

- **Tuotteen kehitysjono:** Tuoteomistajan mukautuva lista kaikesta, mitä tuotteessa saatetaan tarvita. Se toimii perustana tuotteen vaatimuksille sekä sen muutoksille.
- **Julkaisusuunnitelma:** Tuotteen kehitysjono, joka sisältää tulevat ominaisuudet seuraavaan julkaisuun sekä mahdollisesti niihin tarvittavien sprinttien määrä.
- **Sprintin tehtävälista:** Lista suunnittelupalaverissa tuotelistasta valittuja kohtia, jotka kehitystiimi toteuttaa valmiiksi sprintin aikana. Kehitystiimi voi muuttaa listaa saavuttaakseen sprintin tavoitteet.

Scrumin tapahtumat: /69/, /100/, /201/

- **Sprintti:** Maksimissaan kuukauden kestävä sykli, jonka päätyttyä on valmis, julkaisukelpoinen versio ohjelmistosta. Sprintin päätyttyä uusi alkaa välittömästi.
- **Sprintin suunnittelupalaveri:** Palaveri, jossa suunnitellaan sprintin aikana tehtävä työ. Suunnitteluun käytetään maksimissaan kahdeksan tuntia kestävä prosessi.
- **Päiväpalaveri:** Kehitystiimin oma päivittäinen palaveri, jossa jokainen kertoo lyhyesti mitä on tehnyt sitten viime palaverin, mitä aikoo tehdä ennen seuraavaa sekä onko tullut ongelmia. Palaverin tarkoituksena on olla erittäin lyhyt, maksimissaan 15-minuuttinen, mutta siinä voidaan sopia mahdollisista lisäpalavereista.
- **Tuotteen kehitysjonon työstö:** Kehitysjonon katselmointi ja arviointi, missä tuoteomistaja kehitystiimin kanssa lisää kehitysjonoon yksityiskohtia sekä työmääräarvioita. Työmääräarvioita antaa vain kehitystiimi.
- **Sprinttikatselmus:** Sprintin lopussa scrum-tiimi sekä sidosryhmät katselmoivat kehitettyä tuoteversiota ja muuttavat tarvittaessa kehitysjonoa sen mukaan.
- **Sprintin retrospektiivi:** katselmuksen ja seuraavan suunnittelupalaverin välissä scrum-tiimit pohtivat työtään ja tekevät suunnitelmia parantamaan seuraavaa sprinttiä.



Kuva 90. Scrum /100/

Kuvassa 90 on esitettynä Scrumin vaiheita. Vasemmalta alkaen tuotteen kehitysjono, sprintin tehtävälista, 2-4 viikkoa kestävä sprint, jonka tuloksena mahdollisesti julkaistava tuotteen lisäosa. /201/

Hallintatyökalut

Ohjelmistoprojektin tärkeimpiä työkaluja ovat versionhallinta, bugikirjasto sekä wiki tai muu vastaavan tyylinen systeemi dokumentaatiolle. /212/

Versionhallinnalla tarkoitetaan, että tehdessä muutoksia esimerkiksi suunnitelmiin, piirustuksiin tai lähdekoodiin vanha versio säilytetään ja muutokset kirjataan tavalla tai toisella ylös. Yleensä versionhallintaohjelmistoissa on mukana ominaisuus, joka vertaa uutta dokumenttia vanhaan ja kertoo muutokset. Versionhallinnassa on tärkeää käyttää ennalta sovittuja menetelmiä esimerkiksi tiedostojen nimeämisissä. /212/

Ohjelmistosuunnittelussa käytössä on myös ns. bugikirjasto, johon kerätään kaikki tiedot esimerkiksi ohjelmien virheistä ja kaatumisista, mahdollisista jatkotoimenpiteistä sekä niiden ratkaisuista. /212/

4.2.2. Graafinen käyttöliittymä

Käyttöliittymistä usein puhutaan lyhenteellä UI (User Interface) tai GUI (Graphical User Interface). Näiden tarkoituksena on toimia siltana ihmisen ja käytettävän ohjelman tai laitteen välissä. Tekstipohjaisessa käyttöliittymässä ohjelmaa ohjataan kirjaimin tai tekstein, kun taas graafisessa käyttöliittymässä (GUI) käyttäjä ohjaa ohjelmaa kuvien, painikkeiden ja osoittimien avulla. Tässä kappaleessa käydään hieman läpi GUI:n suunnitteluun ja tekemiseen liittyviä asioita. /75/, /202/

Alussa oli tekstirivi

Monet uskovat, että GUI:n käytön aloitti Apple vuonna 1984, kun se esitteli hiirellä ohjattavan tietokoneensa, jossa oli graafinen käyttöliittymä. Kuitenkin hiiri itsessään oli keksitty jo vuonna 1963 ja ensimmäinen GUI:lla toimiva tietokone, Xerox Alto, kehitettiin 11 vuotta ennen Applea vuonna 1973 Xerox PARC:n toimesta. Vuosia myöhemmin Apple, Microsoft sekä monet muut kopioivat hyvän idean kaupalliseen käyttöön. /26/, /200/

Nykyään graafiseen käyttöliittymään ei kuulu enää välttämättä hiiren osoitin (pointer), sillä esimerkiksi puhelimien ja kosketusnäyttöjen myötä sitä ei tarvita osoittamaan valittavia kohteita.

Käyttöliittymän lähtökohtana on käyttäjän toiminta

Käyttöliittymää suunnitellessa on otettava huomioon tekniikan tuomat rajoitteet, kuten näytön koko ja I/O-laitteet, käyttäjien mahdolliset totutut tavat, esimerkiksi käyttöjärjestelmäkohtaiset ominaisuudet, käytetyt värit taustoissa ja teksteissä, mutta myös käytettävyys, yksinkertaisuus, ohjeistus, varoitus sekä kommunikointi käyttäjän kanssa. /70/, /74/, /138/, /149/, /170/

Usein ohjelmatuotannossa käytettävissä prosessimalleissa ei oteta käyttöliittymää tai sen kehittämistä huomioon.

Hyvän käyttöliittymän ominaisuuksia: /70, s.38/, /76/, /138/, /170/

- Se on yksinkertainen.
- Käyttää yhtä kieltä, myös dokumenteissa (käyttäjän kieltä) sekä vakiintuneita termejä.
- Ohjelmaa käyttäessä käyttäjän ei tarvitse muistaa, mitä on tehnyt aiemmin ohjelmassa.
- Sen tulee olla aina samantyylinen ja yhdenmukainen läpi ohjelman.
- Selkeä navigoitavuus: Missä ollaan menossa, mistä ollaan tultu ja mihin voidaan mennä.
- Poistumistie kaikista tilanteista, peruutusmahdollisuus.
- Antaa käyttäjän tehdä mitä haluaa, esim. hypätä jonkun osan ylitse.

- Ehkäisee mahdolliset virhe- ja vaaratilanteet.
- Virhe- ja vaaratilanteissa antaa selkeät ilmoitukset ja ohjeet käyttäjälle.
- Virhetilanteissa tulee tarjota vaivaton virheenkorjaus.
- Riittävät ohjeet sekä dokumentaatiot ohjelman toiminnasta sekä käytöstä.
- Käyttäjän inhimilliset virheet tulee ottaa huomioon.

Käyttöliittymää suunnitellessa on tärkeää muistaa käyttäjän toiveet sekä vaatimukset ja mahdollisesti testata sekä kysyä käyttäjältä mielipiteitä eri ratkaisuvaihtoehdoista. /76/, /149/, /170/

Käyttöjärjestelmää tehdessä ensimmäiseksi täytyy olla tiedossa toiminnallisuus, ns. navigointimalli: mitä tehdään, millaisia ominaisuuksia ikkunoilla on ja millaisia komentoja. Navigointimalliin piirretään vain toiminnallisuus sekä ikkunoiden ja toimintojen riippuvuus toistensa välillä. /75/, /149/, /170/

Tämän jälkeen jokaisesta ikkunasta on hyvä tehdä yksinkertainen luonnos, jossa ryhmitellään toiminnat ikkunaan. Luonnoksen käyttöliittymästä voi suunnitteluvaiheessa piirtää käsin ja myöhemmin esimerkiksi grafiikkaohjelmia käyttäen. /75/, /149/, /170/

4.3. Kuvankäsittely

Kuvankäsittely (image processing) konenäössä voidaan sekoittaa suomenkielessä valokuvien kuvankäsittelyyn (image editing). Valokuvien kuvankäsittelyssä kuvaa pyritään muokkaamaan ja muuttamaan, kun taas kuvankäsittelyllä konenäössä siitä koetetaan etsiä informaatiota.

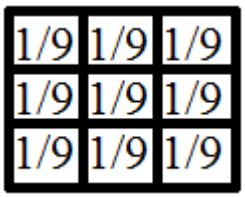
Konenäössä kuvankäsittelyyn menevää aikaa voidaan laskea huomattavasti panostamalla kameraan, optiikkaan sekä oikeanlaiseen valaistukseen. Esimerkiksi kennon pikseleiden määrä sekä mahdolliset värit vaikuttavat oleellisesti laskennan määrään, mutta objektiivien laatu

vaikuttaa kuvan geometrisiin virheisiin ja valojen avulla voidaan tuoda selkeästi kohde esiin, jolloin sitä ei tarvitse etsiä. /49/, /58/, /77/, /83/, /123/, /169/, /202/

Kuvankäsittelyssä pyritään etenemään loogisesti siten, että kuvaa “siistitään” ja korjaillaan ennen raskaimpia, eniten laskentatehoa vieviä tapahtumia. Kuvankäsittelyn tavat voidaan jakaa käyttötarkoituksien mukaan kuvatyyppimuunnoksiin, esikäsittelyyn, binarisointiin (kynnystys), segmentointiin, reunanhakuun, piirteiden ja ominaisuuksien tunnistukseen sekä tulosten käsittelyyn. /41/, /49/, /58/, /77/, /83/, /123/, /169/, /202/

Kernel / Mask

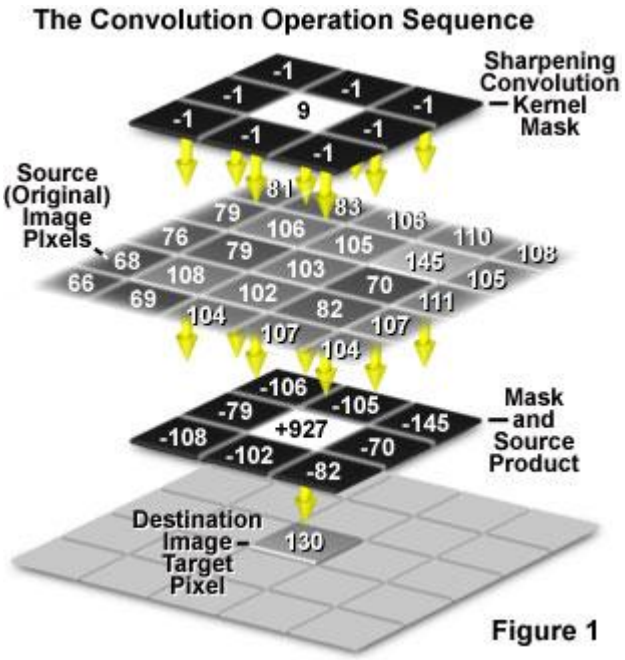
Kerneliksi tai maskiksi kutsutaan matriisia, jossa matriisin arvoiksi on annettu jotkin tietyt kerroinarvot. Yleisin kerneli on 3x3 pikseliä (kuva 91), mutta olemassa on myös 5x5, 7x7 sekä ympyrän että suorakulmion muotoisia. Kerneliä käytetään **konvoluutiossa**. /64/, /99/



Kuva 91. 3x3 kernel

Konvoluutio

Konvoluutiolla tarkoitetaan operaatiota, jossa lasketaan ennalta määrättyyn pisteeseen tietyn kernelin arvojen painotettujen tulojen summa. /60/, /64/



Kuva 92. Kuvassa on esitetty konvoluutio pikselille /99/

Kuvassa 92 kernelissä ankkuripikselissä kertoimena 9 ja muissa -1. Kerneli asetetaan kuvan päälle ja alle jäävät pikselien arvot kerrotaan kernelin kertoimilla ja summataan yhteen:

*pikseli = (-1)*106 + (-1)*105 + (-1)*145 + (-1)*145 + (-1)*70 + (-1)*82 + (-1)*82 + (-1)*102 + (-1)*108 + (-1)*79 + 9*103*

= 130

Konvoluution ongelmana on reunapikselit, sillä kernelille ei riitä pikseleitä kerrottaviksi. Kuvaan reunan pikselit ovat siis konvoluution jäljiltä virheelliset.

Konvoluution avulla kuvalle voidaan suorittaa erilaisia operaatioita, kuten suodatuksia tai korostuksia muuttamalla kernelin arvoja tai kokoa.

4.3.1. Kuvatyypit ja niiden muunnokset

Kameralla voidaan pääsääntöisesti siis ottaa joko harmaasävykuvia tai värikuvia. Konenäössä harvoin tarvitaan esimerkiksi kaikkia värikanavia, saati värejä ollenkaan. Tätä varten ne täytyy muuttaa harmaasävykuviksi ja mahdollisesti vielä binäärikuviksi. /60/, /169/

Kuten kameroiden yhteydessä kävi ilmi, kamerat eivät havaitse värejä, vaan ainoastaan valon määrän. Täten kolmikanavainen värikuva RGB koostuu kolmesta harmaasävykuvasta, joissa jokaisessa on eritavoin valottunut kuva, riippuen värisuodattimien läpi pääsemästä valon määrästä. Yksinkertaisimmillaan muunnos värikuvasta harmaasävykuvaksi on ottaa vain yksi värikanava ja käyttää sitä harmaasävykuvana. Tällainen on tosin järkevää vain, kun tahdotaan tarkastella vain tietyn värin, esimerkiksi punaisen määrää kuvassa. Toinen tapa on laskea pikselikohtaisesti kaikkien värien keskiarvo ja sijoittaa tämä pikseliin. /60/, /64/, /169/

Harmaasävykuva eroaa 24-bittisestä värikuvasta vain siten, että siinä on yksi “värikanava”. Harmaasävykuvat ovat yleensä 8-bittisiä, eli pikseli voi saada arvon 0-255, mutta myös mikä tahansa muu toisen potenssin luku, kuten 4-, 6-, 10- tai 12-bittinen. Väriskaala tosin on vain musta, valkoinen tai jokin sävy näiden väliltä. /60/, /64/, /169/

Binäärikuvaa on toisinsanoen musta-valkoinen kuva, siis siinä ei ole mitään muita värejä kuin musta tai valkoinen. Tällainen kuva saadaan aikaan vain kynnystämällä harmaasävykuva, jolloin valitaan raja-arvo (threshold), jonka yli olevat värit ovat muuttuvat valkoisiksi ja kaikki muu mustaksi. Tämä on yleinen konenäkösovelluksissa, sillä tulosten käsittely helpottuu, kun ratkaisumahdollisuuksia on vain kaksi esimerkiksi harmaasävykuvan 255 sijaan. /60/, /64/, /169/

Kuvatyyppien muuntaminen binääristä harmaasävyksi ja siitä värikuvaksi kannattaa vain silloin, kun kuvaan halutaan tuoda esimerkiksi värejä tai sävyjä korostamaan tuloksia ihmiselle. /60/, /169/

4.3.2. Kalibrointi

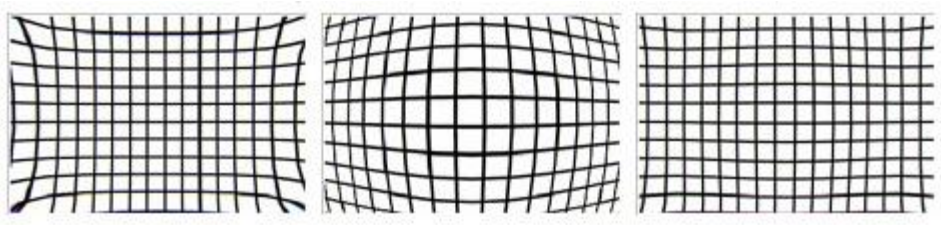
Kalibroinnin merkitys konenäköjärjestelmissä nousee vaadittavan tarkkuuden sekä laitteiden, kuten kameran, valaistuksen sekä kameran optiikan huonon laadun vuoksi. Kalibroinnin tarkoituksena on poistaa näistä aiheutuvat virheet, kuten optiset sekä geometriset vääristymiset ja mitata pikseleiden pituus "luonnossa". /49/, /72/, /78/, /83/

Optiset vääristymät

Optiset vääristymät aiheutuvat kameran optiikka, linssit tai linssistöt. Tyypillisimpiä vääristymiä ovat: /30/, /38/, /49/, /61/

- Tynnyrivääristymä (barrel), joka aiheutuu laajakuvakulmaobjektiiveista.
- Neulatyynyvääristymä (Pincoshion), joka aiheutuu tele/zoom-objektiiveista.
- Näiden yhdistelmistä, ns. mustache tai complex distortion.
- Reunapehmeneminen, eli reunojen epätarkkuus, johtuu linssien reunojen aiheuttamista vääristymistä.
- Väriväristymät, jotka aiheutuvat aallonpituuksien eri taittumiskulmista kuten valon osuessa prismaan.

Vääristymät ovat esitettyinä kuvassa 93, missä oikealta lähtien: neulatyynyvääristymä, tynnyrivääristymä ja näiden yhdistelmä.



Kuva 93. Optisia vääristymiä /38/

Geometriset virheet

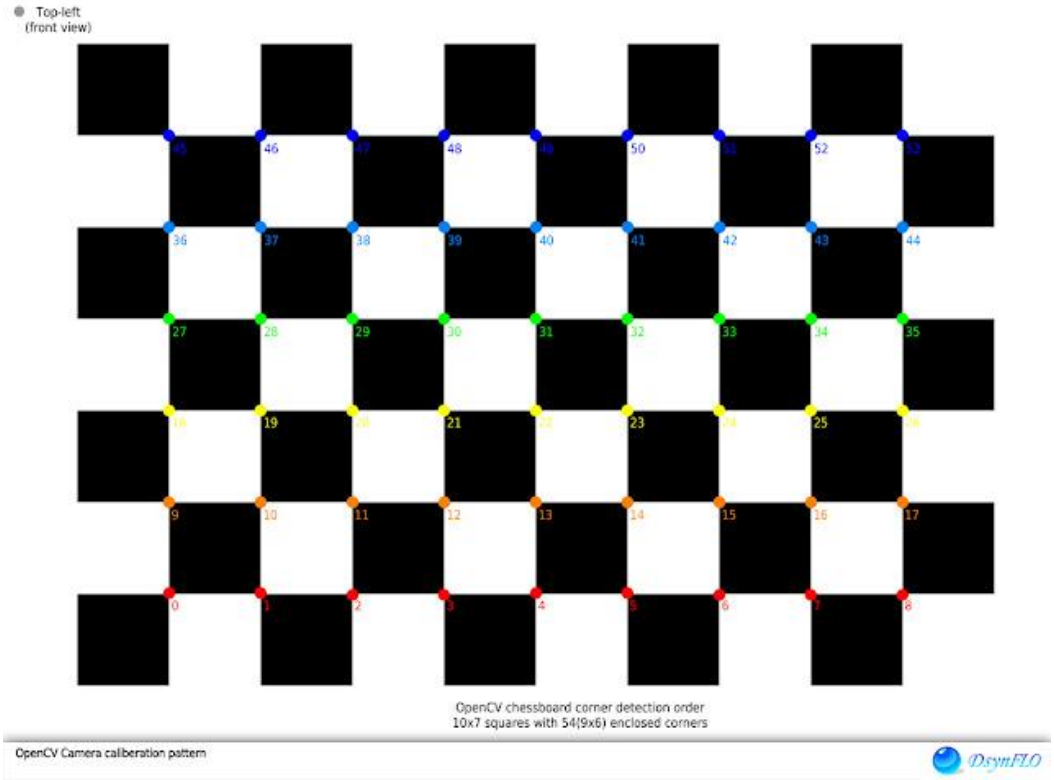
Geometriset virheet syntyvät, kun kolmiulotteisia kohteita koetetaan esittää kaksiulotteisessa kuvassa. Voidaan puhua myös perspektiivin virheistä. Esimerkiksi kun kameraa lähempänä olevat näyttävät isommilta kuin kauempana olevat, rei'istä näkyvät sisäreunat tai kohde on kaltevasti kameraan nähden. Voisi ajatella, että kuvaaja istuu tuolissa ja pöydän toisella puolen on paperi. Korjauksen jälkeen kuvassa on vain tuo paperi "oikaistuna". /49/, /60/, /77/, /78/

Skaalauskerroin

Etäisyyksiä ja mittoja otettaessa kuvasta täytyy muodostaa ns. skaalauskerroin, paljonko kuvan pikselin mitta on luonnossa. Skaalauskerroin saadaan aikaiseksi ns. kalibrointilevyllä. Kalibrointilevyssä on erittäin tarkasti mitoitettuja kuviota, kuten neliöitä, ympyröitä tai vaikka shakkiruudukko ja näiden kuvioiden koot tunnetaan. Otettaessa kuvia vakioetäisyydeltä saadaan tietää, kuinka monta pikseliä mikäkin mitta on. OpenCV:ssä tähän käytetään shakki-kuviota, mikä on esitetty kuvassa 94. Shakkikuviosta etsitään ruudukon kulmia. /83/

Ns. Skaalauskerroin saadaan fyysisen mitan ja sen esittämiseen tarvittavien pikseleiden suhteesta. /83/

Mikäli etäisyys kohteesta muuttuu, on kamera kalibroitava uudestaan. /83/



Kuva 94. OpenCV:ssä käytetään shakkikuviota /39/

4.3.3. Esikäsittely

Esikäsittelyllä pyritään tekemään kuvasta mahdollisimman helppolukuinen myöhemmille vaiheille. Esikäsittelyyn kuuluu mm. kohinan poistaminen kuvasta, kuvatyyppimuunnokset, valoisuuden tasaus, suodatukset, kynnystys sekä reunojen etsiminen. /83/, /169/

Esikäsittelyt voidaan jakaa kahteen ryhmään: Kuva- ja taajuustasomenetelmiin. Kuvatasossa muokataan kuvaa ja pikseleitä ja taajuustasomenetelmissä muokkaus tapahtuu kuvan Fourier-muunnokseen. Osa taajuustason suodatuksista voidaan tehdä myös kuvatasolla, mikä on yleensä laskennallisesti kannattavampaa, mutta monimutkaistuessaan taajuustaso on nopeampi. /119/, /169/ /210/

Kuvatasomenetelmiä ovat mm.: /210/

- alipäästösuodattimet (Smoothing)
- ylipäästösuodattimet (Sharpening)
- derivaattasuodattimet.

Kuvatasoa voidaan kutsua myös **spatiaalitasoksi**. /119/

Taajuustason menetelmiä ovat mm.: /210/

- alipäästösuodattimet
- ylipäästösuodattimet
- kaksiulotteinen Fourier-muunnos
- konvoluutioteoreema.

Kuvan Fourier-muunnos kuvaa väri- tai harmaasävyjen muutoksia.

Yleensä häiriöiden poistoon kuvasta käytetään keskiarvoistusta, jossa lasketaan n*m-matriisista keskiarvo ja sijoitetaan se pikseliin. Tällöin esimerkiksi kohinan yksittäiset mustat tai valkeat pikselit saadaan poistettua kuvasta.

4.3.4. Suodattimia

Suodattimilla kuvasta poistetaan häiriöitä ja korostetaan haluttuja ominaisuuksia, kuten reunoja. Suodatetut pikselit voivat saada isoja tai jopa negatiivisia arvoja. Tämä tulee ottaa huomioon joissain kuvankäsittelyohjelmissa. Tässä kappaleessa käydään läpi joitain yleisimpiä suodattimia ja niiden käyttötarkoituksia. /41/, /60/, /64/, /77/, /119/, /169/ /210/

Mediaanisuodatin

Mediaanisuodattimessa lasketaan kernelin mediaani ja sijoitetaan se ankkuripikselin arvoksi. /141/

2	3	4
6	1	5
7	5	5

Kuva 95. 3X3 matriisi

Kuvan 95 pikseleiden arvojen 1, 2, 3, 4, 5, 5, 5, 6, 7 mediaani on 5.

Keskiarvosuodatin

Keskiarvosuodattimessa lasketaan kernelin keskiarvo ja asetetaan se pikselin arvoksi.

Alipäästösuodatin

Kuvan tasoittamiseen (Smoothing) käytetään alipäästösuodattimia. Matalat taajuudet edustavat yleiskontrastia sekä keskimääräistä intensiteettiä, eli taustaa. Kuvassa 96 on esitetty kaksi alipäästösuodatinta. /77/, /141/

1	1	1
1	1	1
1	1	1

1	2	1
2	4	2
1	2	1

Kuva 96. Kaksi erilaista alipäästösuodatinta

Ylipäästösuodatin

Kuvaa terävöitetään (sharpening) käyttämällä ylipäästösuodattimia. Kuvassa korkeat taajuudet edustavat kontrastin muutosta, eli reunoja. Kuvassa 97 on esitetty kaksi ylipäästösuodatinta. /77/, /141/

-1	-1	-1
-1	9	-1
-1	-1	-1

0	-1	0
-1	5	-1
0	-1	0

Kuva 97. Kaksi erilaista ylipäästösuodatinta

4.3.5. Reunanhaku

Reunanhaulla etsitään kohtia, joissa kontrasti muuttuu jyrkästi, toisin sanoen valoisuuden voimakkaita vaihteluita. Tällaisia ominaisuuksia ovat: /66/, /77/, /78/, /83/

- epäjatkuvuus pintojen syvyydessä
- epäjatkuvuus pintojen suunnassa
- epäjatkuvuus pintamateriaalin fyysisissä ominaisuuksissa
- vaihteluja kohteen valaisussa.

Reunanhakumenetelmät ovat herkkiä häiriöille, joten niitä ei pitäisi tehdä enne suodatuksia.

Reunanhaku konvoluution avulla

Konvoluutiolla etsittäessä kernelin kertoimien summa on nolla. Tästä syystä tasaisella, samanvärisellä tai sävyisellä alueella pikseli saa arvokseen nolla, kun taas korkean kontrastin tullessa kohdalle pikseli erottuu selvästi. Tuloksena on teräviä piirteitä mustalla taustalla. Näitä operaatioita käytettäessä ei tiedetä, ollaanko kummalla puolen reunaa.

Kerneleitä voisi kuvata ylipäästösuodattimiksi. Näitä on kuvassa 98.

1	-2	1
-2	4	-2
1	-2	1

-1	-1	-1
-1	8	-1
-1	-1	-1

0	1	0
1	4	1
0	1	0

Kuva 98. Kolme erilaista reunanhakua konvoluutiolla, joista ensimmäistä kutsutaan Laplacianiksi. /141/

Viivantunnistus

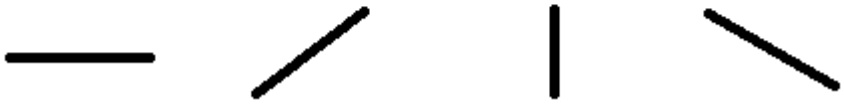
Viivantunnistus vahvistaa vaaka-, pystysuunnassa olevia sekä 45-, -45-asteen kulmassa olevia viivoja. Kuvassa 99 on esitetty viivantunnistuksen periaatetta. mikäli jokin viivoista on olemassa, pikselin arvo vahvistuu.

-1	-1	-1
2	2	2
-1	-1	-1

-1	-1	2
-1	2	-1
2	-1	-1

-1	2	-1
-1	2	-1
-1	2	-1

2	-1	-1
-1	2	-1
-1	-1	2



Kuva 99. Viivantunnistus

Ensimmäisen asteen derivaatta

Derivaatalla kuvataan muutosnopeutta. Ensimmäisen asteen derivaatalla saadaan selvä kulmakerroin, gradientti, joka kertoo, ollaanko tulossa reunalle. Mikäli kulmakerroin on positiivinen, siirrytään tummemmalta alueelta vaaleammalle. Kulmakertoimen ollessa

negatiivinen, tullaan vaaleammalta tummemmalle. Värisävyjen ollessa samanlaiset kulmakerroin on nolla. /210/

Roberts cross

1	0
0	-1

0	1
-1	0

Kuva 100. Roberts cross

Kuvan päälle asetetaan samaan kohtaan kaksi kerneliä, joiden tulojen summat lasketaan yhteen. Kernelit on esitetty kuvassa 100. /78/

Prewitt

Prewittiä käytetään reunojen etsimiseen kuvasta. Kuvan päälle asetetaan samaan kohtaan kaksi kerneliä, joiden tulojen summat lasketaan yhteen. Kernelit on esitetty kuvassa 101 ja sen vaikutus kuvan arvoihin kuvassa 102. /78/, /142/

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

Kuva 101. Unsharp masking

Kuva 102. Vasemmalla Prewittin suodattimella operoitu kuva /142/

Sobel-suodatin

Sobelia käytetään reunojen etsimiseen samaan tapaan kuin Prewittiä. Sobelin kernelit on esitetty kuvassa 103 ja vaikutus kuvaan kuvassa 104. /78/, /143/

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Kuva 103. Sobel

Kuva 104. Oikealla Sobel-suodattimella operoitu kuva /143/

Canny

Yhdistämällä reunanhaku kynnystyksen kanssa saadaan aikaan uusia reunanhakumenetelmiä. Canny-reunanhakualgoritmilla etsitään reunat kuvasta Sobel-suodattimella, jonka jälkeen kuva kynnystetään kahdella eri arvolla: korkealla ja matalalla. Tätä kynnystystä kutsutaan hystereesi kynnystykseksi (hysteresis thresholding). Canny-algoritmi vertaa näitä kahta kynnystettyä kuvaa toisiinsa ja rakentaa niistä yhden yhtenäisen reunakuvan korkeamman suodatinkuvan päälle. /72/, /78, s. 167/

Kynnystys

Suodatuksien jälkeen kuva kynnystetään jollain raja-arvolla (threshold) binäärikuvaksi, jolloin se on helpompi jakaa alueisiin. Kynnystyksestä esimerkkejä löytyy liitteistä. (Liite 3) /78/, /83/, /169/

Toisen asteen derivaatta

Mikäli halutaan tietää, kummalla puolella reunaa reunapikseli sijaitsee, joudutaan käyttämään toisen asteen derivaattaa, jossa kulmakertoimen etumerkki kertoo, kummalla puolen reunaa ollaan. Toisen asteen nollakohdat kertovat reunan tarkan paikan. /210/

4.3.6. Segmentointi

Segmentoinnin tarkoituksena on jakaa kuva mielenkiintoisiin alueisiin, mikä helpottaa piirteiden sekä ominaisuuksien etsimistä. Esikäsittelyn kynnystyksen jälkeen jäljelle jääneet alueet jaetaan ja nimetään tai numeroidaan (labelointi). Mikäli binäärikuvassa valkoiset pikselit ovat toisensa naapureita, ne kuuluvat samaan alueeseen. /41/, /66/, /72/, /78/, /83/, /169/

4.3.7. Piirteiden ja ominaisuuksien tunnistaminen

Esikäsittely sekä segmentointi yleensä ovat samanlaisia kuvankäsittelystä toiseen, mutta piirteiden ja ominaisuuksien tunnistaminen on aina tapauskohtaista. Näitä voivat olla esimerkiksi: /18/, /66/, /78/, /83/, /169/

- pinta-alan laskenta
- reunojen tai viivojen pituudet
- muoto.

Kuvasta voidaan esimerkiksi poistaa kaikki pinta-alaltaan (liian vähän pikseleitä kappalemäärältään sisältäviä) liian pienet nimetyt alueet tai ympyröitä etsiessä liian soikeat tai suorakulmamaiset alueet. /83/

4.3.8. Tulosten käsittely

Kun piirteet ja ominaisuudet on tunnistettu, tulokset käsitellään. Tulosten käsittelyn voi oikeastaan jakaa kahteen osaan: Konenäköjärjestelmän tarvitsemiin tuloksiin sekä ihmisten tarvitsemiin tuloksiin. /18/ ,/83/, /169/

Tulosten käsittelyä voi olla esimerkiksi välimatkojen mittaaminen tai kappaleiden rotaatioiden laskeminen, jonka seurauksena konenäköjärjestelmä tekee sille ennalta määrättyjä tehtäviä, kuten poistaa kappaleen liukuhihnalta tai liikuttaa robottikättä tiettyyn paikkaan. /18/, /83/, /169/

Ihmiselle tuloksia voidaan käsitellä esimerkiksi korostamalla joitain alueita väreillä tai ilmoittamalla välimatkoja, virheitä tai histogrammeja analysointia varten. /18/, /83/, /169/

5. LOPPUTULOS

Tehtävänä oli luoda apuväline BGA-komponenttien asennukseen sulautettujen järjestelmien laboratorion FP-600 ladontakoneeseen. Apuvälineen piti olla kustannustehokas, koska BGA-komponenttien asennusta ei tulla suorittamaan koulun tiloissa kovin usein. Laitteen tulisi olla pieni, eikä se saisi aiheuttaa haittoja ladontakoneen normaaliin käyttöön.

5.1. Esiselvitys

Esiselvityksen tavoitteena oli saada projektisuunnitelma, johon oli määritelty työn tavoitteet, rajaukset sekä tehtävät.

5.1.1. Asennustekniikat

Esiselvitys aloitettiin tutkimalla valmiita BGA-komponentin asennukseen käytettäviä laitteita ja tekniikoita. Esiselvityksessä löydettiin sopivia tekniikoita niin Rework-asemista, prototyyppi- sekä pientuotantolaitteista ja CNC-koneista. Löydettiin kolme erilaista tapaa asentaa BGA-komponentti.

1) Komponentti ja piirilevy on aina samassa kohdassa. Robottikäsi poimii komponentin syöttäjästä ja asentaa sen liukuhihnalla olevalle piirilevylle. Tätä tekniikkaa käytetään suurilla tuotantomäärillä.

2) Konenäkö laskee komponentin ja piirilevyn asennusmerkeistä oikean asennuskohdan. Tätä tekniikkaa käytetään niin automaattisissa ladontakoneissa kuin myös manuaalisissa. Automaattisissa ladontakoneissa robottikäsi asentaa komponentin piirilevylle. Manuaalisissa koneissa käyttäjä vie komponentin oikeaan kohtaan, jolloin käsi lukkiutuu.
3) Kamerat kuvaavat piirilevyä ja komponentin pohjaa. Yleensä kuvat esitetään päällekkäin, jolloin nähdään asennusvirhe. Piirilevyä tai komponenttia liikuttelemalla haetaan tarkka

asennuskohta, jonka jälkeen komponentti asennetaan piirilevylle. Tätä käytetään yleisesti rework-asemissa ja pientuotannossa.

Esitutkimuksen perusteella toinen ja kolmas tapa soveltuivat parhaiten meidän vaatimuksiin.

5.1.2. BGA

Esiselvityksessä selvitettiin, minkä kokoisia BGA-komponentteja on sekä minkälaisia vaatimuksia niiden asennuksessa on otettava huomioon. Komponenttien koot vaihtelivat valmistajien mukaan erittäin pienistä aina 60 - 80 mm suuruisiin asti. Dima FP-600 koneen spesifikaatioissa komponenteille määriteltiin vain maksimikorkeus, joka on poimintapään 0-tasosta 2 cm ylöspäin ja 2,5 cm alaspäin. Rajattiin suurimmaksi käsiteltäväksi komponentiksi 60 mm x 60 mm ja piirilevyn kooksi 210 mm x 210 mm. Suuret komponentit tulee poimia keskikohdasta, jotta komponentti pysyy vaakatasossa.

BGA-komponentin asennustarkkuudeksi annettiin 0,5 mm, koska tämän tarkempaa asennusta ei tulla suorittamaan.

5.2. Järjestelmän suunnittelu

Järjestelmän suunnittelussa lähdettiin liikkeelle erilaisten mahdollisuuksien luonnostelulla. Mietittiin, miten kättä ja komponenttia liikutettaisiin, missä olisivat kameroiden paikat, komponentin ja piirilevyn sijainti. Tietyt etäisyydet kuvien välillä ovat vakiot, mikä helpottaa laskentaa sekä suunnittelua.

5.2.1. Moottoriton versio

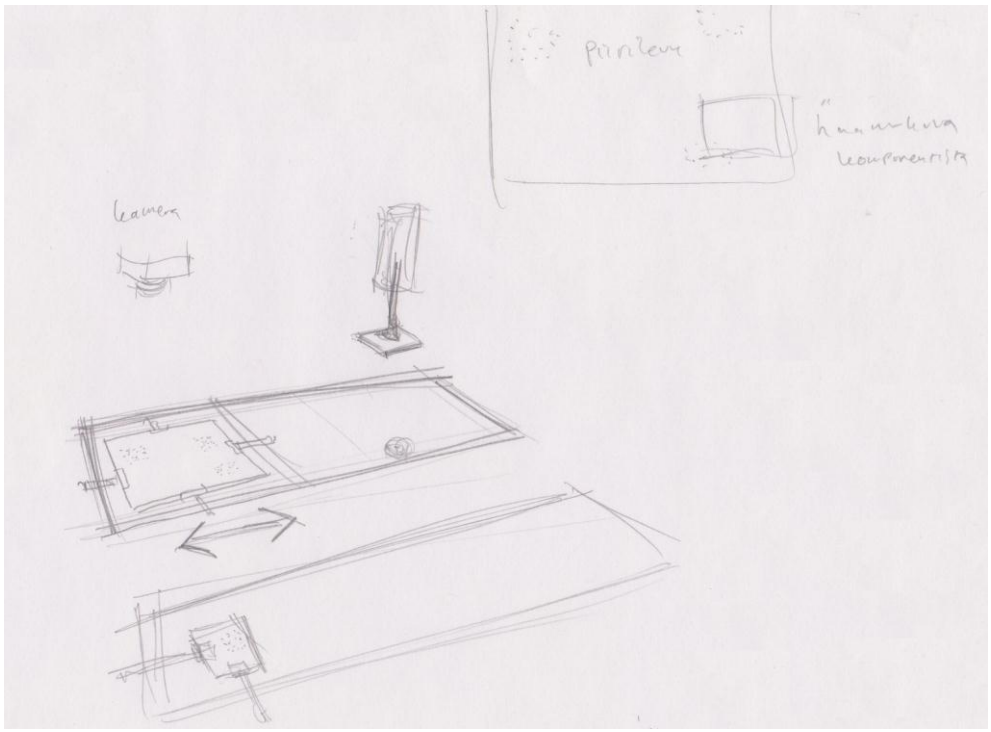
Ensimmäisessä versiossa ei käytetty moottoreita ollenkaan. Piirilevy oli kiinnitetty kelkkaan, jossa piirilevyn paikkaa säädettiin käsin mikrometriruuveilla. Kamerat sijaitsivat piirilevyn

päällä kuvaten asennuskohtaa ja toinen komponentin alla kuvaten komponentin alapuolta. Kättä ohjattiin manuaalisesti toisen kameran päällä. Kun X- tai Y-akseli olivat oikealla kohdalla, ne lukkiutuvat. Kun molemmat lukkiutuvat, tehtiin tarkennukset mikrometriruuveilla sekä pyörittämällä komponenttia oikeaan asentoon ja kelkka työnnettiin komponentin alle. Tämän jälkeen komponentti asennettiin paikoilleen.

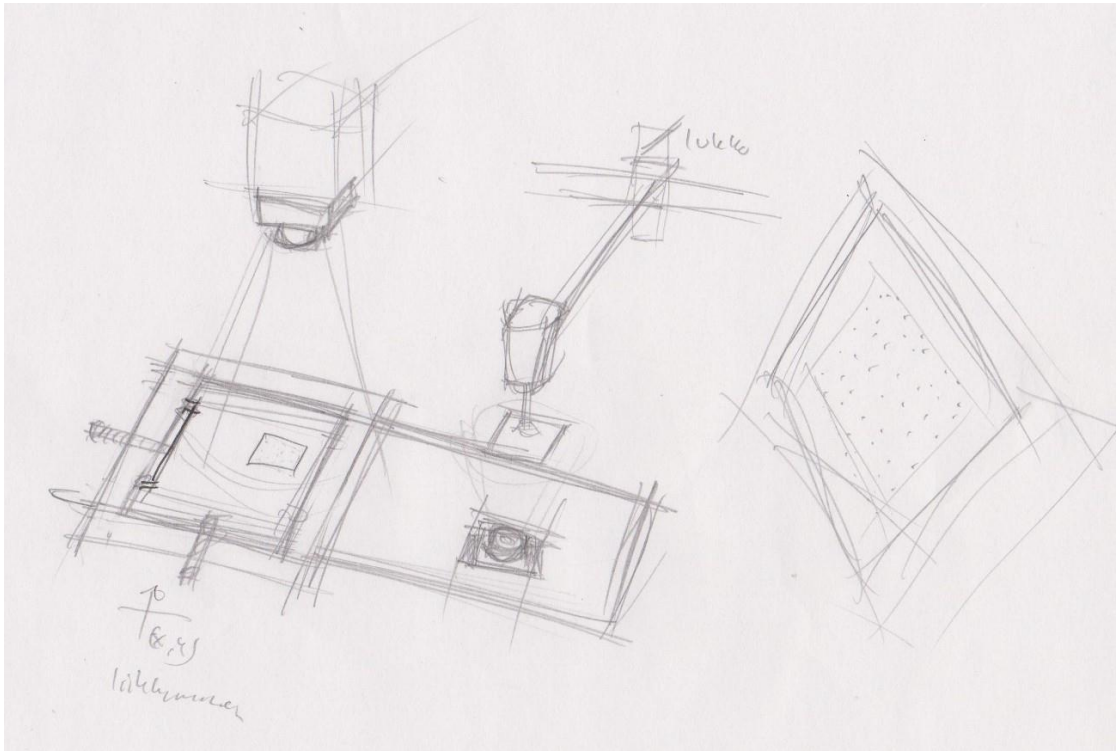
Komponentin oikea paikka saatiin kuvaamalla asennuskohtaa sekä komponentin alapuolta. Kun kuvat näytettäisiin päällekkäin, nähtäisiin mihin suuntaan komponenttia tulisi siirtää. Tämä edellytti, että etäisyydet kuvissa ja “oikeassa maailmassa” tiedettäisiin, sekä kameroiden kuvien etäisyydet toisistaan. Kuvat 105, 106 ja 107 ovat ensimmäisiä luonnoksia siitä, miten laite voisi toimia.

Moottorittomaan versioon oli useita erilaisia ideoita:

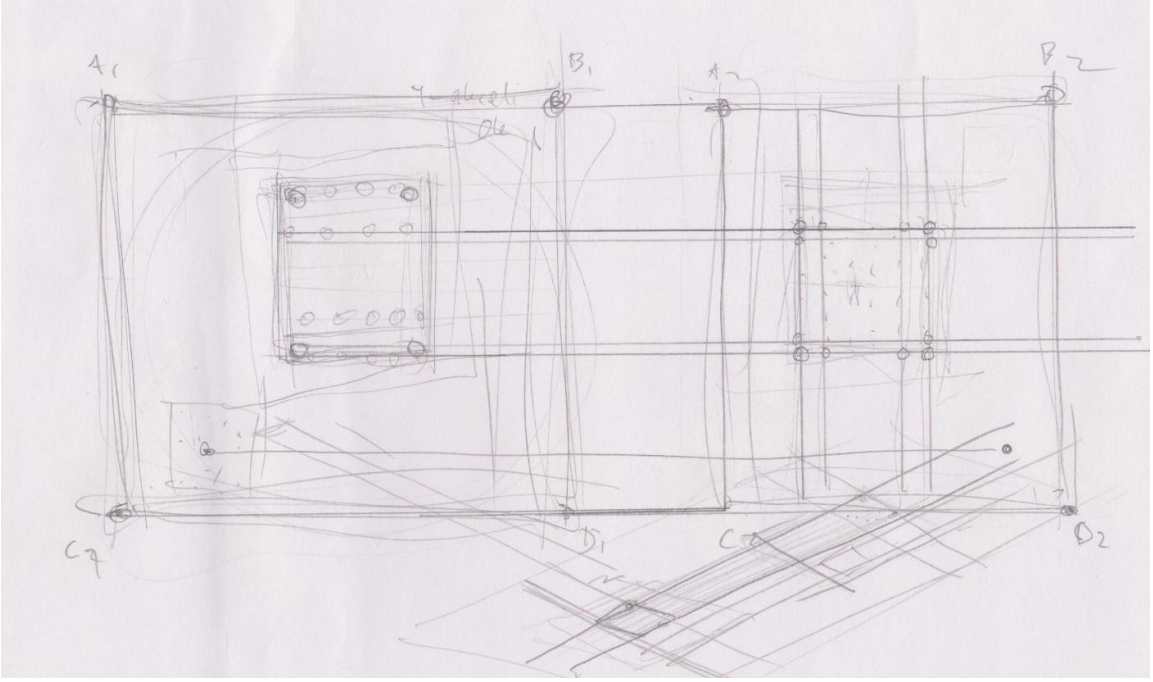
- Ledivalo syttyy, kun oikea kohta on lähellä ja käsi lukkiutunut.
- Ohjelma ohjaisi näytöllä käyttäjää oikeaan kohtaan ja käsi lukkiutuisi ollessa tarpeeksi lähellä.
- CAD-tiedot näkyisivät näytöllä, kuvat mahdollisesti verrattavissa keskenään.
- Kelkka poistettavissa normaalin toiminnan tieltä.
- Etäisyyksien mittaus kuvista.



Kuva 105. Luonnos moottorittomasta versiosta



Kuva 106. Luonnos moottorittomasta versiosta, näytöllä kaksi kuvaa päällekkäin



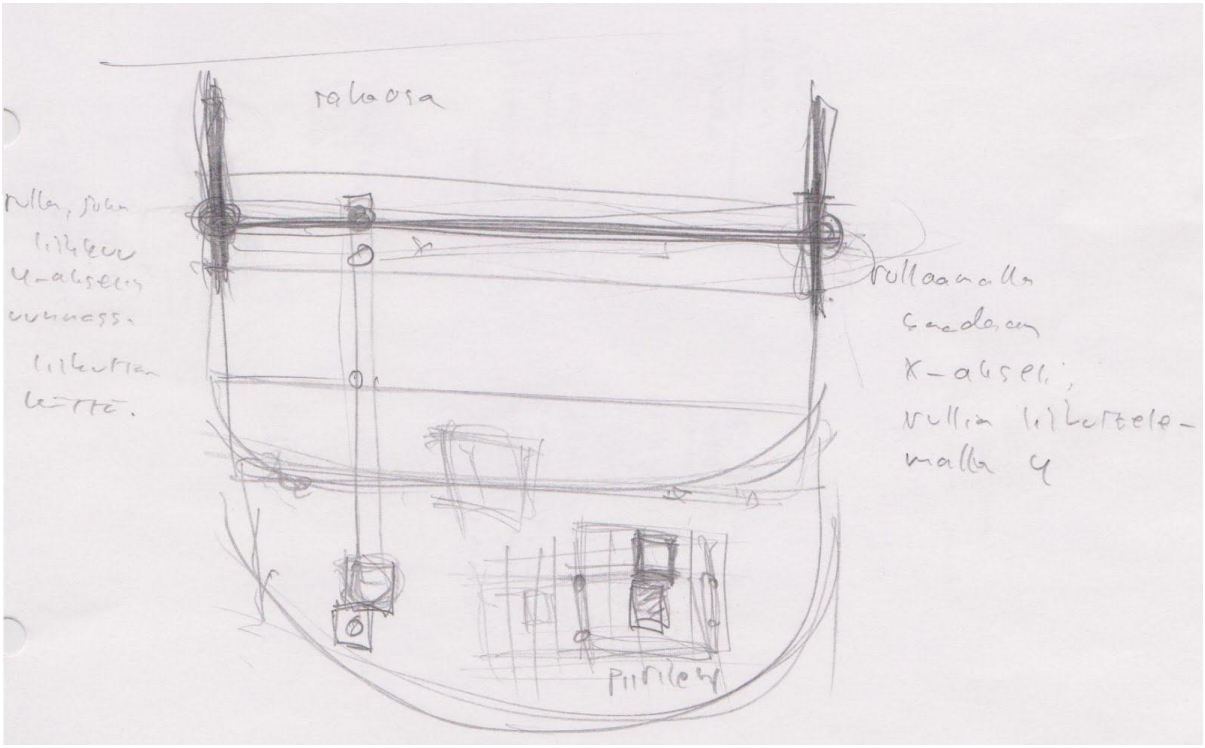
Kuva 107. Etäisyyksien tunteminen sekä idea moottorittoman version toiminnasta

Päädyimme hylkäämään moottorittoman version:

- Kameran ja valojen saaminen komponentin alle olisi vaikeaa.
- Vaatisi fyysisiä muutoksia ladontakoneeseen.
- Toimii vain tietyn muotoisilla piirilevyillä.
- Lukitusmekanismin olisi haastava toteuttaa varmasti että luotettavasti.
- Kiskot tehtävä tilaustyönä, mistä syntyy suuria kustannuksia.

5.2.2. Moottorit kiskoilla

Seuraavassa versiossa kättä liikuteltaisiin moottoreilla ja ei tarvita lukkoja eikä kelkkaa. Moottorit olisi sijoitettu johteiden päälle ladontakoneen taakse, missä on runsaasti tilaa, joten rakenne olisi pois normaalin käytön tieltä. Moottoreiden pitomomentti pitää käden paikoillaan.



Kuva 108. Moottorit kiskoilla

Tässä versiossa kättä ohjattaisiin X- ja Y-suunnissa moottoreilla. Kaksi toistensa kanssa synkronoitua moottoria ohjaavat kättä Y-suunnassa kiskoilla ja yksi ohjaa kättä X-akselin suunnassa. Kiskot on kiinnitetty pöydän rakenteisiin. Moottoreiden ohjaukseen käytetään moottoriohjainta. Kuvassa 108 on käsin piirretty versio laitteesta.

Komponentti on sijoitettu reunalle, missä sitä kuvataan peilin kautta alapuolelta. Piirilevyä kuvataan päältä ja pysyy paikallaan koko toimenpiteen ajan. Tällöin ei tarvitse tehdä laitteeseen fyysisiä muutoksia.

Komponentille ja piirilevylle on ennalta määrätyt kuva-alueen koot, jotka asetimme alussa 6 cm korkuisiksi. Kuvat otetaan, tehdään tarvittavat kuvankäsittelyt, mistä saadaan selville tarvittavat tunnistukset sekä mitat. Tämän jälkeen moottoriohjain liikuttaa moottoreilla komponenttia kameran päällä laskettujen matkojen verran. Komponenttia joudutaan pyörittämään käsin oikeaan rotaatioon. Tätä jatketaan kunnes päällekkäin asetettujen kuvien mukaan komponentti sekä asennuskohta piirilevyllä ovat päällekkäin. Tämän jälkeen moottorit liikuttavat komponenttia vakio matkan asennuskohdan päälle ja käyttäjä asentaa komponentin paikoilleen.

Jälkeenpäin voidaan ottaa kuva asennuksesta ja laskea, onko komponentti varmasti oikealla kohdalla.

Hyviä ominaisuuksia

- Mahdollisimman vähän normaalin toiminnan tiellä.
- Vain käsi kiinni rakenteissa, voidaan irrottaa tarvittaessa.
- Mahdollista asentaa muitakin kuin BGA-komponentteja.
- Mahdollista toteuttaa todella tarkaksi ilman kameroitakin.

Huonoja ominaisuuksia

- Tarvitsee kolme moottoria.
- Y-akselin suunnassa olevien moottoreiden on oltava synkronoitu keskenään. Synkronoinnista haaste.
- Alumiiniprofiilit ja moottorit kulkivat johteilla. Tämä lisää hintaa.
- Komponentin paino laskee poimintapäätä. Kun kalibrointi tehdään tietylle etäisyydelle, syntyy virhettä komponentin tullessa lähemmäs kameraa.
- Piirilevyn päällä oleva kamera saattaa tarvita ylimääräistä optiikkaa, jottei se jää käden tielle.
- Mahdolliset johteet lisäävät kustannuksia

- Moottorit kalliita, monenlaisia tarkkuudeltaan sekä käytöltään

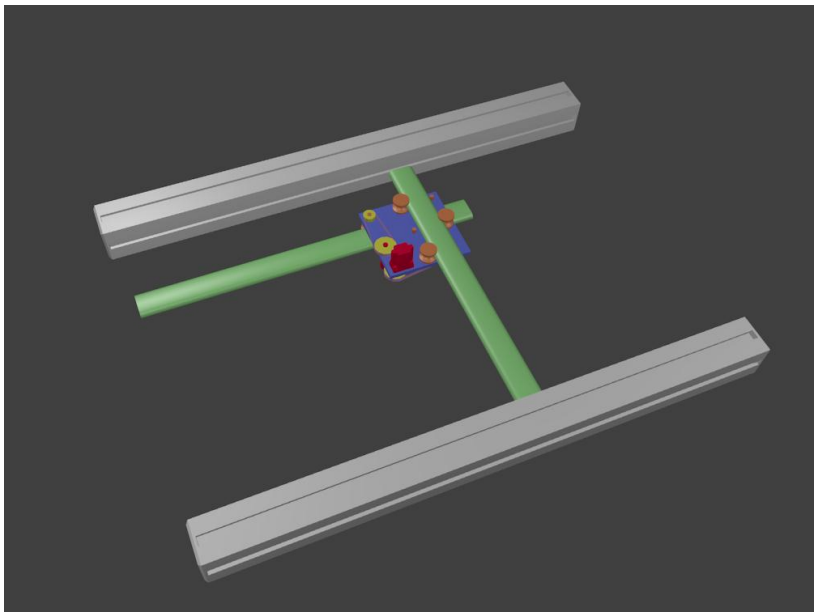
Eri variaatioita

- X-suunnassa kättä liikuttava moottori kiinni kädessä, liikuttelu tapahtuu hihnan avulla, joka on kiinni Y-akselin moottoreissa. Tällöin ei välttämättä tarvita kiskoja tai kelkkaa Y-akselille.
- Kuularuuvein toteutettu liikuttelu. /144/

5.2.3. Moottoroitu varsi

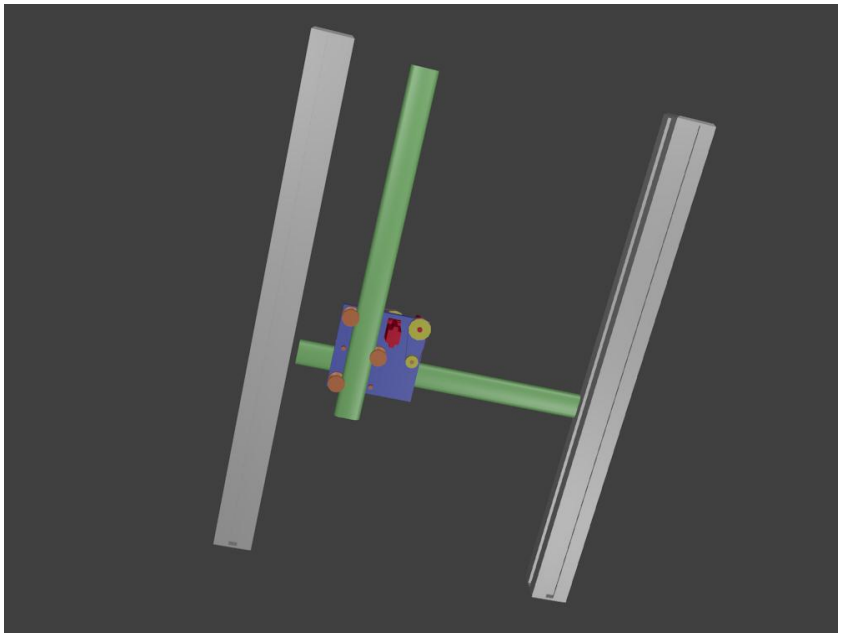
Viimeisessä versiossa käytetään kahta moottoria, jotka ovat kiinnitettyinä samaan kelkkaan. Kelkka on kiinnitetty X-akselin suuntaiseen kiskoon, missä moottori liikuttaa kättä tässä suunnassa. Toinen moottori on kiinnitetty käteen, mikä liikuttaa samalla kättä Y-akselin suunnassa. Systeemi on sijoitettu ladontakoneen taakse, jolloin se ei ole normaalin toiminnan tiellä.

Myös kuvaustapaa ja valaistusta muutettiin. Kuvaus tapahtuu samoissa kohdissa, mutta komponenttia kuvattaisiin yläpuolelta. Komponentin alle tulisi valo, joka diffuusoitaisiin levynläpi (backlight). Komponentti olisi kuvaushetkellä levyn päällä, jolloin sen reunat tulevat tarkasti näkyviin. Komponentin siirtäminen oikealle paikalle tapahtuu muuten samaan tapaan kuin edellisessä versiossa.

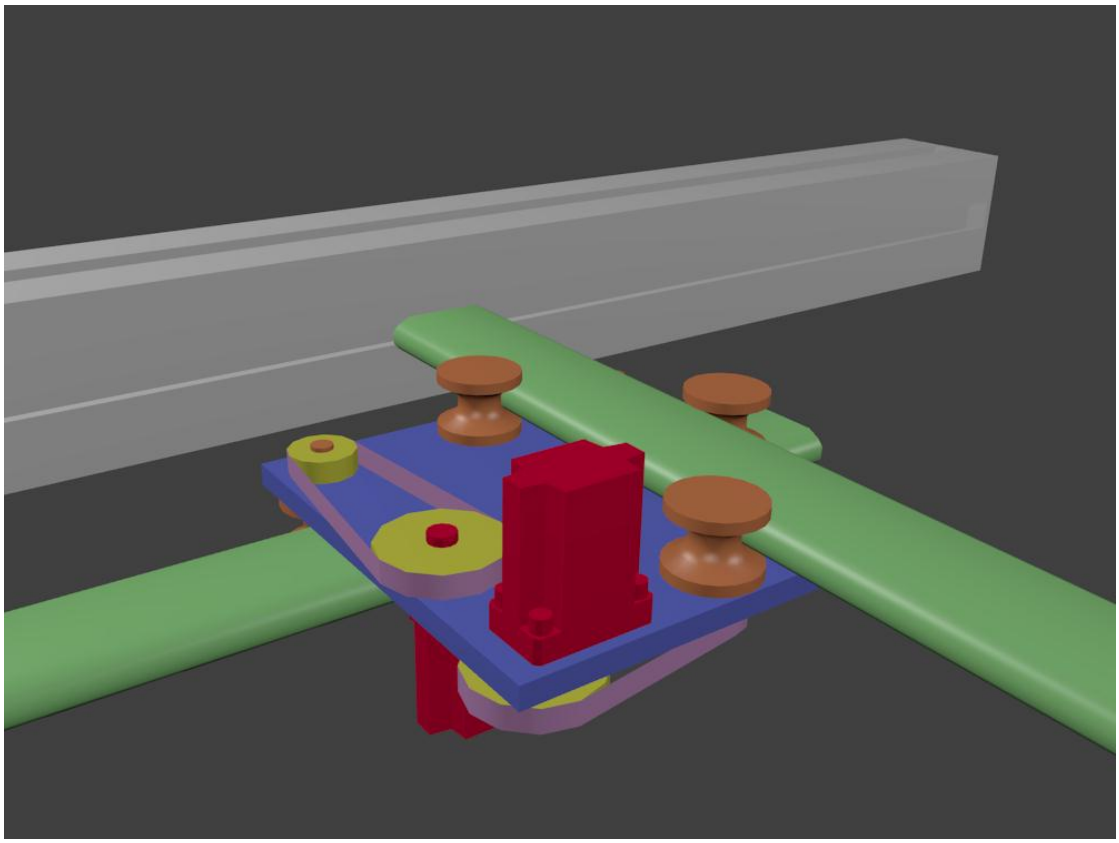


Kuva 109. Moottorit varressa, ylhäältä

Kuvasta 109 ja 110 nähdään, miten kisko on kiinnitetty kahteen alumiiniprofiiliin ja käsi on näiden keskellä samansuuntaisesti.



Kuva 110. Moottorit varressa, alhaalta

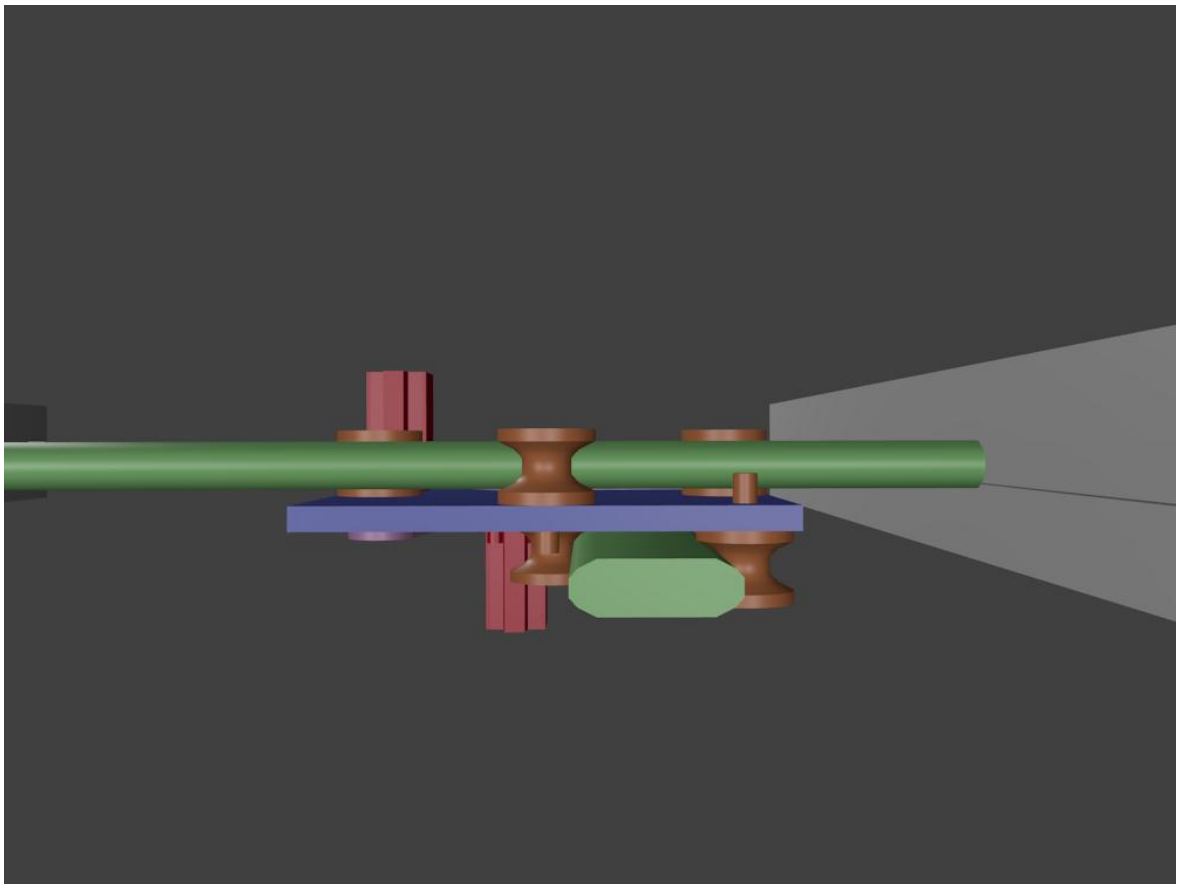


Kuva 111. Kelkka läheltä

Kuvasta 111 nähdään, kuinka moottorit on sijoitettu kelkkaan. Moottorit pyörittävät hihnoilla pyöriä, joilla kättä liikutellaan. Kuvassa 112 on näytetty laite takaapäin.

Kuvissa näkyy laitteen rakenne, missä:

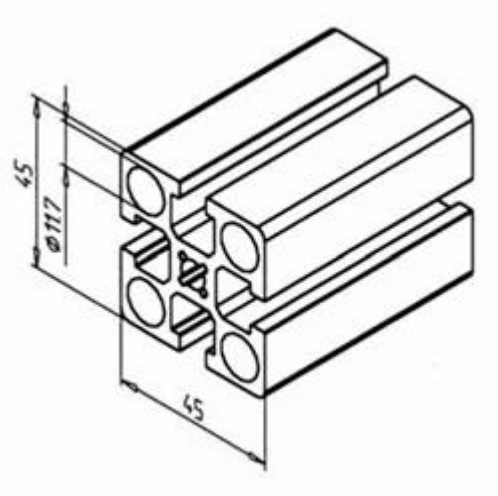
- Alempi vihreä osa on ladontakoneessa oleva varsi, ylempi on lisäosa, joka on kiinni laidoissa oleviin elementteihin.
- Punaiset ovat moottoreita.
- Laidan elementit ovat standardin mukaiset 45x45 profiilit Moveteciltä, mistä pöydänkin elementit on tilattu.



Kuva 112. Kuva takaa

Ladontakoneen pöytä on rakennettu Movetecin alumiiniprofiileista, joten järkevintä on käyttää samoja profiileja apulaitteen rakenteeseen, jotta palaset saadaan yhdistettyä toisiinsa ilman mitään ongelmia. Alumiiniprofiilit ovat kooltaan 45 x 45 mm, jonka malli nähdään kuvasta 113.

Laitteen mekaanista rakennetta miettiessä, tuli ottaa huomioon työaseman fyysiset rajat. Laite ei saisi häiritä normaalia työskentelyä ladontakoneen ääressä ja se pitäisi olla suhteellisen helppo asentaa ja irrottaa. Tästä syystä päätimme, että laite tulee ladontakoneen taakse, koska se on oikeastaan ainoa paikka pöydällä, missä on tilaa. Itse laite ei ole kovin iso, mutta tarvitsee silti sen verran tilaa, ettei sitä voi laittaa etupuolelle. Suurimman osan tilasta vievät putket mihin laite laitetaan kiinni.



Kuva 113. 45 x 45 mm alumiiniprofiili /98/

Hyvät ominaisuudet

- poissa tieltä
- vähän mekaanisia osia
- pieni koko
- edullinen.

Huonot ominaisuudet

- Kahdella pienellä moottorilla liike voi olla suhteellisen hidasta.

Erilaisia variaatioita

- Kamera kiinni poimintapäähän. Otetaan kuva, tehdään siirrot ja otetaan uusi kuva.
- Moottorit suoraan kiinni pyöriin ilman hihnaa.

5.3. Moottorit

Kuva 114. GWS S35 STD Continuous Rotation Servo /211/

Kuvassa 114 on käyttämämme servomoottori. Pieni kokoinen servo, joka pyörii 360 astetta eli siinä ei ole potentiometriä. Normaalissa servossa pulssitetaan minne halutaan liikkua, tässä ne pulssit muutetaan jatkuvaksi liikkeeksi. Sitä voidaan ohjata suoraan servomoottoriohjaimella, joka kiinnitetään tietokoneeseen. Servon ominaisuudet:

- nopeus 6 V:lla 0,14 s/60°
- pitomomentti 2,8 kg/cm
- nopeus 4,8 V:lla 0,16 s/60°
- johtimen pituus 11” = ~28cm.

Tämä moottori valittiin sen takia, koska se oli halpa ja niitä sai tilattua Suomesta. Tämä oli myös yhteensopiva Micro Maestro servo-ohjaimen kanssa, joten sekin oli yksi tärkeimmistä ominaisuuksista. /132/

Työssä haluttiin myös käyttää servomoottoreita, niiden hyvän tarkkuuden ja helppokäyttöisyyden takia. Työn alussa tutkimme eri vaihtoehtoja moottoreiksi, ennen kuin päädyimme servomoottoreihin. Askelmoottorit olivat myös varteenotettava vaihtoehto, mutta koska ne eivät ole niin yksinkertaisia käyttää ja ne voivat myös maksaa paljon enemmän kuin servomoottorit niin päädyimme käyttämään servomoottoreita alkuun.

Työn edistyessä huomattiin, että meidän valitsemat servomoottorit eivät sopineet parhaalla mahdollisella tavalla tähän työhön. Servomoottorimme ollessa jatkuvasti pyörivä, niin sen paikan tunnistus on hankalampaa kuin normaaleissa servomoottoreissa, mutta ajattelimme ettei tästä tulisi niin suurta ongelmaa. Ongelma olikin suurempi kuin odotimmekaan, koska meillä ei ole mitään ulkoisia antureita millä paikan saisi luettua. Normaalilla servomoottorilla saisimme paikan tietoon, mutta siihen tulisi lisätä jonkinlainen vaihteistus, jotta liikerata riittää kattamaan koko pöydän alan.

Moottoreita tutkiessa tuli ottaa huomioon hinta, tarkkuus ja ohjauksen helppous. Aluksi tarkasteltiin lineaariaktuaattoreita, koska niitä voidaan ohjata erittäin tarkasti ja niissä olisi koko systeemi yhdessä paketissa. Näiden kallis hinta pakotti tutkimaan muita vaihtoehtoja, koska 30 cm lineaarivaihteet saattoivat maksaa satoja euroja.

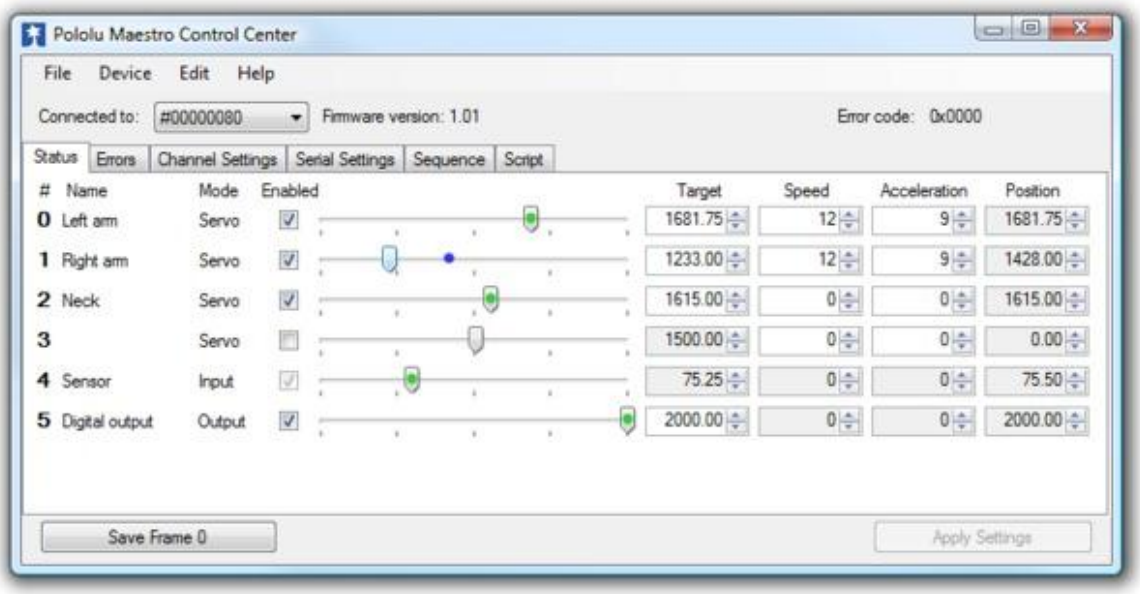
Askelmoottoreita tutkittiin myös kauan, mutta servomoottorit tuntuivat kokemattomille paremmalta vaihtoehdolta. Askelmoottoreillakin tämä työ voisi olla toteuttamiskelpoinen, mutta koska ne maksoivat huomattavasti enemmän kuin servomoottorit, niin emme tutkineet niitä hirveän tarkasti.

5.4. Moottoreiden ohjaus

Moottoreiden ohjaus tapahtuu Micro Maestro servo-ohjaimella. Ohjaimeen voidaan liittää kuusi servoa yhtä aikaa ja niitä voidaan ohjata Pololun omalla ohjelmalla Maestro Control Centerillä. Ohjelma on suhteellisen yksinkertainen käyttää. /132/

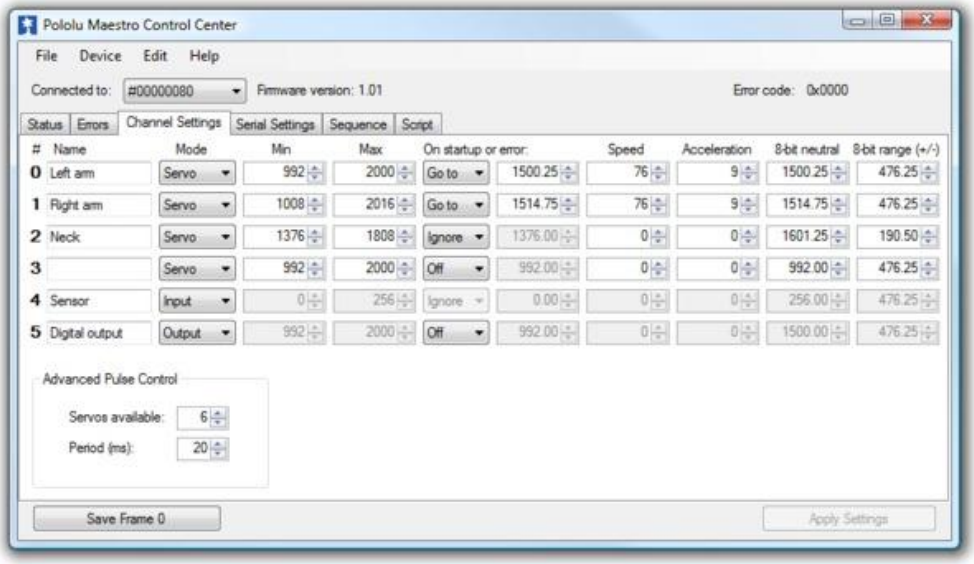
Micro Maestroa ohjataan PC:llä USB-väylän kautta. PC:llä on oma ohjelma servo-ohjaimelle, Maestro Control Center, jolla ohjataan ohjaimeen kiinnitettyjä servoja. Ohjaimella voidaan säätää servomoottorille “target” paikka eli se paikka mihin servomoottori pyörii. Säädin on millisekunteina, keskikohta on 1 ms, maksimi 1,5 ms ja minimi 0,5 ms. Maestro Control Centeristä löytyy myös säädöt servon pyörimisnopeudelle ja kiihtyvyydelle. Pyörimisnopeudet

ja kiihtyvyydet on määritelty jokaiselle servomoottorille erikseen, joten moottoreiden speksejä pitää tarkastella, jos haluaa jotain tiettyjä nopeuksia.



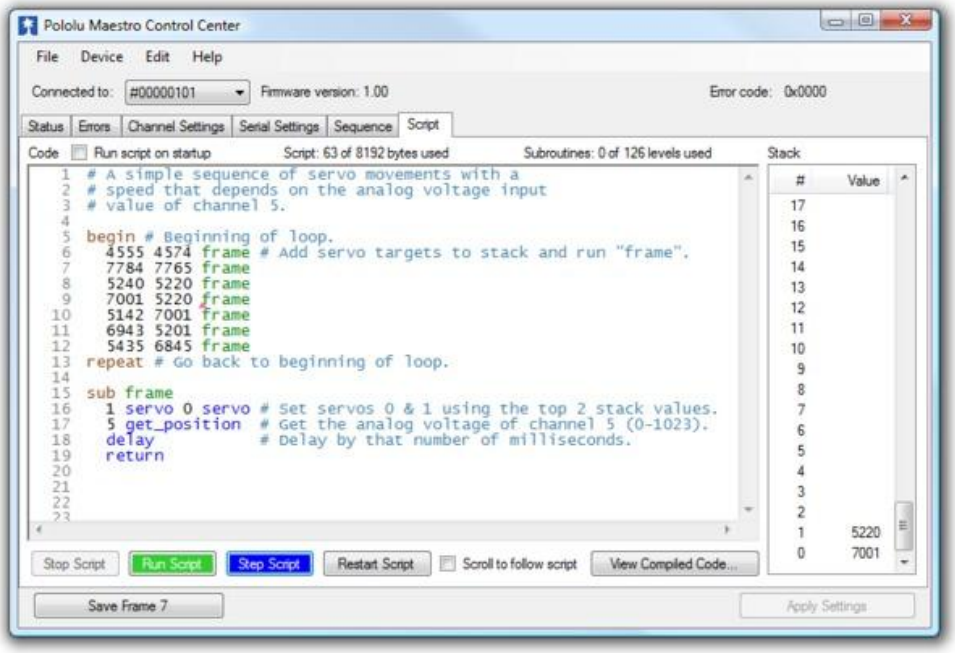
Kuva 115. Maestro Control Center Status /131/

Kuvassa 115 näkyy tilanne kun servo-ohjaimeen on kiinnitetty 3 servomoottoria ja miten niitä voidaan ohjata. Position numero kertoo missä asennossa servomoottori on, targetilla kerrotaan mihin asentoon se halutaan, speedillä määritellään maksimipyörimisvauhti ja accelerationilla moottorin kiihtyvyys. Vihreä täplä kertoo, että servo on tehnyt sen mitä sen on käsketty tehdä ja sininen kertoo, että se on vielä matkalla haluttuun asentoon. Jos käyttää jatkuvasti pyöriviä servomoottoreita, niin targetilla ohjataan nopeutta eikä paikkaa.



Kuva 116. Control Center Channel Settings /131/

Kuvassa 116 nähdään Channel Settings välisivu, jossa määritellään mitä halutaan tehdä heti, kun laite saa virrat. Sillä voidaan määritellä jokaiselle servomoottorille omat aloituspaikkansa. Servomoottoreille voidaan myös määritellä paikkansa, jos jostain syystä sattuu jokin virhe esim. sähkökatkos.

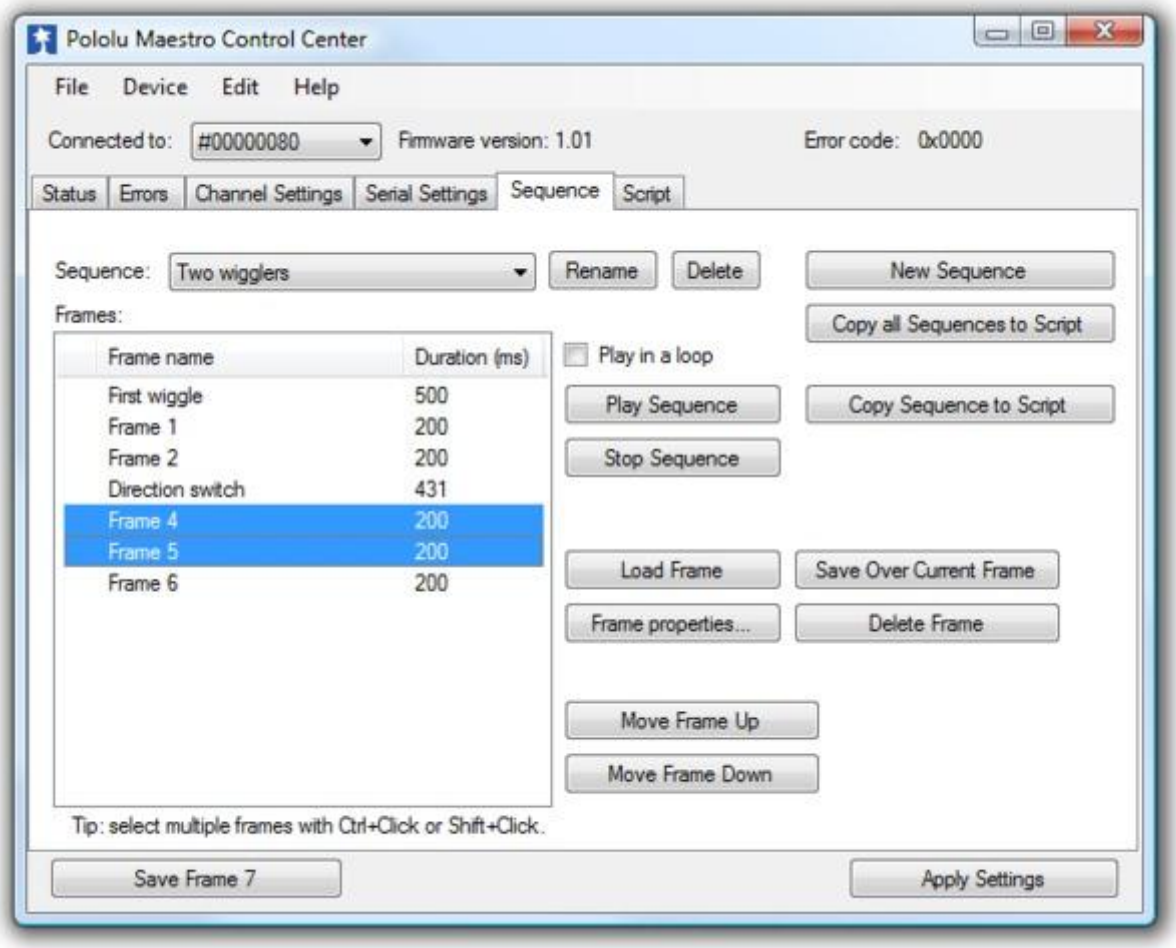


Kuva 117. Maestro Control Center /131/

Maestro Control Centerissä on myös mahdollisuus kirjoittaa omaa koodia, jolla ohjataan servomoottoreita. Kuvassa 117 näkyy esimerkkikoodi.

```
begin
        4000 0 servo    #aseta servo channel 0:ssa 1 ms asentoon
        500 delay       #odota 500 ms
        5000 0 servo    #aseta servo 1,25 ms asentoon
        500 delay       #odota 500 ms
        6000 0 servo    #1,5 ms asento
        500 delay
        7000 0 servo    #1,75 ms asento
        500 delay
        8000 0 servo    #2 ms asento
        500 delay
repeat          #aloita koodi alusta
```

Esimerkki koodista millä ajetaan servomoottori minimiasennosta maksimiasentoon ja toistetaan tätä. Ohjelmalla voidaan tehdä myös paljon monimutkaisempia scriptejäkin. Servo-ohjaimesta riippuen scriptien koot voivat olla 1 - 8 kb kokoisia suurimmillaan. Micro Maestrolla suurimmat scriptit voivat olla 1 kb ja muilla ohjaimilla 8 kb.



Kuva 118. Maestro Control Center /131/

Jos ei halua käsin kirjoitella jokaista vaihetta, niin ne voidaan tehdä myös Sequencerin avulla. Sequencerilla tehdään lista moottoreiden toiminnoista, jotka halutaan tehdä tietyssä järjestyksessä. Listaan tehdään jokaiselle vaiheelle oma Frame, jossa tapahtuu halutut toiminnot. Jokaiselle Framelle määritellään oma kesto, kuinka kauan siinä vaiheessa pysytään ennen kuin siirrytään seuraavaan vaiheeseen. Tätä kautta voidaan tehdä helposti liikeketjuja

moottoreille. Jokainen vaihe voidaan keskeyttää kesken ajon ja niiden järjestystä voidaan vaihdella suoraan lennosta. Kuvassa 118 nähdään miten tämä toimii käytännössä.

5.5. Kamerat

Työn aikana tutkittiin useita kameroita sekä vertaillessa niiden hinnan suhdetta laatuun. Työhän kameroiden valinta aloitettiin etsimällä ja vertailemalla konenäössä käytettäviä kameroita, sillä ei vielä tiedetty millaista kameraa työhön tarvittaisiin. Suunnittelun edetessä määritettiin kuvausalueiden fyysisiksi korkeuksiksi kuusi senttimetriä. Tarkkuudeksi laitteelle oli asetettu 0,5 mm, jolloin näiden perusteella pystyttiin selvittämään, kuinka monta pikseliä 0,5 mm esittämiseen vähintään tarvittiin kamerakohtaisesti. Esimerkiksi 640x480 resoluutiolla yksi pikseli on 60 mm / 480 pix = 0,125 mm / pix, jolloin 0,5 mm alueelle mahtuu 4 pikseliä.

5.5.1. FireWire kamerat

FireWire-kameroita löydettiin muutamaa web-kameraa lukuun ottamatta vain konenäköön tarkoitettuja kameroita sekä mikroskooppeja. Heti työn alussa saatiin Kemi-Tornion Ammattikorkeakoulun Tekniikan yksikön TKI:n Optisen mittaustekniikan laboratoriosta lainaan FireWire-kameran (Foculus-FO234SB/SC), FireWire-kortin sekä zoom-objektiivin, minkä sopivuutta testattiin työhön. Kyseistä kameraa olisi voitu lainata aika-ajoin laitteen toimintaan, mutta tämä koettiin hankalaksi. Kameraa käytettiin kuitenkin kuvankäsittelyn harjoitteluun, sillä se oli tarkempi kuin web-kamerat sekä saatava videokuva on valmiiksi harmaasävyinen.

FireWire-kortin asentamisen jälkeen OpenCV:ssä ei voitu käyttää web-kameroita. Tästä asiasta on lisää OpenCV:n ongelmissa.

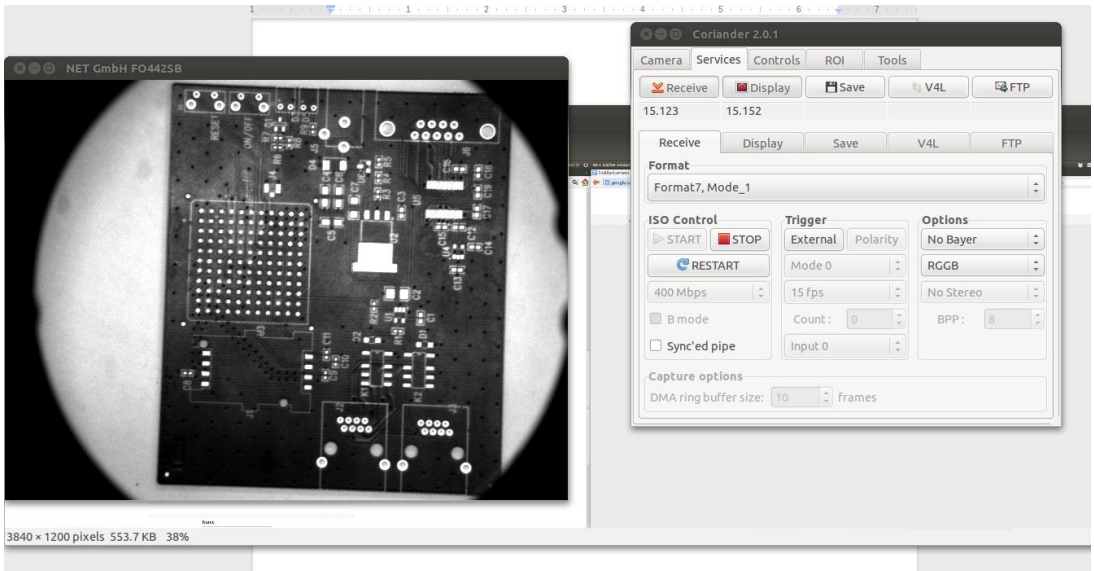
FireWire-kamerat hylättiin, sillä niitä valmistettiin lähinnä teollisuuskäyttöön, mikä nostaa huomattavasti niiden hintaa. Esimerkiksi lainaamamme kamera oli vanhaa mallia, mutta olisi vielä maksanut useita tuhansia euroja.

Coriander

FireWire-kameroissa on todella hyvä tuki videoformaatin siirtämiseen, lisäksi ne toimivat Linux-käyttöjärjestelmissä ongelmitta. Lainassa olleelle kameralle löydettiin Linuxille katseluohjelma “coriander”. Ohjelmalla voi muuttaa kameran tuottaman kuvan ominaisuuksia kamerasta riippuen. Kuva 119 on kuvankaappaus ohjelman käytöstä.

Linuxissa asennat corianderin seuraavasti:

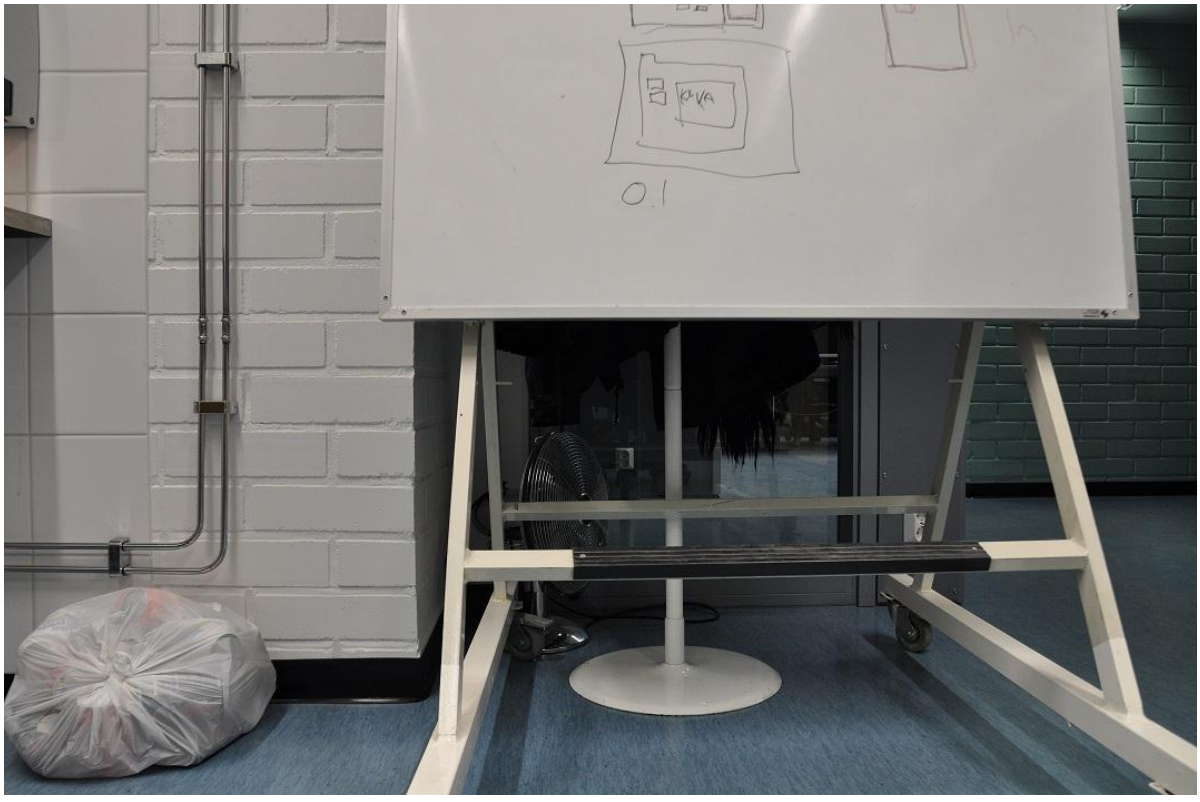
- Käynnistät terminaalin.
- Kirjoitat “*sudo apt-get install coriander*”, mikä asentaa ohjelman koneellesi.
- Asennuksen jälkeen kirjoita terminaaliin “coriander”, mikä käynnistää ohjelman.



Kuva 119. Coriander

5.5.2. Compact-, järjestelmäkamerat ja gphoto2

Koska FireWire sekä konenäkökamerat koettiin liian kalliiksi, mietittiin järjestelmä- sekä compact-kameroiden käyttämistä. Näiden kameroiden etäkäyttöön valmistajilta löydettiin kalliita kuvankäsittelyohjelmistoja, jotka tukivat lähinnä kalliimpia malleja compact- sekä järjestelmäkameroista. Avoimien sovellusten puolelta löydettiin Unixiin pohjautuville käyttöjärjestelmille, kuten Linuxille, gPhoto2, jolla voidaan ohjata tai ladata kuvia tietokoneen kautta kameralta USB:llä. Saatavissa on myös kirjasto libgphoto2, mikä toimii GNU LGPL lisenssin alla. Kuva 120 on otettu järjestelmäkameralla käyttäen gphoto2:sta. /53/



Kuva 120. Kuva käyttäen gphoto2:sta

gPhoto2

Tutkittiin, millaisia kameroita gPhoto2 tukee. Ilmeni, että se tukee noin 1400 eri kameraa, mutta suurimmassa osassa kameroita sillä pystyttiin vain lataamaan otetut kuvat kameroilta. Ainoastaan vanhemmat Olympos-merkkiset kamerat olivat tuettuina.

Testattiin komentoriviltä toimivaa käyttöliittymää (gphoto2) sekä GUI (gtkam). Näistä gphoto2 saatiin toimimaan paremmin. Linuxissa se asennetaan seuraavasti komentoriviltä:

sudo apt-get install gphoto2

Liitetyn kameran käyttöoikeudet Linuxissa

Ohjelmaa käyttäessä törmättiin muutamiin ongelmiin, mihin ei Windows-käyttöjärjestelmissä ollut tottunut, kuten kirjoitus-, luku-, suoritusoikeuksiin tiedostoilla, mutta myös laitteilla. Oikeudet ilmoitetaan ja muutetaan joko numeroin tai kirjaimin ja oikeudet jaetaan edelleen omistajan/luojan, ryhmän ja muiden oikeuksiin. Näitä varten on olemassa komentoriviohjelmia: /87/

- "*chown*":lla asetetaan yksi käyttäjä tai ryhmä omistajaksi kansiolle tai tiedostolle. "-R" lisäyksellä hakemistolle alihakemistoineen.
- "*chgrp*":lla asetetaan omistajaryhmä kansiolle tai tiedostolle. "-R" lisäyksellä hakemistolle alihakemistoineen.
- "*chmod*":lla muutetaan tiedostojen tai hakemistojen oikeuksia.

Numeroin annettavat ja ilmoitettavat oikeudet ovat 0-7, mitkä tarkoittavat seuraavaa: /87/

- 0 = ei oikeuksia
- 1 = suoritus
- 2 = kirjoitus
- 3 = kirjoitus ja suoritus (1+2)
- 4 = luku

- 5 = luku ja suoritus (1+4)
- 6 = luku ja kirjoitus (2+4)
- 7 = luku, kirjoitus ja suoritus (1+2+4). /87/

Esimerkiksi luodaan ryhmä “oppari”, annetaan luku, kirjoitus ja suoritusoikeudet “tiedosto.txt” ja poistetaan ne muilta. Lopuksi tarkastetaan oikeudet tiedostoon:

groupadd oppari
chgrp oppari /missä/lie/tiedosto.txt
chmod 0070 /missä/lie/tiedosto.txt
ls -l /missä/lie/tiedosto.txt

“*ls -l*” tulostaa komentoriville oikeudet tiedostoon. Esimerkissä se näyttäisi kutakuinkin: *“----rwx--- 1 root oppari...*”, missä:

- Ensimmäinen “-” merkki tarkoittaa, että kyseessä on tavallinen tiedosto.
- Seuraavat kolme “-” merkkiä omistajan oikeuksia, mitä ei ole.
- “r” tarkoittaa, että “oppari” ryhmällä on lukuoikeus.
- “w” tarkoittaa, että “oppari” ryhmällä on kirjoitusoikeus.
- “x” tarkoittaa, että “oppari” ryhmällä on suoritusoikeus.
- Seuraavat kolme “-” merkkiä muiden oikeuksia, mitä ei ole.

Laitteille voidaan asettaa vastaavanlaisesti oikeuksia. Laitteet Linux-käyttöjärjestelmissä löytyvät *“/dev/*” hakemiston takaa ja oikeudet laitteeseen voidaan tarkastaa samalla lailla kuin esimerkin tiedostoonkin, kunhan tiedetään polku oikein. Esimerkiksi “*ls -l /dev/video0*” tulostaa ensimmäisen liitetyn videolaitteen oikeudet, erona se, että ensimmäisen “-” tilalla olisikin “c”: “*c---rwx--- 1 root oppari...*”. /87/

Komentoja

Listauksen gphoto2:n tuetuista laitteista saa kirjoittamalla komentoriville:

gphoto2 --list-ports

Seuraavaksi testataan, löytääkö ohjelma USB-kameraa:

gphoto2 --auto-detect

Mikäli kaikki toimii oikein, pitäisi laitteen löytää kamera sekä sen merkki. Seuraavaksi voidaan katsoa tarkempia tietoja kytketystä kamerasta:

gphoto2 --summary

Seuraavalla komennolla voidaan tuetuilla kameroilla ottaa kuva ja ladata se tietokoneelle.

gphoto2 --capture-image-and-download

Hylkäsimme idean käyttää compact- tai järjestelmäkameraa, sillä tarvittiin kaksi samanlaista kameraa. Lisäksi kirjaston kameroiden ohjaustuet eivät pysyneet markkinoiden tahdissa ja tuetun, edullisen kameran löytäminen koettiin liian hankalaksi sekä aikaa vieväksi. Kameran näkyminen USB-massamuistina haittaa kameran käyttämistä kuvaustarkoituksiin. Lyhyen tähtäimen ratkaisuna komentorivillä irrotettiin kamera laitteesta ja etsittiin sitä “gphoto2 --auto-detect” komennolla. Pidemmälle käytölle olisi muutettu USB-kameran portin ominaisuuksia. Lisäksi libgphoto2-kirjaston avulla kamerat olisi yhdistetty käyttöliittymään.

5.5.3. Web-kamerat

Työssä päädyttiin käyttämään kameroina Web-kameroita. Ne olivat myös ensimmäisiä, joilla testattiin OpenCV:n toimintaa. Kuvassa 121 on kuva ensimmäisestä kamerasta, millä kirjaston toimintaa testattiin.

Kuva 121. Eräs OpenCV testikameroista

Hyviä ominaisuuksia

- Ovat edullisia, helppoja ja nopeita asentaa.
- Toimivat useilla eri käyttöjärjestelmillä.
- Toimivat poikkeuksetta Linux-koneissa.
- Suurimmassa osassa tiedonsiirto ja virransaanti on toteutettu USB-2.0:lla.

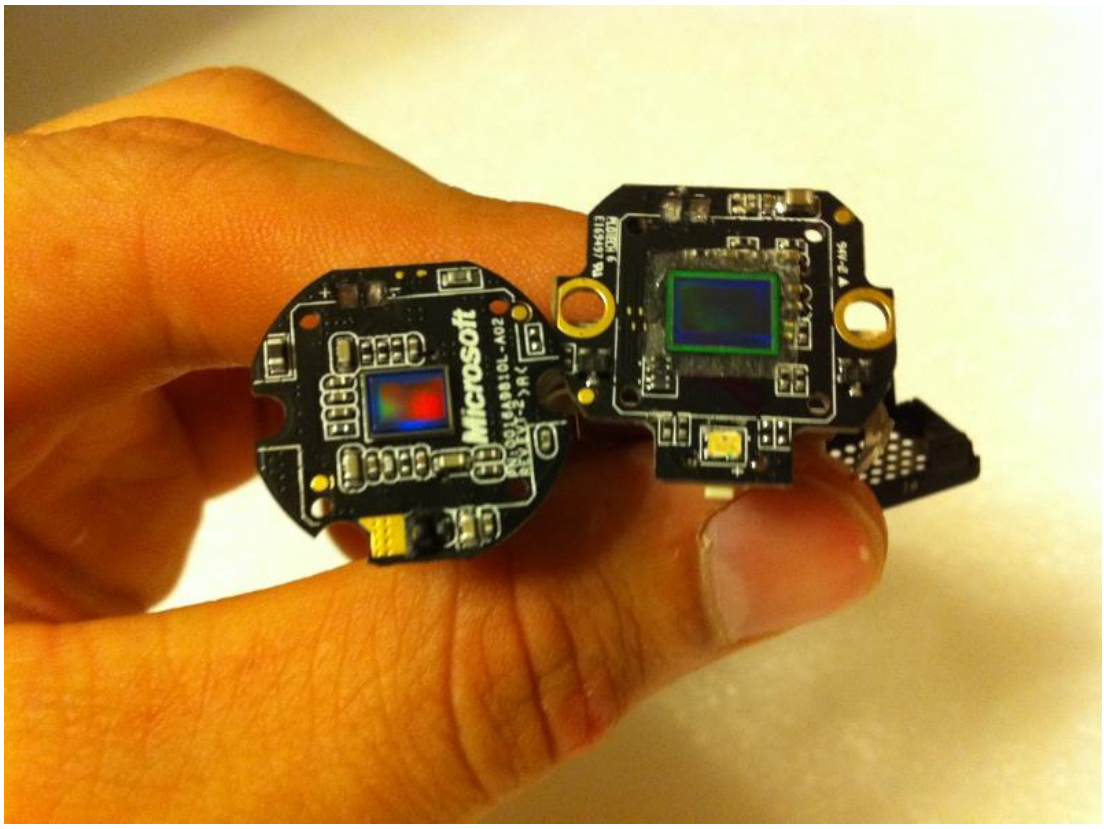
- Kuvan laatua, ohjelmaa sekä valaistusta helppo testata erilaisilla kuvankaappaus-ohjelmilla.

Rajoittavia ominaisuuksia

- Kameroissa on lyhyt kaapeli, joten kamera ja tietokone on oltava lähekkäin.
- Vanhoille kameroille ei ole ajureita uusimmissa Windows-käyttöjärjestelmissä.
- Noin 30 fps:llä voidaan saada maksimissaan 2 megapikselin kuvia.
- Ei mahdollista liittää optiikkaa ilman omia muutoksia kameraan, jolloin tarkkaan kuvaukseen kameran on oltava lähellä kohdetta.

Kameran valinta

Laitteeseen verrattiin erilaisia Web-kameroita. Parhaaseen hinta/laatusuhteeseen päästiin Microsoft LifeCam Studiolla. Studiota edullisemmasta versiosta Cinemasta löydettiin tietoja kennostakin, mutta Studiosta näitä tietoja ei löydetty. Studion kennon fyysinen koko on kaksinkertainen Cinemaan nähden, mikä tarkoittaa, että se voi kerätä enemmän valoa ja on näin vähemmän altis häiriöille. Kokoeron voi havaita kuvasta 122.



Kuva 122. Vasemmalla Microsoft LifeCam Cinema ja oikealla Studio /24/

Microsoft LifeCam Studion resoluutio on 1920 x 1080, mikä on noin 2 Mpikseliä. Tiedonsiirto tapahtuu USB-2.0:n kautta, josta se saa samalla käyttämänsä virran. Kamera on pienikokoinen ja se on helppo kiinnittää. Kameran tarkemmat tiedot löytyvät valmistajan datalehdestä. Kameran pääominaisuuksia on esitetty kuvassa 123. /84/

Kuva 123. Microsoft LifeCam Studio /85/

Studiossa havaittiin selviä ongelmia sen omien ajureiden kanssa. Kun Studiota käytettiin Windows-käyttöjärjestelmässä sen omilla ajureilla, päästiin noin 15 – 18 fps. Fps nousi 28 - 30 kun ajurit poistettiin tai käytettiin kameraa Linux-käyttöjärjestelmässä.

Kamerassa on auto-focus, automaattinen tarkennus, mitä käyttäjä ei pysty ohjaamaan. Tämä saattaa syntyä ongelmaksi joissain tilanteissa, missä kuva tai kamera liikkuu, jolloin kuva on epätarkka tarkennuksen seurauksena. Kameran täristessä epätarkkuus on jatkuvaa. Ainoa tapa saada tämä pois päältä on kolata kamerasta linssien moottorin virtajohdot irti. Lähin etäisyys, mihin kameralla pystyi tarkentamaan, on valmistajan tietojen mukaan n. 10 cm. Auto-focus on kamerassa suhteellisen hidas, mutta riittävä laitteen toiminnalle.

Kameran mikrofoni on erittäin huonolaatuinen, mutta tämä ei ole laitteen kannalta tärkeä ominaisuus, sillä sitä ei käytetä.

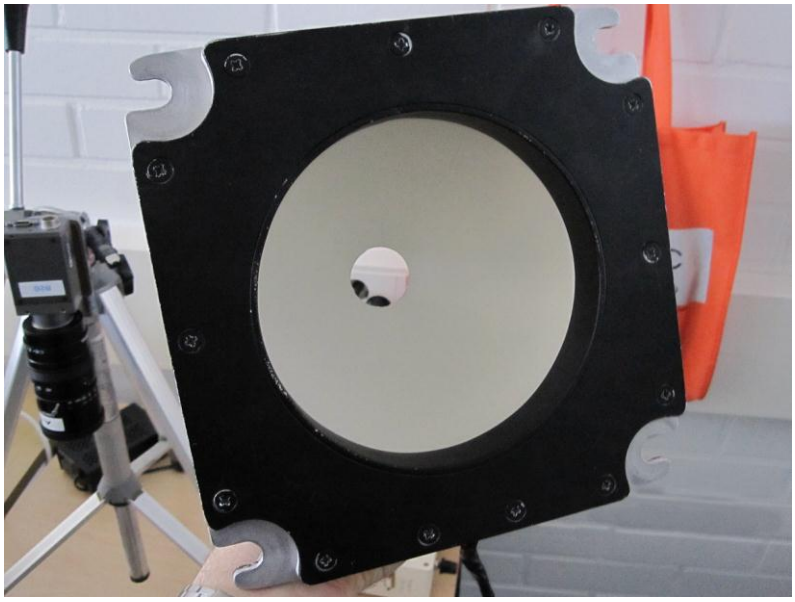
Kun kameralla kuvattiin, kamera lämpeni merkittävästi, mistä saattaa syntyä huomattavia häiriöitä kuviin käytettäessä laitetta pitemmän aikaa. Ongelmaa voidaan poistaa jäähdyttämällä kameraa.

5.6. Valaistus

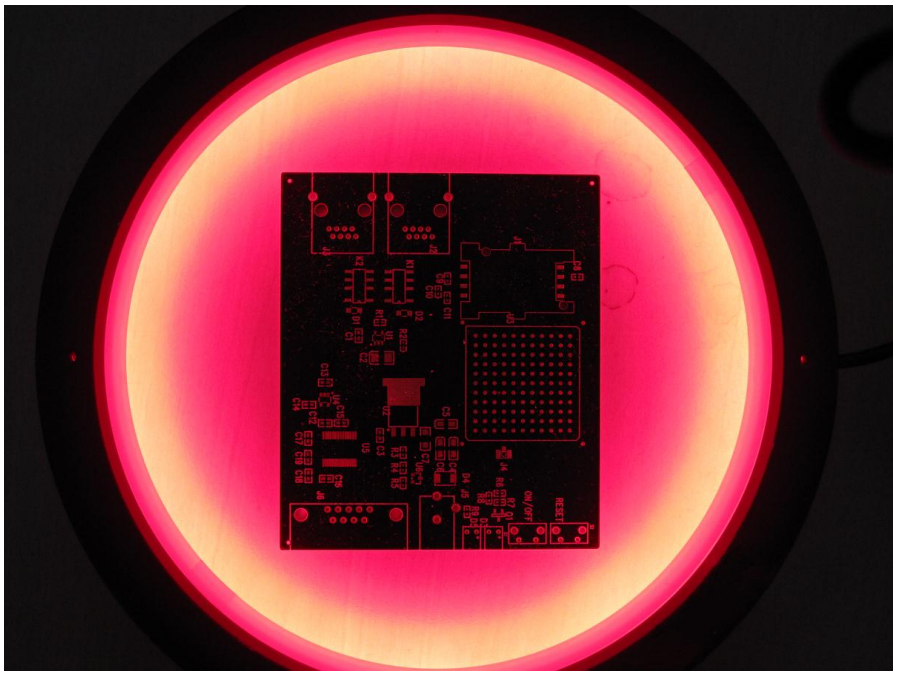
Valaistusta testattiin eri tavoin piirilevyille. Parhaaseen tulokseen päästiin, kun käytettiin kupolimaista dome-valoa (kuva 125) dark field-valaisimen päällä (kuva 124), jolloin piirilevyjen kytkennät erottuivat parhaiten. Kuvista 126 ja 127 nähdään valojen vaikutuksia piirilevylle ja kuvia yhteisvaikutuksista liitteestä 3. Kyseiset valot kuitenkin veisivät tilaa laitteelta huomattavasti ja ne olisi sijoitettava lähelle kohdetta, mikä ei ole mahdollista. Lisäksi valaistusta rakentaessa tulee ottaa huomioon ympäristön valojen vaikutus piirilevyyn sekä kohteeseen.

Käytettäessä web-kameroita päätettiin joko käyttää LED-valoja, jotka diffuusoitaisiin levyn, kuten hiotun pleksin avulla tai jättää valon käyttö kokonaan pois. Komponentille käytettäisiin kyseistä menetelmää taustavalona, jolloin reunat tulisivat selvästi näkyviin, tai komponentin reunoja korotetaan muilla keinoin, kuten taustan värillä. Piirilevyn valaistukseen menetelmää käytettäisiin lähinnä varjojen poistamiseen.

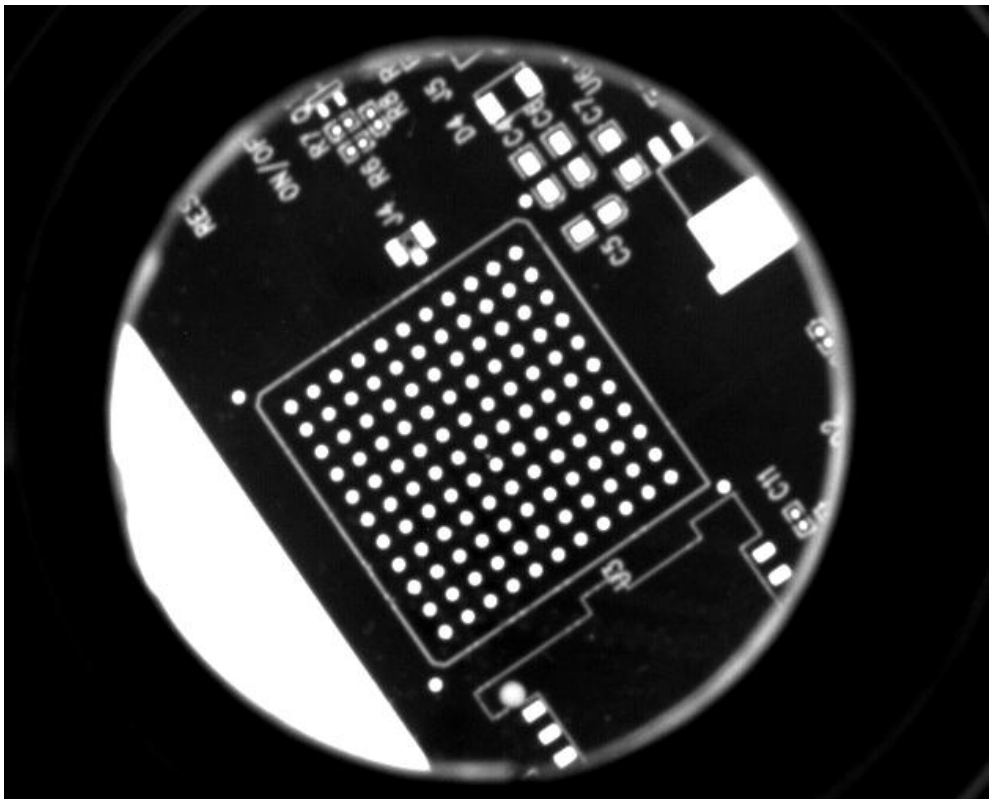
Kuva 124. Käytetty dark field-valaisin



Kuva 125. Käytetty dome-valaisin



Kuva 126. Piirilevy valaistaan dark field-valaisimella



Kuva 127. Asennuskohta valaistu dome-valaistuksella, kuva otettu konenäkökameralla

5.7. Kuvankäsittely

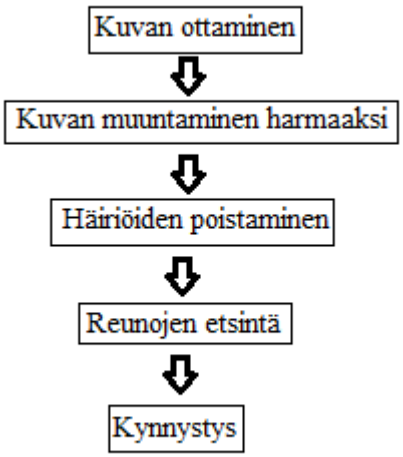
Kuvankäsittelyn suunnittelu koettiin vaikeaksi vähäisen kokemuspohjan takia ja siihen ei pystytty ajan puutteen takia keskittymään niin hyvin kuin se olisi vaatinut. Kuvankäsittelyn toimet suunniteltiin pääpiirteittäin, jonka jälkeen tutustuttiin erilaisiin OpenCV:n tarjoamiin suodattimiin, reunanhakuun sekä mm. viivojen sekä ympyröiden etsintäfunktioihin.

5.7.1. Kalibrointi

Ennen kuin voitaisiin aloittaa kuvankäsittely, täytyy järjestelmä kalibroida samalle tasolle kuin missä piirilevy sekä komponentti ovat. Tähän tarkoitukseen olisi lainattu konenäkölaboratoriosta erittäin tarkkoja kalibrointilevyjä tai käytetty OpenCV:ssä mukana tulevaa shakkiruudukkoa, minkä avulla OpenCV:ssä kalibrointi voidaan suorittaa. Kalibroinnin tarkoituksena on saada selville kuvassa näkyvien tarkkojen mittojen pituus pikseleinä sekä poistaa linsseistä johtuvat vääristymät. Tätä tietoa käyttäen voidaan mitata pituuksia kuvista. /39/

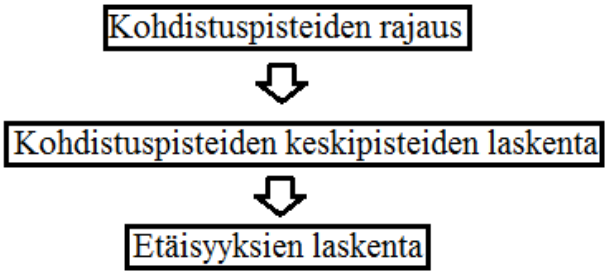
5.7.2. Kuvankäsittelyn suunnittelu

Kuvankäsittelyn suunnittelu aloitettiin siitä tilanteesta, kun kamerat kuvaavat sekä piirilevyä että komponenttia. Ideana oli, että käyttäjä pystyy ensimmäisissä versioissa tekemään itse mahdollisimman paljon komento kerrallaan sekä vaikuttamaan esimerkiksi kynnystysarvoihin, jotta päästäisiin parhaisiin tuloksiin sekä voitaisiin tutkia, millaisilla arvoilla parhaat tulokset saavutetaan. Myöhemmin voitaisiin automatisoida kaikki komennot muutamien vaiheiden alle.



Kuva 128. Kuvankäsittelyn alkuvaiheita

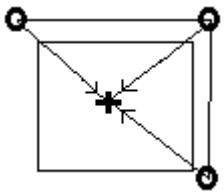
Kuvassa 128 on esitetty molempien kameroiden alkuvaiheet. Ohjelmassa ensin kaapataan kuva web-kameroilta, jonka jälkeen kuva muutetaan värikuvasta harmaakuvaksi, sillä tarvittavat funktiot suoritetaan harmaasävy- tai binäärikuville värikuvien sijaan. Tämän jälkeen kuvasta poistetaan häiriöitä ja parannetaan kontrastia parhaimman tuloksen saamiseksi. Viimeisinä yhteisinä vaiheina on mahdollinen reunojen etsintä sekä kuvan kynnystäminen harmaasävykuvasta binäärikuvaksi.



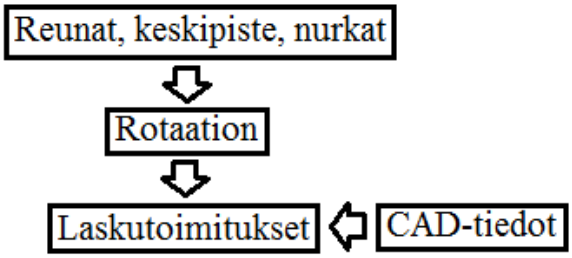
Kuva 129. Piirilevyn kuvankäsittely

Piirilevylle tehtävät käsittelyt alkavat kohdistuspisteiden rajauksella, kuten nähdään kuvasta 129. Rajaus tehtäisiin ainakin ensimmäisissä versioissa käsin käyttäjän toimesta kohdistuspiste

kerrallaan. Tämän jälkeen laskettaisiin jokaisen kohdistuspisteen keskipisteet sekä näiden etäisyydet toisistaan ja verrattaisiin CAD-ohjelmista saataviin tietoihin ja laskettaisiin asennettavan komponentin keskipisteen paikka, missä sen on oltava, jotta komponentti olisi oikeassa kohdassa. Kuvassa 130 on esitetty kuinka komponentin keskikohta voitaisiin laskea kohdistuspisteiden avulla.

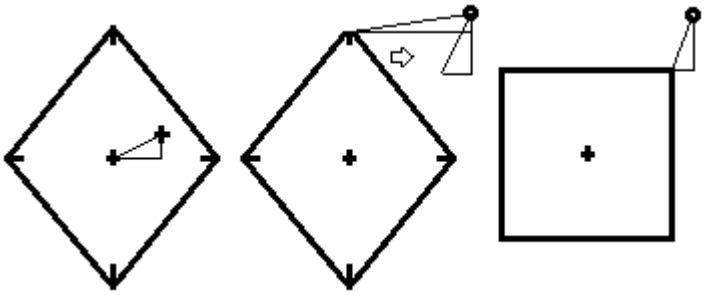


Kuva 130. Keskipisteen laskenta

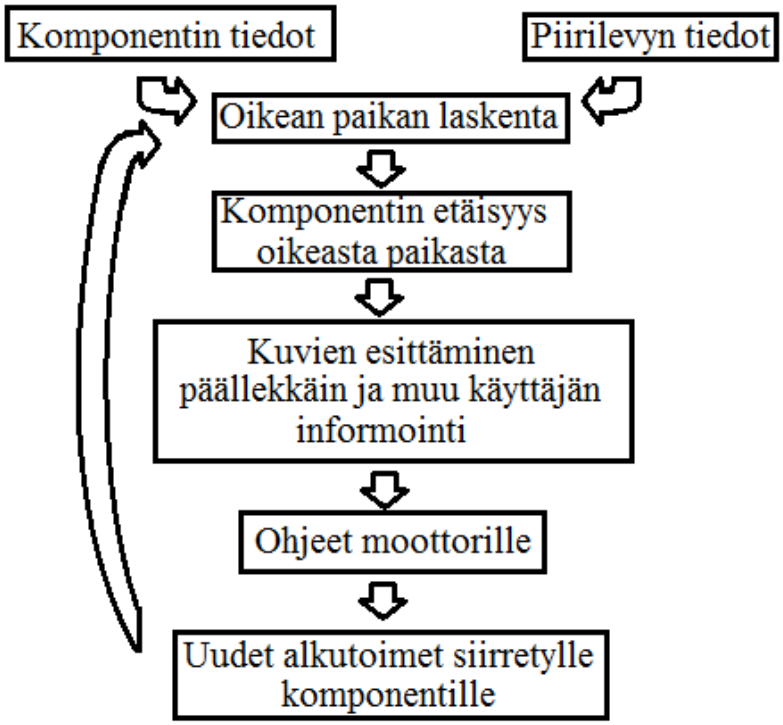


Kuva 131. Komponentin kuvankäsittely

Komponentille tehtävät käsittelyt aloitettaisiin komponentin reunojen, keskipisteiden sekä kulmien etsinnällä ja laskennalla. Tämän jälkeen verrattaisiin komponentin keskipisteen paikkaa piirilevyn kuvasta laskettuun komponentin keskipisteeseen sekä komponentin kulmien ja asennusmerkkien avulla oikea rotaatio. Kuvat 131, 132 ja 133 havainnollistavat miten työ tehdään.



Kuva 132. Komponentin siirtäminen ja rotaatio



Kuva 133. Yhteiset operaatiot

Kun Komponentin keskipisteen etäisyys oikeasta asennuskohdasta on laskettu, muutetaan tämä etäisyys mitoiksi moottoreille. Kun moottori on siirtänyt komponenttia toisella asennuspaikalla, otetaan komponentista uusi kuva ja aloitetaan komponentin osalta käsittely alusta, kunnes päällekkäin asetetuissa kuvissa komponentti on oikealla kohdalla. Tämän jälkeen moottorit siirtävät komponentin ennalta määrätyn matkan X-akselin suuntaisesti asennuskohdalle.

Muita ideoita

- Piirilevy valaistaan tietyllä värillä, kuten punaisella, ja käytetään punaisen värin värikanavaa harmaasävykuvana. Tällöin ei tarvitse muuntaa kuvaa harmaasävyksi.
- Kohdistuspisteiden valaisu siten, että korostuvat alueena muusta piirilevystä. Täten voitaisiin etsiä kohdistuspisteet ohjelman avulla eikä käsin.
- Komponentin keskikohdan sijaan tai lisäksi voitaisiin oikeaa asennuskohtaa laskea nurkkien tai reunojen avulla.
- Rotaation oikein saamiseksi voidaan esimerkiksi käyttää apuna videokuvaa, jolloin voidaan ohjata käyttäjää reaaliajassa kuvien ottamisen sijaan.

5.8. Ohjelmointi

5.8.1. OpenCV

OpenCV:n tutustuminen aloitettiin jo esiselvityksen aikana ja se jatkui läpi työn. Sen ohjelmointi aloitettiin Python-kielellä, sillä siitä meillä oli eniten kokemusta. Pythonilla toteutettu versio kuitenkin oli vielä selvästi keskeneräinen ja tästä johtuen ohjelmointikieli vaihdettiin C++:n. Tällöin saatiin myös Propalin työntekijältä Janne Vaaraniemeltä apua suunnittelussa ja ohjelmoinnissa. Ohjelmakielen vaihtamisen lisäksi kehitysympäristö vaihdettiin Qt:n, millä pystyttiin vaivattomasti ja nopeasti luomaan graafisia käyttöliittymiä.

Juuri ennen kun työ aloitettiin, oli OpenCV:stä julkaistu uusi versio 2.3.1, jossa oli muutettu asioita joiltain osin erilaiseksi, mikä hankaloitti esimerkiksi vanhojen oppaiden seuraamisen. Lisäksi tämä sisälsi uusia bugeja ja puutteita dokumentoinnissa. Kehittäjien tarkoituksena on kuitenkin uudistaa dokumentaatioita sekä ohjeita, mikä nähtiin jo työn aikana. /114/

OpenCV:n asentaminen

OpenCV:n kirjastojen asennuksessa epäonnistuttiin usean päivän koettamisen jälkeenkin Windows 7 sekä Vista käyttöjärjestelmille. Myöhemmin vastaan tulleet ohjeet olisivat saattaneet toimiakin. Sen sijaan koettiin helpommaksi asentaa OpenCV:n kirjastot Linux-käyttöjärjestelmälle. Aikaa käyttöjärjestelmän, kirjaston sekä ohjelmointiympäristöjen asentamiseen kului vain muutamia tunteja, vaikka Linux-osaamista ei ollut vielä karttunut. Hankalimmaksi koettiin cmake kääntäjän käyttö. /25/, /112/, /116/, /133/, /167/

5.8.2. Python

OpenCV:seen tutustuttiin ensin Python-kielelle käännettyyn versioon. Uudessa versiossa siihen oli tehty muutoksia mm. nimeämisen osalta. Tästä aiheutui vain pieniä ongelmia vanhempien versioiden tulkinnassa. Lisäksi kirjastoa ei tukenut uusinta pythonin 3.x versiota vaan vanhempaa 2.7 ja kaikkia OpenCV:n uuden kirjaston, cv2:n funktioita ei ollut, toisin kuin dokumentoinnissa mainittiin, käännetty Python-kirjastolle. /115/

Pythonin omat kirjastot ovat suoraan asennuksesta alkaen Ubuntussa mukana, mutta tarvittiin vielä IDE, Integrated Development Environment koodin tulkkaamiseen. Työn aikana käytimme sekä Pythonin omaa IDLE:ä sekä PyPe, mitkä olivat helppoja asentaa joko komentorivin tai Ubuntun oman latausohjelman kautta.

Ohjelmointi

Kun kaikki OpenCV:n kirjastot oli asennettu koneelle, voitiin aloittaa kirjaston testaaminen yksinkertaisin esimerkein. Kirjaston opettelemista hankaloitti se, että OpenCV:llä tehdyt oppaat ja esimerkit olivat usein C++:lla eivätkä Pythonilla ohjelmoituja. /73/

Virhetilanteiden analysointiin käytettiin try-except rakennetta, missä try:n sisään kirjoitettiin suoritettava funktio ja except:n sisään virheiden etsimisessä avuksi sys.exc_info().

```
try:
        # tänne funktio/koodi

except:
        print (sys.exc_info()[0])
        print (sys.exc_info()[1])
        print (sys.exc_info()[2])
        #end
```

Tämä rakenne palauttaa kolme “none” arvoa, mikäli ei tapahdu virhettä. Mikäli virhe ilmenee “try” osiossa, hypätään “except” osioon, missä sys.exc_info() palauttaa kolme arvoa, virheen tyypin, arvon sekä “traceback”:n eli tavallaan seuranna sinne, missä virhe tuli.

Kuvan ottaminen

Tämän kappaleen tarkoituksena on antaa perustietoja sekä koodinpätkiä, jotta jokainen voi päästä alkuun. Alussa tuodaan OpenCV:n kirjasto mukaan. Kirjastoja on kaksi, vanhempi cv sekä uudempi cv2. Kaikkia samoja funktioita ei ole molemmissa kirjastoissa ja saman funktion nimet voivat vaihdella. Pythonissa kirjastot otetaan käyttöön:

```
import cv, cv2
```

Kuvan ottaminen aloitetaan laitteen alustamisella. Mikäli laitteen numeroksi annetaan 0, tarkoitetaan tällä oletuslaitetta. Mikäli käytössä on vain yksi kamera tai kameralla ei ole väliä, voidaan sille antaa arvoksi -1. Kuva kaapataan kamerasta GrabFrame:lla puskuriin ja haetaan puskurista RetrieveFrame:lla. Tämä voidaan suorittaa myös yhdellä funktiolla, QueryFrame, mutta tämä ei toimi kuin yhden kameran kanssa.

```
pCapture =  cv.CaptureFromCAM(0)
…
```

cv.GrabFrame(pCapture)
pVideoFrame = cv.RetrieveFrame(pCapture)

Kuvan voi myös ladata tiedostosta tai tallentaa:

image = cv.LoadImage(‘picture.png’, cv.CV_LOAD_IMAGE_UNCHANGED)
cv.SaveImage(‘image.png’, image)

Luodaan ikkuna, missä kuvan halutaan näkyvän:
cv2.namedWindow('window',cv.CV_WINDOW_AUTOSIZE)

OpenCV:ssä voidaan luoda ikkunoihin liukukytkimiä, joilla tuoda ohjelmaan lisää ohjattavuutta, kuten arvojen vaihtamista, esim:

```
def set_scale1(val1):
   global distance
   distance = val1

cv2.createTrackbar('distance', 'window', distance, 260, set_scale1)
```

OpenCV:ssä on monenlaisia erikokoisia kuvia, mutta ne täytyy luoda ennen kuin kuvan tiedot voidaan siirtää niihin. Luodaan värikuva esimerkiksi web-kameran kuvaa tai piirtämistä varten:

image = cv.CreateImage(cv.GetSize (pVideoFrame), 8, 3)

Luodaan harmaasävykuva samalla tavalla, mutta vähennetään värikanavat yhteen:

gray = cv.CreateImage(cv.GetSize (pVideoFrame), 8, 1)

Koskaan ei kannata tehdä muutoksia alkuperäiseen kuvaan, koska muuten ei voida koskaan palata takaisin lähtötilanteeseen, joten kopioidaan kuva tyhjään kuvaan:

```
cv.SetZero(gray)
gray = cv.CloneImage(pVideoFrame)
```

tai vaihtoehtoisesti värikuvaan:

```
cv.SetZero(image)
image = cv.CloneImage(pVideoFrame)
```

Suurin osa funktioista ja operaatioista tehdään kuitenkin harmaasävykuvalle tai mustavalkokuvalle. Värikuva muunnetaan harmaasävykuvaksi seuraavasti:

```
cv.CvtColor(image,gray, cv.CV_RGB2GRAY)
```

Jotta kuva näkyisi ikkunassa, täytyy se näyttää siinä. Kuvaa ei näytetä mikäli WaitKey(ms):tä ei käytetä. Siinä odotetaan tietyn aikaa näppäintä, ennen kuin jatketaan:

```
cv.ShowImage("window", gray)
cv.WaitKey(50)
```

Lopuksi on hyvä tuhota ikkuna muistista:

```
cv2.destroyWindow("window")
```

Esimerkkejä

Lisää esimerkkejä ja pohja on työn lopussa liitteinä. (Liite 1), (Liite 2), (Liite 3).

5.8.3. C++ ja Qt:n käyttö

C++:n käyttöön siirryttiin työn valvojan käskystä, kun Pythonin versioiden kanssa törmättiin bugeihin ja vaikeuksiin, joiden ylitse emme päässeet. Lisäksi emme voineet hyödyntää suurinta osaa esimerkeistä, ohjeista tai Sulautettujen järjestelmien henkilökunnan osaamista sekä neuvoja täysin Pythonilla.

Kun vaihdoimme ohjelmointikielen, päätimme tehdä käyttöliittymän Qt:llä. OpenCV:n ja Qt:n yhdistäminen oli tullut esiin jo esitutkimuksessa, missä lähteestä löydettiin esimerkkejä Qt:llä käyttöliittymän tekemiseen. Pythonin testeissä käytettiin vain näppäimistön kirjaimia sekä OpenCV:n omaa ikkunaa. /78/

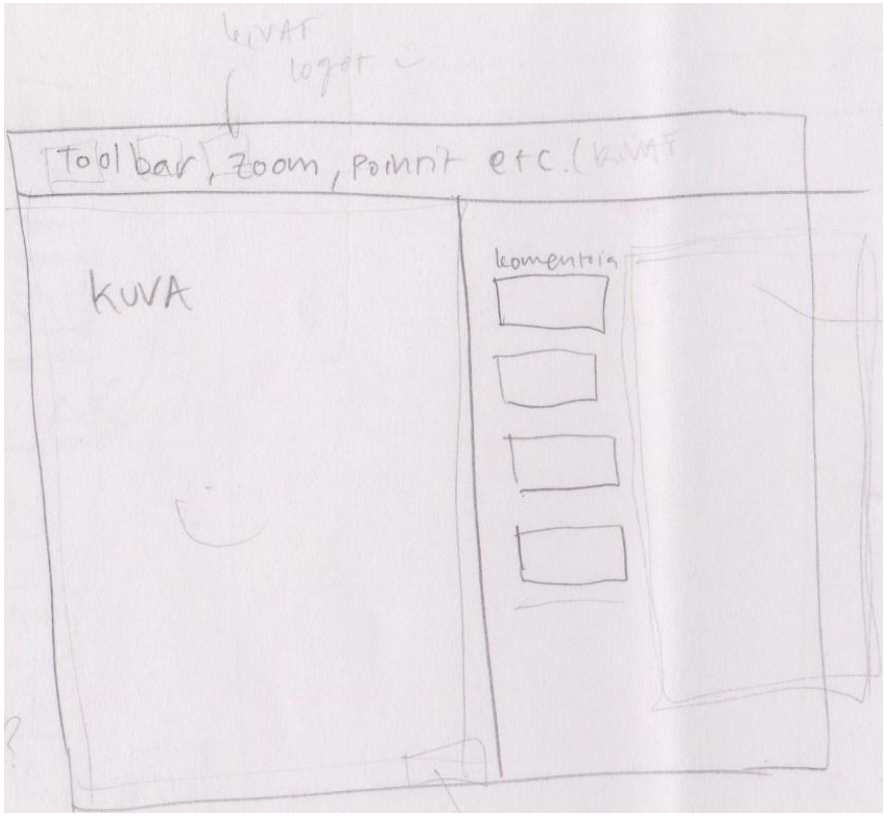
Vaihdon alussa tutkittiin mahdollisuutta integroida kuvan esittäminenkin Qt:n käyttöliittymään QLabel, QGraphicsView sekä Qt OpenGL:llä, mutta ne hylättiin ajanpuutteen ja asiaan soveltumattomuuden takia. Näiden avulla saataisiin uusia kuvankäsittelyn sekä esittämisen ominaisuuksia.

Säikeiden tekoon löytyi kaksi tapaa, “perinteinen” tapa, missä luodaan säie (thread), joka käynnistetään ja tekee mitä on käsketty, sekä toinen “oikeampi” tapa movetothread, missä ikään kuin työnnetään funktio toisen säikeen hoidettavaksi. Valitettavasti tästä “oikeasta” tavasta oli hyvin niukasti informaatiota sekä dokumentteja saatavilla, kuinka se toimii.

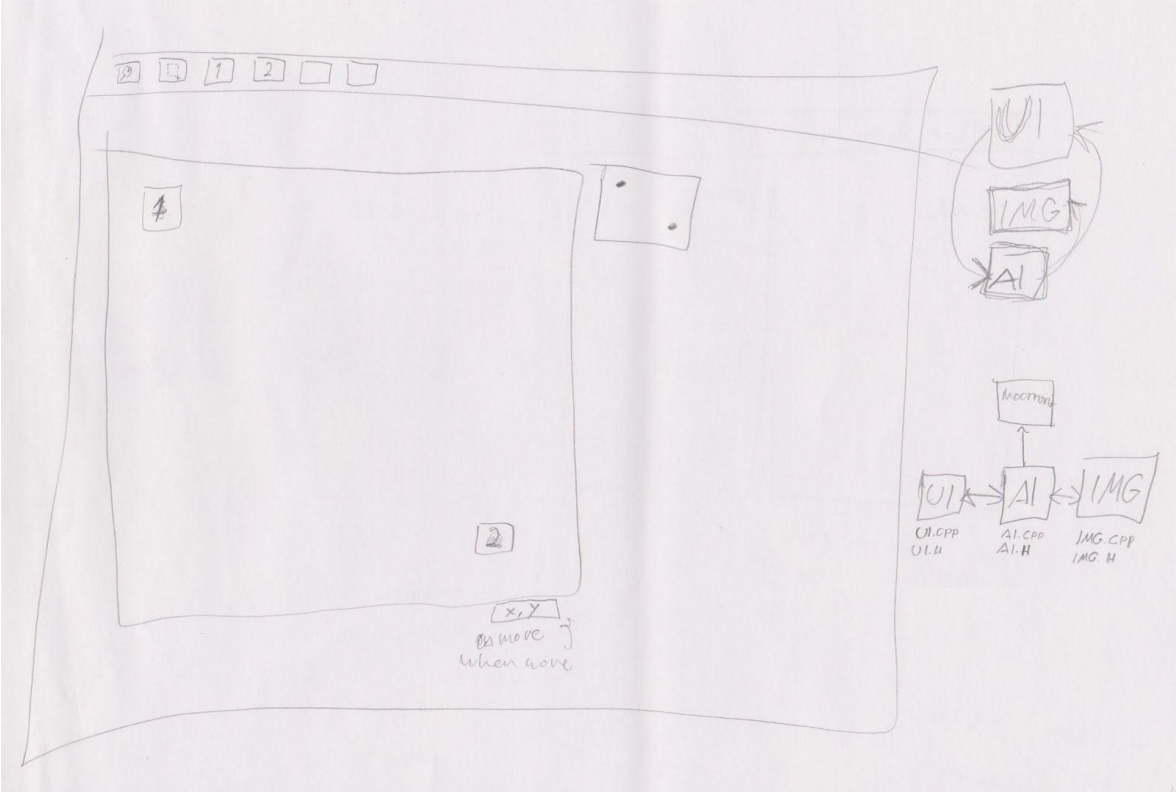
Käyttöliittymän suunnittelu

Tarkoituksen oli, että käyttöliittymien painikkeiden takana ei tehtäisi mitään funktioita, vaan komennot välitettäisiin “pääohjelmalle”, toiselle säikeelle, mikä tulkitsisi, mitä pitäisi tehdä ja antaisi tämän funktion suorittamisen kolmannen säikeen hoidettavaksi. Kun funktio on suoritettu, tarpeelliset ilmoitukset sekä tiedot palautettaisiin takaisin “pääohjelmalle” ja tarvittaessa käyttöliittymälle. Tällä tavoin voidaan taata, että aikaa vievien funktioiden takia ei tarvitse odotella, toisinsanoen tällä tavoin voidaan tehdä muutakin, kun yksi laskee.

Käyttöliittymän suunnittelun luonnoksia tehtiin sekä tussitaululle, että paperille lyijykynällä. Suunnitelmien edetessä huomattiin sen yksinkertaistuvan ja “turhien” lisäominaisuuksien katoavan. Viimeisimmässä suunnitelmassa käyttöliittymässä on vain komentoja ja OpenCV:n oma ikkuna toimii kuvan näyttäjänä. Kuvat 134 ja 135 ovat varhaisia luonnoksia käyttöliittymästä. (Liite 4)



Kuva 134. Käyttöliittymän suunnittelua



Kuva 135. Käyttöliittymän suunnittelua

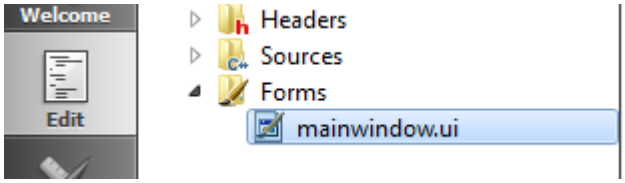
Qt esimerkki

Tässä kappaleessa esitettään kuinka tehdään yksinkertainen ohjelma, jossa painiketta painamalla otetaan kuvia web-kameralta 40 ms välein, muunnetaan kuva harmaasävyksi ja esitetään OpenCV:n omassa ikkunassa. Kuvaaminen loppuu, kun ikkunan ollessa aktiivisena painetaan jotain näppäimistön näppäintä. Esimerkin tarkoituksena on toimia mallina tai runkona sitä tahtoville.

Ensimmäiseksi on luotava QtCreatorilla uusi Qt Widget projekti, Qt Gui Application. Tämän jälkeen Qt:n uuden projektin projektitiedostoon on lisättävä seuraavat tiedot, jotta OpenCV kirjasto saataisiin toiminaan Qt:llä.

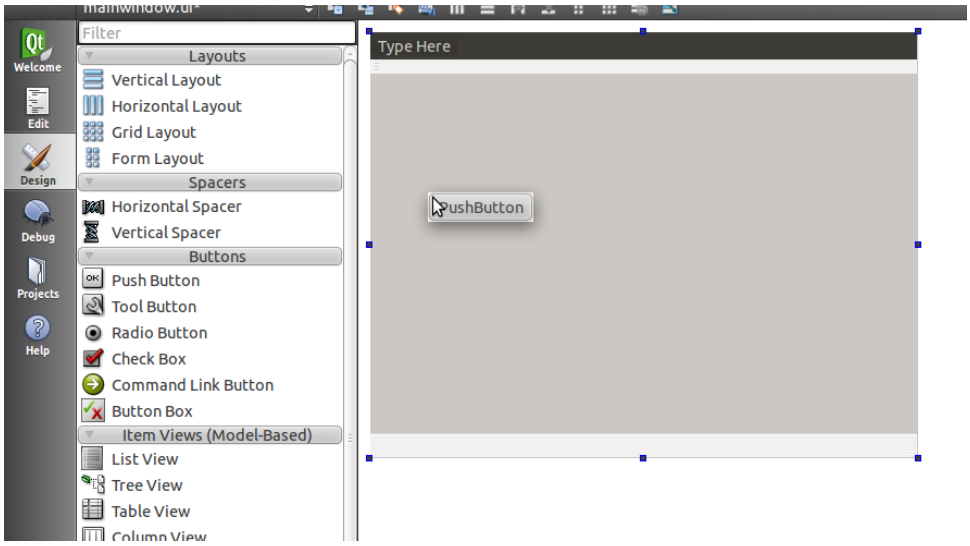
```
win32{
        INCLUDEPATH += C:\Opencv-2.3.1\include\

        LIBS += -LC\OpenCV-2.3.1\lib \
        -lopencv_core220 \
        -lopencv_imgproc220 \
        -lopencv_features2d220 \
        -lopencv_calib3d220
}

unix{
        CONFIG += link_pkgconfig
        PKGCONFIG += opencv
}
```

Lisäyksen jälkeen avataan “Forms” kohdan alta ui tiedosto, jolloin siirrytään suoraan Qt:n Design-työkaluun (kuva 136).



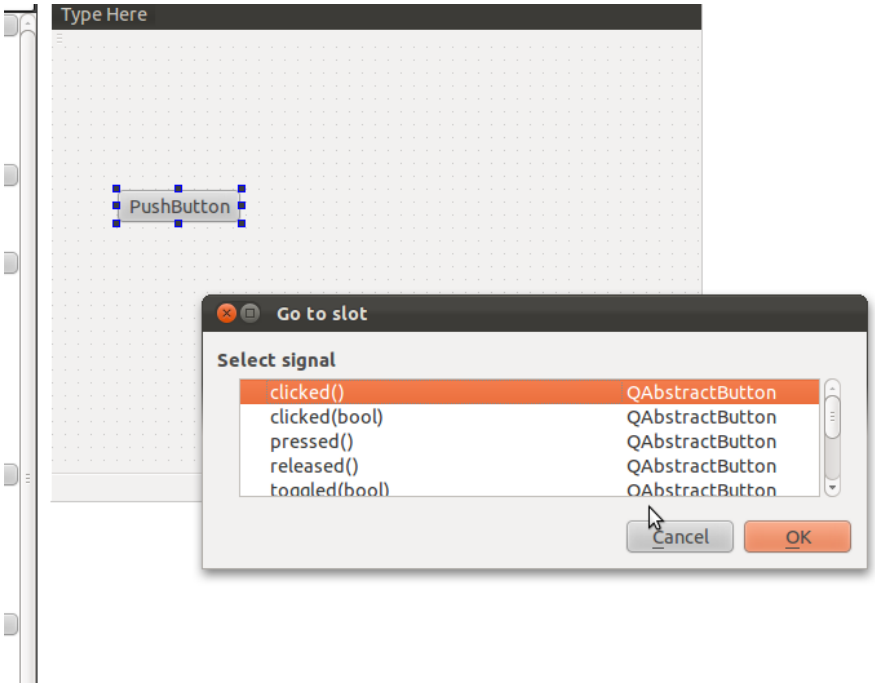
Kuva 136. ui

Design-työkalussa vasemmalla on pitkä lista erilaisia liukukytkimiä, painikkeita sekä muita graafisen käyttöliittymän elementtejä listattuna ryhmittäin. Valitaan “Buttons” kohdan alta “PushButton” ja vedetään se oikealla olevaan kenttään (kuva 137).



Kuva 137. Painikkeen vetäminen paikoilleen

Kun PushButton on saatu paikoilleen, valitaan se ja avataan hiiren oikeaa painiketta painamalla valikko. Valikosta valitaan “Go to slot” kohta, jolloin tulee “Select signal” valikko näkyviin (kuva 138). Valikosta valitaan kohta “clicked()” ja painetaan ok.



Kuva 138. Go to slot valikko

“clicked()” valinta luo cpp-tiedostoon funktion, joka suoritetaan kun painiketta painetaan. Samalla se määrittelee itsensä “private slot” kohtaan h-tiedostoon (kuva 139).

```
15
16  void MainWindow::on_pushButton_clicked()
17  {
18
19  }
20
```

Kuva 139. Clicked() valinta

Tämän jälkeen siirrytään h-tiedostoon, minne lisätään OpenCV:n kirjasto *“#include <opencv2/opencv.hpp”*, jolloin saadaan kirjaston funktiot käyttöön. Tämän lisäksi “*private*” osion alle lisätään määrittelyt käytettävästä kuvasta: “*cv::Mat image;*” (kuva 140).

mainwindow.h* image: cv::Mat

```
1   #ifndef MAINWINDOW_H
2   #define MAINWINDOW_H
3
4   #include <QMainWindow>
5   #include <opencv2/opencv.hpp>
6
7   namespace Ui {
8       class MainWindow;
9   }
10
11  class MainWindow : public QMainWindow
12  {
13      Q_OBJECT
14
15  public:
16      explicit MainWindow(QWidget *parent = 0);
17      ~MainWindow();
18
19  private slots:
20      void on_pushButton_clicked();
21
22
23  private:
24      Ui::MainWindow *ui;
25      cv::Mat image;
26
27  };
28
29  #endif // MAINWINDOW_H
30
```

Kuva 140. mainwindow.h

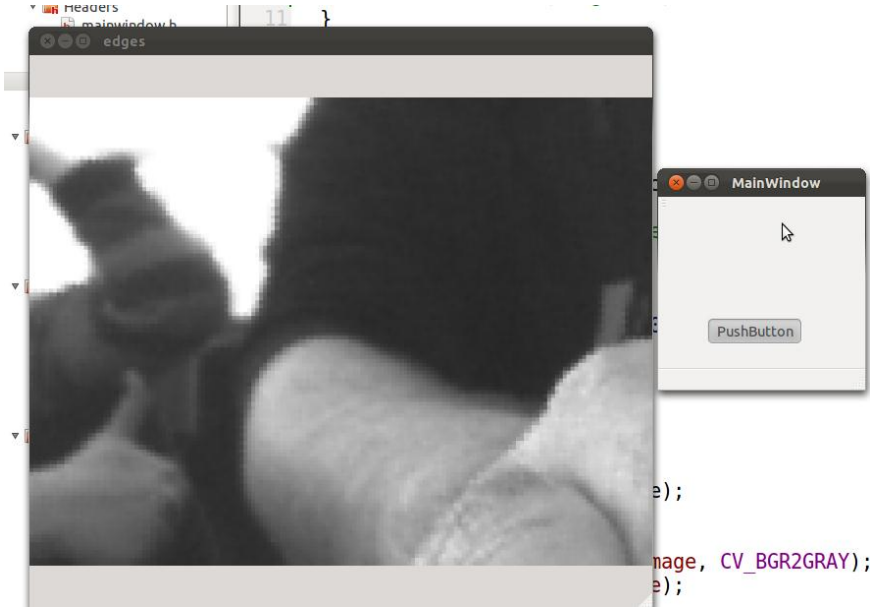
Kun kirjasto on lisätty ja kuva määritelty, siirrytään takaisin cpp-tiedostoon, missä "*ui->setupUi(this);*" rivin alle lisätään "*cv::namedWindow("edges",0);*", mikä luo "*edges*" nimisen ikkunan (kuva 141). Ikkunan voi nimetä mieleisekseen. "0" tarkoittaa samaa kuin "*CV_WINDOW_AUTOSIZE*". "on_pushButton_clicked()" funktion sisään lisätään alla olevat koodit.

mainwindow.cpp MainWindow::on_pushButton_clicked()

```
9
10      cv::namedWindow("edges",0);
11  }
12
13  MainWindow::~MainWindow()
14  {
15      delete ui;
16  }
17
18  void MainWindow::on_pushButton_clicked()
19  {
20      // tänne kirjoitetaan OpenCV:n koodeja
21
22      bool stop(false);
23
24      cv::VideoCapture device(0);
25      cv::Mat frame;
26
27      while(!stop){
28
29
30          device.grab();
31          device.retrieve(frame);
32
33
34          cv::cvtColor(frame,image, CV_BGR2GRAY);
35          imshow("edges", image);
36
37          if(cv::waitKey(40) >=0)
38              stop=true;
39      }
40      device.release();
41
42  }
43
```

Kuva 141. Painiketta painaessa suoritettava koodi

Esimerkissä määritellään alussa boolean-tyyppiä oleva “stop”, mikä asetetaan epätodeksi. Tätä käytetään lopettamaan videokuvan otto, kun näppäintä painetaan. Tämän lisäksi määritellään kuvankaappauslaite “device”, jonka uskotaan olevan oletuslaite, sekä uusi kuva “frame”, minne alkuperäinen kaapattu kuva otetaan puskurista. Tämän jälkeen aloitetaan while-silmukka, mitä tehdään niin kauan kun “stop” on epätosi, eli “stop” on muuttunut true-arvoon. Silmukassa ensiksi kaapataan kuva kameralta puskuriin ja haetaan kuva “frame”:n. Tämän jälkeen kuva muunnetaan värikuvasta harmaasävykuvaksi “image” kuvaan ja lopuksi näytetään muunnettu kuva OpenCV:n ikkunassa. Jokaisen kuvan jälkeen odotetaan 40 ms mitä tahansa näppäintä. Ilman tätä odotusta ohjelma ei toimi ollenkaan. Mikäli jotain näppäintä painetaan, muutetaan “stop” epätodesta todeksi, jolloin poistutaan silmukasta. Silmukasta poistuttua vapautetaan varattu laite. Kuvassa 142 on kuvankaappaus esimerkin käytöstä.



Kuva 142. Videokuvaa web-kameralta harmaasävyksi muutettuna

Muita esimerkkejä

Lisää erilaisia esimerkkejä, kuten Canny-funktioita, kuvien avaamista ja tallentamista on liitteinä (LIITE 4).

5.8.4. FrameBuffer overflow

Tämä virheilmoitus tuli sen jälkeen, kun olimme asentaneet FireWire- kortin koneeseen ja koetimme asettaa kameraksi USB web-kameraa. Virheilmoitus tuli sekä Python- että C++ kielillä. Virheen tullessa kameraa ei voitu asettaa kuvankaappauslaitteeksi. FireWire kameralla tätä virheilmoitusta ei tullut. Mika Wilén opinnäytetyössään toteaa FrameBufferista seuraavaa:

"*FrameCapturer-luokka vastaa kuvien lukemisesta käyttäjän antaman konfiguraation määrittelemästä videolähteestä. Luokka toimii omassa säikeessään ja täyttää FrameBuffer-kuvapuskuria videolähteen tarjoamalla nopeudella. Mikäli videolähteeksi on valittu videotiedosto, yksittäiset kuvat kaapataan videon toistonopeudella, vaikka kirjaston suorituskyvyn puitteissa kuvat olisikin mahdollista sekä kaapata että prosessoida moninkertaisella nopeudella*." /202/

FrameBuffer on siis kuvapuskuri, johon kerätään kameralta saapuva kuva mahdollisimman nopeasti. Etsinnöistä huolimatta virheilmoitukseen ei löytynyt vastausta lähdekoodista, virallisilta tahoilta eikä foorumeilta, vaikka ongelma ei ollut harvinainen.

Ongelman kiertämiseksi löytyi yllättäen ratkaisu tehdessä käyttöliittymää Qt:llä. OpenCV:llä on mahdollista käyttää kahdeksaa kameraa yhtä aikaa, joten jokaiselle näistä on oma puskuri. Käymällä kaikki kahdeksan puskuria sekä muutaman ylimääräisen paikan silmukassa läpi ja samalla tarkastaen, onko kamera kytketty ja voiko sen "avata" kuvankaappaukselle virheilmoituksista huolimatta pystyttiin ottamaan Web-kameralla kuvaa.

Ongelma ilmeisesti syntyi, kun kirjaston kääntämisen jälkeen tehtiin laitemuutoksia koneeseen. Ongelma olisi varmasti kadonnut kirjaston uudelleen asennuksella, mikä oli toiminut myös monelle muulle ongelmasta kärsineelle. Ongelma korjaantui kuin itsestään opinnäytetyön kirjoitusvaiheessa, jossa FireWire-kortti irrotettiin ja asennettiin uudelleen samaan paikkaan.

6. YHTEENVETO

Mielestämme työmme onnistunein osa oli projektisuunnitelma, mikä oli hyvä ja sitä seurattiin tahtomattakin. Ainoa epäonnistunut osa siitä oli aikataulun toteutuminen, mikä johtui osaamisen puutteesta ja työmäärän aliarvioinnista verrattuna saamaamme opetukseen. Tästä johtuen laitteesta ei saatu aikaiseksi prototyyppiä, vaan työ oli lähinnä tutkimustyötä ja selvitystä siitä, miten tällaisen laitteen voisi tehdä kustannustehokkaasti.

Eräs suunnittelun helpottamiseen käytetty väline oli iso tussitaulu, jolle mm. kirjattiin ongelmia sekä tehtiin ensimmäisiä versioita luonnoksista ja suunnitelmista. Pelkästään asian kirjoittaminen tai piirtäminen taululle usein auttoi asiassa eteenpäin.

Dokumentoinnissa Google Docs osoittautui korvaamattomaksi. Käyttäjät pystyvät kirjoittamaan samaa dokumenttia yhtä aikaa eri koneilta. Dokumentin pystyy jakamaan muiden kanssa sähköpostin välityksellä. Muille käyttäjille voidaan antaa eri oikeuksia dokumentin selaukseen. Lukuoikeudella käyttäjä pystyi vain lukemaan dokumentin, kirjoitusoikeudella käyttäjä sai luvan muokata dokumenttia, kommentointioikeudella käyttäjä voi lukea dokumentin ja antaa kommentteja, jotka näkyvät dokumentin laidalla. Google Docsissa on oma version hallinta järjestelmä, joka tuli silloin tällöin tarpeeseen, kun halusi poistettuja kappaleita takaisin. Valmiin dokumentin voi ladata esim. Word tai PDF-tiedostona.

Myös toinen ohjelma, Dropbox, osoittautui käteväksi työtä tehdessä. Se on pilvipalvelin, minne voi ilmaisversiossa tallentaa tiedostoja aina 2 Gt:un asti ja päästä käsiksi koneelta, johon Dropbox on asennettuna. Tällä tavoin työn aikana tärkeät dokumentit voitiin jakaa kahden tai useamman käyttäjän kesken sekä työkoneelta toiselle.

Työtä olisi varmasti ollut useammallekin kuin kahdelle henkilölle, koska jokainen asia työssä oli aivan uutta asiaa, jolloin suurin osa ajasta käytettiin tiedonhankintaan, ohjelmoinnin sekä kuvankäsittelyn harjoitteluun. Moottoreiden valintaan käytettiin myös aikaa turhan paljon, koska tiedot olivat usein sekavia ja ristiriidassa eri lähteiden kanssa. Mekaniikan

suunnittelussakin meni kauan, koska ei tiedetty mitä asioita tuli ottaa huomioon tai miten suunnittelun pitäisi edetä. Ohjelmointi oli sen verran vaativaa, että pelkästään siinä olisi ollut töitä kahdelle, toinen tekisi käyttöliittymän ja moottoreiden ohjauksen ja toinen keskittyisi kuvankäsittelyyn. Ainoa valmis asia oli jo olemassa oleva ladontakone, muut asiat piti ensin tutkia, sen jälkeen suunnitella ja toteuttaa.

Työtä hankaloitti myös tietämättömyys työhön budjetoidusta rahamäärästä sekä kuinka osia tilataan. Laite voidaan toteuttaa useilla eri tekniikoilla, mutta tutkimuksen edetessä hintojen noustessa tietyillä tekniikoilla jouduttiin etsimään uusia, edullisempia toteutusvaihtoehtoja.

Ohjelmoinnin suunnitteluun, vuokaavioihin ja vastaaviin olisi myös tullut käyttää enemmän aikaa. Valitettavasti tämä todettiin liian myöhään testaillessa kuvankäsittelykirjastoa ja harjoitellessa ohjelmointia, sillä tietämykset esimerkiksi UML:stä olivat hatarat entuudestaan. Lisäksi ohjelmointikielen vaihtuminen Pythonista C++:n aiheutti totuttelu ongelmia sekä turhautumista, sillä Pythonin opetteluun oli käytetty työstä paljon aikaa. Loppujen lopuksi vaihto oli parempi ratkaisu.

Meidän testaamilla servomoottoreilla tätä työtä ei saisi tehtyä ilman vaihteiston ja potentiometrin lisäämistä moottoreihin. Moottorit jaksavat liikuttaa vartta helposti, mutta paikkatietoa niistä ei saa, ilman lisäosia. Jonkinlaiset anturit voisivat sopia tähänkin tarkoitukseen jotta saadaan tarkat paikkatiedot.

Työn aikana syntyi monia erilaisia toteutustapoja sekä tekniikoita laitteeseen. Suurin osa näistä jouduttiin hylkäämään joko ajanpuutteen tai näiden tuomien lisäkustannuksien takia. Tällaisia olivat:

- Erilaisia rakennevaihtoehtoja, kuten moottorien integrointi ladontakoneen sisäpuolelle, jolloin ei tarvitsisi tilata kelkkaa tai alumiinikiskoja erikseen.
- Moottoreiden korvaaminen lineaariyksiköillä, mutta tämä toisi lisäkustannuksia.
- Web-kameran korvaaminen paremmalla kameralla, johon saadaan kiinni objektiiveja. Tällöin kameraa ei ole pakko sijoittaa lähelle kohdetta.

- gphoto2:n käyttö compact- tai järjestelmäkameroita käyttäen.
- Tietokoneen hiiren käyttö käden liikkeiden mittaamiseen.

Hyvät askelmoottorit ja niihin hyvät ohjaimet saattaisivat olla parempi vaihtoehto jos on vähän enemmän rahaa käytettävissä, koska vääntöä ei tarvitse paljoa niin askelten hukkuminenkin olisi epätodennäköisempää.

Tietääksemme lähialueella ei ole vastaavanlaista laitetta, joilla pystyisi latomaan BGA-komponentteja. Tämä saattaisi lisätä laitteen käyttöä valmistuessaan.

7. LÄHDELUETTELO

A

/1/ Actel, EIA Standard Board Layout Drawing for BGA, CCGA, CSP, and QFN, [pdf], [http://www.actel.com/documents/RecPCBdesign.pdf], 27.2.2012.

/2/ Advanced Techniques US Inc. (ATCO), BGA Removal and Installation, [WWW-dokumentti], [http://atco-us.com/applications/info/3], 27.2.2012.

/3/ Adrio Communications, [WWW-dokumentti], [http://www.radio-electronics.com/info/manufacture/soldering/smt-soldering/bga-solder.php], 27.2.2012.

/4/ Adrio Communications, [WWW-dokumentti], [http://www.radio-electronics.com/info/data/smt/bga.php], 27.2.2012.

/5/ Aegis electronic group INC, [kuva], [http://www.aegis-elec.com/products/img/Foculus-FO124SB.jpg], 4.3.2012.

/6/ Ahmed Althaf, What is BGA, [WWW-dokumentti], [http://www.articlesbase.com/computers-articles/what-is-bga-711605.html], 27.2.2012.

/7/ AIA, Leading the Way in Machine Vision Industry Standards, [WWW-dokumentti], [http://www.visiononline.org/vision-standards.cfm], 4.3.2012.

/8/ Airila Mauri, Mekatroniikka, 7. muuttumaton painos, Otatieto, 2004.

/9/ Altera, Designing With High-Density BGA Packages for Altera Devices, [pdf], [http://www.altera.com/literature/an/an114.pdf], 27.2.2012.

/10/ Alternative vision, Separation prism technology, [WWW-dokumentti], [http://www.alt-vision.com/color_prisms_tech_data.htm],1.3.2012.

/11/ Anaheim Automation, Introduction To Step Motor Systems, [WWW-dokumentti], [http://www.anaheimautomation.com/intro.htm], 27.2.2012.

/12/ August Research Systems Inc.,Digital camouflage, [kuva], [http://www.augustrs.com/DigiCamotmpl.html], 1.3.2012.

/13/ AWAIBA, CMOS IMAGE SENSOR TECHNOLOGY, [WWW-dokumentti], [http://www.awaiba.com/en/technology/], 4.3.2012.

/14/ Awaiba, AREA SCAN IMAGE SENSORS FOR INDUSTRIAL VISION, [kuva], [http://www.awaiba.com/en/products/areascan-image-sensors/], 27.1.2012.

/15/ Ayob Masri , Cowling Peter Kendall Graham, Optimisation for Surface Mount Placement Machines, [WWW-dokumentti], [http://www.cs.nott.ac.uk/~gxk/papers/icit02b.pdf], 27.2.2012.

B

/16/ Blanchette Jasmin, Summerfield Mark,C++ GUI Programming with Qt 4, 2. painos, 2009.

/17/ Bradski Gary, Kaehler Adrian, Learning OpenCV Computer Vision with the OpenCV Library, [pdf], [http://www.cse.iitk.ac.in/users/vision/dipakmj/papers/OReilly%20Learning%20OpenCV.pdf], 4.3.2012.

/18/ Brno University of Technology, IMMI - Rapidminer 5 Image Mining Extension, [WWW-dokumentti], [http://spl.utko.feec.vutbr.cz/en/component/content/article/46-image-processing-extension-for-rapidminer-5], 4.3.2012.

C

/19/ Cambridge in colour, Digital camera sensor sizes, [kuva], [http://www.cambridgeincolour.com/tutorials/digital-camera-sensor-size.htm], 27.1.2012.

/20/ Cambridge in colour, Digital camera sensors, [http://www.cambridgeincolour.com/tutorials/camera-sensors.htm], 1.3.2012.

/21/ Cambridge in colour, Image Interpolation, [http://www.cambridgeincolour.com/tutorials/image-interpolation.htm], 1.3.2012.

/22/ Chen Ting, A Study of Spatial Color Interpolation Algorithms for Single-Detector Digital Cameras, Course Project, Department of Electrical Engineering, Stanford University, 1999, [pdf], [http://vigir.missouri.edu/~gdesouza/Research/ColorCCD/ColorInterpolation.pdf], 1.3.2012.

/23/ Chugh Anurag, kuva, [kuva], [http://lare-india.blogspot.com/2010_03_01_archive.html], 27.2.2012.

/24/ Cloudynights, Cineman ja Studion vertailu, [kuva], [http://www.cloudynights.com/ubbthreads/attachments/4736186-Size%20Matters.JPG], 4.3.2012.

/25/ Cmake, Open-source build system, [WWW-dokumentti], [http://www.cmake.org/cmake/help/runningcmake.html], 3.3.2012.

/26/ Computer Active, The men who really invented the GUI, [WWW-dokumentti], [http://www.computeractive.co.uk/pcw/pc-help/1925325/the-invented-gui], 4.3.2012.

/27/ CVI melles griot, Machine Vision Lighting Fundamentals, [pdf], [http://www.cvimellesgriot.com/products/Documents/TechnicalGuide/Machine_Vision_Lighting_Fundamentals.pdf], 4.3.2012.

D

/28/ Danaher motion, Linear units, [kuva], [http://www.tollo.com/pdf/LMS_EU200502-03_july07.pdf], 27.2.2012.

/29/ Delta enterprise, Konenäköjärjestelmät, [pdf], [https://noppa.aalto.fi/noppa/kurssi/as-116.1100/luennot/AS-116_1100_luentokalvot__konenako.pdf], 4.3.2012.

/30/ DigiFAQ, Vääristymät, [WWW-dokumentti], [http://digifaq.info/digifaq/3_vaaristymat.html], 4.3.2012.

/31/ DigiFAQ, Alustus salamavaloon, [WWW-dokumentti], [http://digifaq.info/digi_omat/rsalama/index.html], 4.3.2012

/32/ Digikamera.net, Polarisaatiosuotimet, [WWW-dokumentti], [http://www.digicamera.net/vinkit/vinkki_polarisaatio.htm], 4.3.2012.

/33/ Digitalfanatics, The Qt Object Model, [WWW-dokumentti], [http://www.digitalfanatics.org/projects/qt_tutorial/chapter02.html], 4.3.2012.

/34/ Digital Outback Photo, color separation, [WWW-dokumentti], [http://www.outbackphoto.com/dp_essentials/dp_essentials_03/essay.html], 1.3.2012.

/35/ Dima smt systems, FP Manual Pick & Place, [pdf], [http://www.dimasmt.com/cmdata/docs/FP-500-600%20E.pdf], 7.3.2012.

/36/ DVINFO, Blooming, [kuva], [http://www.dvinfo.net/forum/attachments/canon-vixia-series-avchd-hdv-camcorders/2729d1176263434-hv10-20-rolling-shutter-ccd_flash.jpg], 4.3.2012.

/37/ Dvxuser, SENSOR ARTIFACTS AND CMOS ROLLING SHUTTER, [WWW-dokumentti], [http://dvxuser.com/jason/CMOS-CCD/], 4.3.2012.

/38/ Dx0 Labs, Lens distortion, [WWW-dokumentti], [http://www.dxo.com/en/photo/dxo_optics_pro/features/optics_geometry_corrections/distortion], 4.3.2012.

/39/ DysonFLO, Camera calibration using OpenCV, [WWW-dokumentti], [http://dsynflo.blogspot.com/2010/03/camera-calibration-using-opencv.html], 4.3.2012.

E

/40/ MICROSCAN, Eight Tips for Optimal Machine Vision Lighting, [WWW-dokumentaatio], [http://www.qualitydigest.com/inside/metrology-article/eight-tips-optimal-machine-vision-lighting.html#], 4.3.2012.

/41/ Electronics Engineering, Digital image processing, [WWW-dokumentti], [http://nptel.iitm.ac.in/courses/Webcourse-contents/IIT-KANPUR/Digi_Img_Pro/ui/TOC.htm], 4.3.2012.

/42/ Electromagnetic spectrum, [kuva], [http://www.lpi.usra.edu/education/fieldtrips/2005/activities/ir_spectrum/images/emspectrum.jpg], 4.3.2012.

/43/ Engineer, The, SMT Pick-and-Place Machine for Placement of SMD Electronic Components on PCB, [WWW-dokumentti], [http://the-engineer.hubpages.com/hub/SMT-Pick-and-Place-Machine], 27.2.2012.

/44/ EMVA, An introduction to machine vision, [WWW-dokumentti], [http://www.emva.org/cms/index.php?idcat=38&lang=1], 4.3.2012.

/45/ Etsin, [kuva], [http://digikuvaus.medianurkka.com/wp-content/uploads/2010/03/etsin-300x300.jpg], 4.3.2012.

F

/46/ FFI, INTRODUCTION TO COMPUTER GRAPHICS AND VISUALIZATION, [kuva], [http://prosjekt.ffi.no/unik-4660/lectures04/chapters/Introduction.html], 1.3.2012.

/47/ Field of vision, [kuva], [http://codinghorror.typepad.com/.a/6a0120a85dcdae970b0120a86d9495970b-pi], 4.3.2012.

/48/ Fineline, FP-100/200/300/500/600/700 specifications, [pdf], [http://www.desab-elektronik.se/prod/dimasmt/broschyr/fineline.pdf], 7.4.2012.

/49/ Forsyth David, Ponce Jean, Computer Vision A Modern Approach, Pearson Education Inc, 2003.

/50/ Foveon, An entirely new way to capture color, [WWW-dokumentti], [http://www.foveon.com/article.php?a=67], 1.3.2012.

/51/ Freescale, Ball Grid Array (BGA) Packaging Technology, [WWW-dokumentti], [http://www.freescale.com/webapp/sps/site/overview.jsp?code=TM_RD_PKG_BGA], 27.2.2012.

G

/52/ GigE Vision Referemce Desing, Document Revision A-0.2.3, [standardi].

/53/ gphoto2, Digital camera software, [WWW-dokumentti], [http://www.gphoto.org/], 1.3.2012.

/54/ Graftekimaging INC, Machine Vision Lighting, [WWW-dokumentti], [http://www.graftek.com/pages/lighting-ni.htm], 4.3.2012.

/55/ Gurevich Gerald,Wilson Tracy, Nice Karim, How Digital Cameras Work, [WWW-dokumentti], [http://electronics.howstuffworks.com/cameras-photography/digital/digital-camera6.htm], 4.3.2012.

/56/ GyesenSteiner Associates, CCD or CMOS?, [kuva], [http://www.gyes.eu/photo/sensor_pixel_sizes.htm], 6.1.2012.

H

/57/ Hamamatsu, Pixel binning, [WWW-dokumentti], [http://learn.hamamatsu.com/articles/binning.html], 1.3.2012

/58/ Haußecker Horst, Jähne Bernd, Computer Vision And Applications, Academic Press, 2000.

/59/ HiCow, What is BGA?, [WWW-dokumentti], [http://www.hicow.com/printed-circuit-board/pin-grid-array/solder-2794424.html], 27.2.2012.

I

/60/ Imageprocessingbasics, [WWW-dokumentti], [http://www.imageprocessingbasics.com/], 3.3.2012.

/61/ Image Processing On Line, [WWW-dokumentti], [http://www.ipol.im/], 3.3.2012.

/62/ Imaging resource, CMOS versus CCD & what's it all mean, [WWW-dokumentti], [http://www.imaging-resource.com/PRODS/D30/D30A4.HTM], 6.1.2012.

/63/ Imaging Source, The, How color cameras work, [pdf], [http://www.theimagingsource.com/downloads/howcolcamswp.en_US.pdf], 1.3.2012.

/64/ imaging toolkit, [WWW-dokumentti], [http://www.imaging-toolkit.com/imaging-glossary.htm], 3.3.2012.

/65/ Intel, Ball Grid Array (BGA) Packaging, [pdf], [http://www.intel.com/content/dam/www/public/us/en/documents/packaging-databooks/packaging-chapter-14-databook.pdf], 27.2.2012.

J

/66/ Jauhiainen Jukka, Digitaalinen kuvankäsittely II, Oulun seudun ammattikorkeakoulu, [pdf], [http://www.oamk.fi/~jjauhiai/opetus/DIP/kuvankasittely2.pdf], 4.3.2012.

/67/ JISC Digital Media, Digital Cameras, [kuva], [http://www.jiscdigitalmedia.ac.uk/stillimages/advice/digital-cameras], 6.1.2012.

/68/ Joentakanen Kuisma, Pygame-peliohjelmointi, [pdf], [https://publications.theseus.fi/bitstream/handle/10024/21239/kuisma_joentakanen.pdf?sequence=1], 11.3.2012.

K

/69/ Kainulainen Antti, Agile-menetelmät, Ylempi AMK-tutkinto, Jyväskylän AMK, [pdf], [https://publications.theseus.fi/bitstream/handle/10024/17857/jamk_1214986335_4.pdf?sequence=2], 4.3.2012.

/70/ Kalimo Anna, Graaffisen käyttöliittymän suunnittelu, TIEKE RY, 1995.

/71/ Kallio Aleksi, C++:n hämmästyttävä maailma, [WWW-dokumentti], [ftp://nic.funet.fi/index/c++opas/yleista.html], 11.3.2012.

/72/ Kemppainen Teemu, Hyyti Heikki, Sjöberg Olli, Salonen Sami, Sjöberg Timo, Kannari Lotta, Kenttä- ja palvelurobotiikan harjoitustyö, [pdf], [http://autsys.tkk.fi/fsr/attach/ProjectWork/2010-group1.pdf], 4.3.2012.

/73/ Kuntz Noah, Tutorials, [WWW-dokumentti], [http://www.pages.drexel.edu/~nk752/tutorials.html], 3.3.2012

/74/ Kuutti K., Rajanen M., Johdatus GUI-suunnitteluun, [pdf], [http://www.tol.oulu.fi/kurssit/kayttoliittymat/GUI-05_osa1.PDF], 4.3.2012.

/75/ Kuutti K., Rajanen M., Harjoitustyöohje, [pdf], [http://www.tol.oulu.fi/kurssit/kayttoliittymat/GUI-05_osa2.PDF], 4.3.2012.

/76/ Kuutti K., Rajanen M., Talon sisäisen tyylioppaan tekeminen, [pdf], [http://www.tol.oulu.fi/kurssit/kayttoliittymat/GUI-05_osa3.PDF], 4.3.2012.

L

/77/ Laaksonen Jukka, Digitaalinen kuvankäsittely, luentokalvot, [pdf], [http://www.cis.hut.fi/Opinnot/T-61.247/edita2002/kalvot.pdf], 3.3.2012.

/78/ Laganiere Robert , OpenCV 2 Computer Vision Application Programming Cookbook, 2011.

/79/ Laine Mikko, Askelmoottorit, [WWW-dokumentti], [http://www.ele.tut.fi/teaching/ele-3350/askelmoottorit.pdf], 27.2.2012.

/80/ Laitetekniikka, Terävyysalue, [WWW-dokumentti], [http://www.laitetekniikka.com/digikuvaus/teravyysalue.html], 28.2.2012.

/81/ Laitetekniikka, Digikamera, [WWW-dokumentti], [http://www.laitetekniikka.com/digikuvaus/kamera.html], 4.3.2012.

/82/ Laitetekniikka, Polttovali, [WWW-dokumentti], [http://www.laitetekniikka.com/digikuvaus/polttovali.html], 4.3.2012.

/83/ Leinonen Jukka, Projekti-insinööri, Optisenmittauksen laboratorio, Kemi-Tornion AMK, TKI, 15.2.2012.

/84/ LifeCam Studio, Support, [WWW-dokumentti], [http://www.microsoft.com/hardware/en-us/p/lifecam-studio/Q2F-00001#support], 7.3.2012.

/85/ LimCorp, Microsoft new LifeCam Studio with incredible 1080p HD sensor, [kuva], [http://limcorp.net/2010/microsoft-lifecam-high-hd-sensor], 4.3.2012.

/86/ Linux aktivaattori, Python (opas), [WWW-dokumentti], [http://www.l-a.fi/Python_(opas)], 4.3.2012.

/87/ Linux.fi, Tiedoston oikeudet, [WWW-dokumentti],[http://linux.fi/wiki/Chmod], 1.3.2012.

/88/ Litwiller Dave, CCD vs. CMOS: Facts and Fictions, [pdf], [http://www.teledynedalsa.com/public/corp/Photonics_Spectra_CCDvsCMOS_Litwiller.pdf], 6.1.2012.

/89/ Litwiller Dave, CMOS vs. CCD, [pdf], [http://www.teledynedalsa.com/public/corp/CCD_vs_CMOS_Litwiller_2005.pdf], 28.2.2012.

M

/90/ Machine design, Repas Roberts, Lighting the Way for Vision, [http://machinedesign.com/article/lighting-the-way-for-vision-0221], 4.3.2012.

/91/ Machine Vision Association of SME, Lighting and Optics Reference Guide, [pdf], [http://www.engatechvision.com/images/mvaposter.pdf], 3.3.2012

/92/ Microchip, Stepping Motors Fundamentals, [WWW-dokumentti], [http://ww1.microchip.com/downloads/en/AppNotes/00907a.pdf], 27.2.2012.

/93/ Microscan, Machine Vision Lighting Training, [pfd], [http://www.microscan.com/en-us/trainingandresources/lighting.aspx], 2012.

/94/ Microscan, NERLITE Machine Vision Lighting Products Quick Reference, [pdf], [http://files.microscan.com/_att/3a9b342f-f8b6-4ec6-a566-239bcd388a4e/NERLITE_ATEST.pdf], 4.3.2012.

/95/ Microscan, NERLITE Machine Vision Lighting Products Product Line Card, [pdf], [http://files.microscan.com/_att/43585a99-7a45-4992-b272-de36f1c46f5e/NERLITE_linecard.pdf], 4.3.2012.

/96/ Microscan, Bright Field and Dark Field Lighting, [WWW-dokumentti], [http://www.microscan.com/en-us/technology/machine%20vision%20lighting/brightfieldanddarkfieldlighting.aspx], 4.3.2012.

/97/ Midwest Optical Systems INC, [kuva], [http://www.machinevisionfilters.com/bp324.html], 4.3.2012.

/98/ MiniTec, profile 45 x 45 F, [kuva], [http://www.minitecframing.com/Products/Aluminum_Profiles/Aluminum_Profile_Catalog_Pages/20.1033_Aluminum_Profile_45x45F.html], 10.3.2012.

/99/ Molecular Expressions, Convolution Kernel Mask Operation, [WWW-dokumentti], [http://micro.magnet.fsu.edu/primer/java/digitalimaging/processing/kernelmaskoperation/], 4.3.2012.

/100/ Mountain goat software, An Overview of Scrum for Agile Software Development, [WWW-dokumentti], [http://www.mountaingoatsoftware.com/scrum/overview], 4.3.2012.

/101/ Mouser Electronics, PBGA, [kuva], [http://fi.mouser.com/ProductDetail/Freescale-Semiconductor/MPC8321VRAFDC/?qs=X%252b52Ci9YKLJn8jCQTXORtQ%3d%3d], 27.2.2012.

/102/ Mutanen Arto, Polttoväli, [WWW-dokumentti], [http://www.artomutanen.net/opas/Kamera/polttovaeli.html], 4.3.2012.

N

/103/ National Instruments, A Practical Guide to Machine Vision Lighting - Part I, [WWW-dokumentti], [http://zone.ni.com/devzone/cda/tut/p/id/6901], 4.3.2012.

/104/ National Instruments, A Practical Guide to Machine Vision Lighting - Part II, [WWW-dokumentti], [http://zone.ni.com/devzone/cda/tut/p/id/6902], 4.3.2012.

/105/ National Instruments, A Practical Guide to Machine Vision Lighting - Part III, [WWW-dokumentti], [http://zone.ni.com/devzone/cda/tut/p/id/6903], 4.3.2012.

/106/ National Instruments, Acquiring from GigE Vision Cameras with Vision Acquisition Software, [WWW-dokumentti], [http://zone.ni.com/devzone/cda/tut/p/id/5651], 4.3.2012

/107/ Nexlogic, BGA-packages,[WWW-dokumentti], [http://www.nexlogic.com/services/pcb-assembly/ball-grid-arrays/ball-grid-array-%28bga%29.aspx], 27.2.2012.

/108/ Nordic Electronics Packaging Guideline, The, Overview of the BGA Technology, [WWW-dokumentti], [http://extra.ivf.se/ngl/E-BGA/ChapterE1.htm], 27.2.2012.

O

/109/ Ohjelmointiputka, C++-ohjelmointi: Osa 1, [WWW-dokumentti], [http://www.ohjelmointiputka.net/oppaat/opas.php?tunnus=cpp_ohj_01], 11.3.2012.

/110/ OnSig, IC Package Design and Validation, [kuva], [http://www.onsig.com/produc8.gif], 27.2.2012.

/111/ OpenCV, v2.3 documentation, [WWW-dokumentti], [http://opencv.itseez.com/index.html], 27.2.2012.

/112/ OpenCV, Installation in Windows, [WWW-dokumentti], [http://opencv.itseez.com/doc/tutorials/introduction/windows_install/windows_install.html#windows-installation], 3.3.2012.

/113/ OpenCVWiki, OpenCV, [kuva], [http://opencv.willowgarage.com/wiki/], 27.2.2012.

/114/ OpenCV developer site, [WWW-dokumentti], [http://code.opencv.org/projects/opencv], 4.3.2012.

/115/ OpenCV 2.1 Python Reference, [wiki], [http://opencv.willowgarage.com/documentation/python/index.html], 4.3.2012.

/116/ OpenCV 2.3 Documentation, Installation in Linux, [WWW-dokumentti], [http://opencv.itseez.com/doc/tutorials/introduction/linux_install/linux_install.html#linux-installation], 3.3.2012.

/117/ Orbis, Konenäkö- Lyhyt oppimäärä, [WWW-dokumentti], [http://www.orbis.eu/Suomi/Yritys/Uutiset-ja-tapahtumat/Uutisarkisto/Konenako-lyhyt-oppimaara/], 1.3.2012.

/118/ Orbis, White Paper Baslerilta: Vertailu yleisimmistä kameroiden digitaalisista rajapinnoista, [WWW-dokumentti], [http://www.orbis.eu/default.asp?docId=20498], 4.3.2012.

P

/119/ Partanen Teemu, Suodattimet kuvanparantamisessa, [WWW-dokumentti], [http://www.niksula.cs.hut.fi/~tjpartan/Studio4/Harjoitukset/Harjoitus3/], 4.3.2012.

/120/ pc-based-versus-smart-camera, [kuva], [http://www.thomasnet.com/articles/image/pc-based-versus-smart-camera.jpg], 4.3.2012.

/121/ Pelimanni, Veli-Matti, Opinnäytetyö, Esitutkimus epäpuhtauspartikkeleiden tunnistamiseksi öljystä konenäön avulla, 2006.

/122/ Peltomäki Juha, Qt peruskurssi, [powerpoint], [http://www.slideshare.net/juhapeltomaki/qt-peruskurssi], 4.3.2012.

/123/ Peters Alan, Lectures on Image Processing, [WWW-dokumentti], [http://www.archive.org/details/Lectures_on_Image_Processing], 4.3.2012.

/124/ Photonics Media, Tips & Techniques: The Art and Science of Machine Vision Inspection, [kuva], [http://www.photonics.com/Article.aspx?AID=36134], 4.3.2012.

/125/ Pikseli, Järjestelmälliseksi, [pdf], [http://www.pikseli.fi/pdf/0305_jarjestelmalliseksi.pdf], 4.3.2012.

/126/ Pikseli, Sanasto, [WWW-dokumentti], [http://www.pikseli.fi/uusi/?id=127], 4.3.2012.

/127/ Pikseli, Objektiivien erilaisuus, [pdf], [http://www.pikseli.fi/pdf/0305_opiskat.pdf], 4.3.2012.

/128/ Pikseli, Terävyys alue, [pdf], [http://www.pikseli.fi/pdf/Pikseli_0106_6466.pdf], 4.3.2012.

/129/ Pikseli, Tee se itse, Valotuksen käsisäätö [pdf], [http://www.pikseli.fi/pdf/0405_tee_se_itse.pdf], 4.3.2012.

/130/ Pololu, Servo, servo motor, servomotor (definitely not server), [WWW-dokumentti], [http://www.pololu.com/blog/15/servo-servo-motor-servomotor-definitely-not-server], 27.2.2012.

/131/ Pololu, Micro Maestro 6-Channel USB Servo Controller(Assbemled), [kuva], [http://www.pololu.com/catalog/product/1350/pictures], 11.3.2012.

/132/ Pololu robotics & electronics, Micro Maestro 6-Channel USB Servo Controller (Assembled), [WWW-dokumentti], [http://www.pololu.com/catalog/product/1350], 4.3.2012.

/133/ Pylvänen Markus, Eclipse ja Open CV -kirjasto konenäkösovelluksen kehittämisen työvälineinä, kanditutkelma, [pdf], [http://users.jyu.fi/~mpylvai/eclipse_opencv/kandidaatintutkielma_MPy_070726.pdf], 3.3.2012.

/134/ Python.org, About, [WWW-dokumentti], [http://www.python.org/about/], 4.3.2012.

/135/ Python Wiki, Beginners guide, [WWW-dokumentti], [http://wiki.python.org/moin/BeginnersGuide/Overview], 4.3.2012.

Q

/136/ Qt Nokia, Cross-platform application and UI framework, [WWW-dokumentti], [http://qt.nokia.com/], 4.3.2012.

/137/ Qt Nokia, Signals & Slots, [WWW-dokumentti], [http://qt-project.org/doc/qt-4.8/signalsandslots.html], 4.3.2012.

R

/138/ Reaktor, Rajapintasuunnittelun perusteet, [WWW-dokumentti], [http://reaktor.fi/osaaminen/rajapintasuunnittelun_perusteet/], 4.3.2012.

/139/ Reaktor, Ketterät eli Agile-menetelmät, [WWW-dokumentti], [http://reaktor.fi/osaaminen/agile/], 4.3.2012.

/140/ Reaktor, Qt-sovelluskehys, [WWW-dokumentti], [http://reaktor.fi/osaaminen/qt-sovelluskehys/], 4.3.2012.

/141/ Robo Realm, Convolution Filter, [WWW-dokumentti], [http://www.roborealm.com/help/Convolution.php], 4.3.2012.

/142/ Robo Realm, Prewitt edge, [WWW-dokumentti], [http://www.roborealm.com/help/Prewitt.php], 4.3.2012.

/143/ Robo Realm, Sobel edge, [WWW-dokumentti], [http://www.roborealm.com/help/Sobel.php], 4.3.2012.

/144/ Rollco, Kuularuuvi, [WWW-dokumentti], [http://blog.rollco.fi/?tag=kuularuuvit], 27.2.2012.

S

/145/ Sanyo Denki, [kuva], [http://www.wexon.fi/UserFiles/File/Luettelo/2011-2012/Mekatroniikka-Osio6B.pdf], 27.2.2012.

/146/ Siemens Dematic, Surface Mount Technology Basics, Siemens, 2001.

/147/ SKS-mekaniikka, Thomson aktuaattorit, [WWW-dokumentti], [http://www.sks.fi/tuotteet/thomson_aktuaattorit],27.2.2012.

/148/ Smartvisionlights, IR Lighting for Machine Vision Lights, [kuva], [http://www.smartvisionlights.com/ir-lighting], 4.3.2012.

/149/ Sonninen Kalle, GRAAFISEN KÄYTTÖLIITTYMÄN SUUNNITTELU KÄYTETTÄVYYDEN NÄKÖKULMASTA, Opinnäytetyö, Turun ammattikorkeakoulu, [pdf], [https://publications.theseus.fi/bitstream/handle/10024/38032/Sonninen_Kalle.pdf?sequence=1], 4.3.2012.

/150/ Sony, [kuva], [http://www.sony.net/SonyInfo/News/Press/200806/08-069E/02.jpg], 4.3.2012.

/151/ Sony, [kuva], [http://www.sony.net/SonyInfo/News/Press/201201/12-009E/CMOS.jpg], 4.3.2012.

/152/ Sony, Sony Develops Next-generation Back-Illuminated CMOS Image Sensor which Embodies the Continuous Evolution of the Camera, [WWW-dokumentti], [http://www.sony.net/SonyInfo/News/Press/201201/12-009E/index.html], 4.3.2012.

/153/ Spread Spectrum Scene, Universal Serial Bus (USB) Information, [WWW-dokumentti], [http://sss-mag.com/usbp2.html#usb], 4.3.2012.

/154/ STG, IC packages, [WWW-dokumentti], [http://www.china-pcbassembly.com/ic-packages-n167-1.html], 27.2.2012.

/155/ Structuredlighting, [kuva], [http://www.looptechnology.com/graphics/Structuredlighting.gif], 4.3.2012.

T

/156/ Tampereen teknillisen yliopisto, Ketterät menetelmät, [WWW-dokumentti], [http://hlab.ee.tut.fi/hmopetus/vpsist-oppimateriaali/4-menetelmia-ja-malleja/4-4-ketterat-menetelmat], 4.3.2012.

/157/ Teledyne Dalsa, CCD vs. CMOS, [taulukko], [http://www.teledynedalsa.com/corp/markets/ccd_vs_cmos.aspx],1.3.2012.

/158/ Teledyne Dalsa, Image Sensor Architectures for Digital Cinemetography, [pdf], [http://www.teledynedalsa.com/public/dc/documents/Image_Sensor_Architecture_Whitepaper_Digital_Cinema_00218-00_03-70.pdf], 1.3.2012.

/159/ Test&Measurement World, CCD vs. CMOS image sensors, [WWW-dokumentti], [http://www.tmworld.com/article/512499-CCD_vs_CMOS_image_sensors.php], 1.3.2012.

/160/ Texas Instruments, Flip Chip Ball Grid Array Package Reference Guide, [pdf], [http://www.ti.com/lit/ug/spru811a/spru811a.pdf], 27.2.2012.

/161/ Tietonator, "Perinteinen" oliomenetelmä ketteräksi, [pdf], [http://www.pcuf.fi/sytyke/syysseminaarit/risteily2003/AgileSytyke2003.pdf], 4.3.2012.

/162/ Tikkanen Hannu, PADS Piirilevysuunnitteluopas II, ds-design systems OY, 2004.

/163/ Timonen Antti, Ladontakoneen tarkkuuden tutkiminen, opinnäytetyö, 2004.

/164/ Tractronics, Pick and Place Fiducial Requirements, [pdf], [http://www.tracktronics.com.au/pdf/PicknPlace_Fiducials.pdf], 24.1.2012.

/165/ Tuorla Observatory, Interpolointi, [WWW-dokumentti], [http://www.astro.utu.fi/zubi/math/interpol.htm], 1.3.2012.

U

/166/ Ubergizmo, Inside iphone4, [kuva], [http://www.ubergizmo.com/wp-content/uploads/2011/07/26-inside-iPhone-4.jpg], 4.3.2012

/167/ Ubuntu, CompilingEasyHowTo, [WWW-dokumentti], [https://help.ubuntu.com/community/CompilingEasyHowTo], 3.3.2012.

/168/ US's Navy's Electronics Manufacturing Center of Excellence, Tech Tips, [WWW-dokumentti], [http://www.empf.org/empfasis/2008/Nov08/tech_tips_1108.html], 24.1.2012.

V

/169/ Vaaraniemi Janne, C++ -kuvankäsittelyfunktiokirjasto, Opinnäytetyö, Kemi-Tornion Ammattikorkeakoulu, 2007.

/170/ VTT Valmistustekniikka Vuori Matti, Kivistö-Rahnasto Jouni,Toivonen Sirra, Hyvä käyttöliittymäsuunnittelu lähtee käytön tarpeista, [pdf], [http://www.vtt.fi/inf/julkaisut/muut/1998/av7-98.pdf], 4.3.2012.

/171/ Vision systems design, Intelligent cameras embed new low-cost functions, [WWW-dokumentti], [http://www.vision-systems.com/articles/print/volume-9/issue-8/features/product-focus/intelligent-cameras-embed-new-low-cost-functions.html], 1.3.2012.

/172/ Vision systems design, Novel Illumination,[WWW-dokumentti], [http://www.vision-systems.com/articles/2010/01/novel-illumination.html], 4.3.2012.

W

/173/ Webcam world, Webcam Hardware, [WWW-dokumentti], [http://www.webcamworld.com/setupawebcam/hardware.html], 4.3.2012.

/174/ Webconferencingcouncil, Web-kamera, [kuva], [http://webconferencingcouncil.com/wp-content/uploads/2009/05/webcam.jpg], 4.3.2012.

/175/ White Timothy, LeBlanc Steve, Diffuse on-axis light source Patent, [pdf], [http://www.google.com/patents/US5187611], 4.3.2012.

/176/ Wikipedia, Webcam, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Webcam], 4.3.2012.

/177/ Wikipedia, Ball Grid Array, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Ball_grid_array], 27.2.2012.

/178/ Wikipedia, Surface-mount technology, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Surface-mount_technology], 27.2.2012.

/179/ Wikipedia, SMT placement equipment, [WWW-dokumentti], [http://en.wikipedia.org/wiki/SMT_placement_equipment], 27.2.2012.

/180/ Wikipedia, Computer vision, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Computer_vision], 27.2.2012.

/181/ Wikipedia, Charge-coupled device, [kuva], [http://en.wikipedia.org/wiki/Charge-coupled_device],27.2.2012.

/182/ Wikipedia, Objektiivi, [WWW-dokumentti], [http://fi.wikipedia.org/wiki/Objektiivi], 4.3.2012.

/183/ Wikipedia, Valoherkkyys, [WWW-dokumentti], [http://fi.wikipedia.org/wiki/Valoherkkyys], 4.3.2012.

/184/ Wikipedia, Aukkosuhde, [WWW-dokumentti], [http://fi.wikipedia.org/wiki/Aukkosuhde], 4.3.2012.

/185/ Wikipedia, Interlaced video, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Interlaced_video], 4.3.2012.

/186/ Wikipedia, Progressive scan, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Progressive_scan], 4.3.2012.

/187/ Wikipedia, Rolling shutter, [kuva], [http://upload.wikimedia.org/wikipedia/commons/thumb/e/e4/CMOS_rolling_shutter_distortion.jpg/800px-CMOS_rolling_shutter_distortion.jpg], 4.3.2012.

/188/ Wikipedia, Rolling shutter, [kuva], [http://upload.wikimedia.org/wikipedia/commons/thumb/1/1d/Lightning_rolling_shutter.jpg/800px-Lightning_rolling_shutter.jpg], 4.3.2012.

/189/ Wikipedia, Normal lens, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Normal_lens], 4.3.2012.

/190/ Wikipedia, C++, [WWW-dokumentti], [http://en.wikipedia.org/wiki/C%2B%2B], 11.3.2012.

/191/ Wikipedia, Wide-angle lens, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Wide-angle_lens], 4.3.2012.

/192/ Wikipedia, Telephoto lens, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Telephoto_lens], 4.3.2012.

/193/ Wikipedia, Bayonet, [kuva], [http://upload.wikimedia.org/wikipedia/commons/c/c6/Bayonet-mount-01.svg], 4.3.2012.

/194/ Wikipedia, Normaali, [WWW-dokumentti], [http://fi.wikipedia.org/wiki/Normaali_%28matematiikka%29], 4.3.2012.

/195/ Wikipedia, Valo, [WWW-dokumentti], [http://fi.wikipedia.org/wiki/Valo], 4.3.2012.

/196/ Wikipedia, Fluorescence, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Fluorescence], 4.3.2012.

/197/ Wikipedia, Fosforesenssi, [WWW-dokumentti], [http://fi.wikipedia.org/wiki/Fosforesenssi], 4.3.2012.

/198/ Wikipedia, Python, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Python_(programming_language)], 4.3.2012.

/199/ Wikipedia, Software engineering, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Software_engineering], 4.3.2012.

/200/ Wikipedia, Xerox Alto, [WWW-dokumentti], [http://en.wikipedia.org/wiki/Xerox_Alto], 4.3.2012.

/201/ Wikipedia, SCRUM, [WWW-dokumentti], [http://fi.wikipedia.org/wiki/Scrum], 4.3.2012.

/202/ Wilen Mika, Tietokonenäkö havaitsevissa käyttöliittymissä, Opinnäytetyö, Tampereen ammattikorkeakoulu, [pdf], [https://publications.theseus.fi/bitstream/handle/10024/10013/Wilen.Mika.pdf?sequence=2], 4.3.2012.

Z

/203/ Zaber, Microstepping Tutorial, [WWW-dokumentti], [http://www.zaber.com/wiki/Tutorials/Microstepping], 27.2.2012.

MUUT

/204/ 1stvision, Gigabit Ethernet Cameras(GigE), [WWW-dokumentti], [http://www.1stvision.com/overviews/gige.htm], 4.3.2012.

/205/ 100fps, "What is Deinterlacing? Facts, solutions, examples.", [WWW-dokumentti], [http://www.100fps.com/], 4.3.2012.

/206/ Solarbotics, Stepper motor basics, [pdf], [http://www.solarbotics.net/library/pdflib/pdf/motorbas.pdf], 8.3.2012.

/207/ Micromo, Microstepping: myths and realities, [WWW-dokumentti], [http://www.micromo.com/microstepping-myths-and-realities.aspx], 8.3.2012.

/208/ Wexcon, Askelmoottorit, [WWW-dokumentti], [http://www.wexon.fi/sivu.php?id=86], 9.3.2012

/209/ Micromo, why steppers lose steps, [WWW-dokumentti], [http://www.micromo.com/why-stepper-motors-lose-steps.aspx], 9.3.2012.

/210/ Jauhiainen Jukka, Digitaalinen kuvankäsittely, Oulun seudun ammattikorkeakoulu, [pdf], [http://www.oamk.fi/~jjauhiai/opetus/DIP/kuvankasittely.pdf], 4.3.2012.

/211/ Pololu, GWS S35 STD Continuous Rotation Servo, [kuva], [http://www.pololu.com/catalog/product/948/specs], 9.3.2012.

/212/ Rikula Pauli, ohjelmistosuunnittelija, 19.2.2012.

/213/ Robotzone, How do servos work? [WWW-dokumentti], [http://www.servocity.com/html/how_do_servos_work_.html], 12.3.2012.

8. LIITELUETTELO

Python 2.7 esimerkki

```
#-*- coding: cp1252 -*-
# Tämän liitteen tarkoituksena on toimia python esimerkkinä.

#tuodaan kirjastot
import cv,cv2, sys

#koetetaan, joskos saataisiin kamera alustettua ja löydettyä
try:
	#alustetaan
	pCapture =  cv.CaptureFromCAM(0)
	#mikäli ei saada alustettua, tulostetaan tieto ja poistutaan ohjelmasta
	if( not pCapture ):
		print("Ei voitu alustaa\n")
		sys.exit(0)

#mikäli tapahtuu virheitä, tulostetaan tiedot siitä
except Exception:
	print (sys.exc_info()[0])
	print (sys.exc_info()[1])
	print (sys.exc_info()[2])
	sys.exit(0)

#luodaan “window” niminen ikkuna ja sen koko määrätään automaattisesti
cv2.namedWindow('window',cv.CV_WINDOW_AUTOSIZE)

#kokeillaan laoittaa pääohjelmaa
try:
```

```
#toistetaan loputtomasti
 while(1):
        #kaapataan kuva ja haetaan se puskurista
        cv.GrabFrame(pCapture)
        pVideoFrame = cv.RetrieveFrame(pCapture)

        #mikäli ei ole kuvaa, tulostetaan tieto ja poistutaan ohjelmasta
        if(not pVideoFrame):
                print("epaonnistui\n")
                sys.exit(0)

        #luodaan värikuva ja harmaasävykuva, alustetaan kuvat mustiksi
        image = cv.CreateImage(cv.GetSize (pVideoFrame), 8, 3)
        gray = cv.CreateImage(cv.GetSize (pVideoFrame), 8, 1)

        cv.SetZero(gray)
        cv.SetZero(image)

        #kloonataan kaapattu kuva luotuun värikuvaan
        image  = cv.CloneImage(pVideoFrame)

        #muunnetaan värikuva harmaasävykuvaksi
        cv.CvtColor(image, gray, cv.CV_RGB2GRAY)

        #tehdään kynnystys
        cv.Threshold(gray, gray, 250, 255, cv.CV_THRESH_BINARY)
        #näytetään kynnystetty kuva ikkunassa
        cv.ShowImage("window", gray)
```

```
#odotetaan näppäinkomentoa 40ms, eli sekunnissa tulee 25 kuvaa

        key = cv.WaitKey(40)

#mikäli esc-näppäintä painetaan tuhotaan ikkuna ja poistutaan ikisilmukasta, jolloin
#ohjelma loppuu. Ubuntussa nurmlock:n ollessa päällä se sai arvoksi 1048603.
        if(key==27)
                cv2.destroyWindow("window")
                break

#mikäli ei jostain syystä saada ikisilmukkaa toimimaan tulostetaan virhe
except:
        print (sys.exc_info()[0])
        print (sys.exc_info()[1])
        print (sys.exc_info()[2])
        sys.exit(0)
```

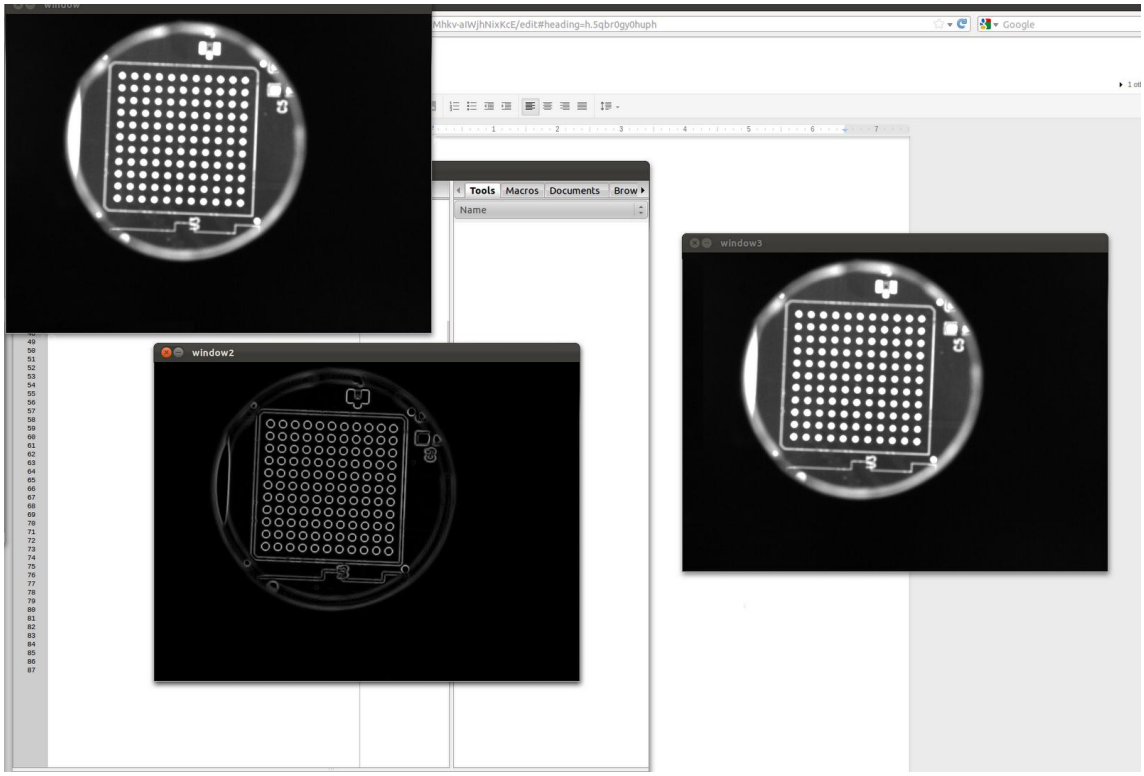
Python 2.7:n pohja

```
#-*- coding: cp1252 -*-
#Tämän tiedoston tarkoituksena on toimia pohjana mahdollisille töille

#tuodaan kirjastot
import sys, cv, cv2

#tulostetaan varuksi konsoliin aloitusteksti
print("start")

#0 on default camera
pCapture =  cv.CaptureFromCAM(0)
print(pCapture)

#Luodaan alkuperäisen kuvan kokoinen ikkuna
cv2.namedWindow('window',cv.CV_WINDOW_AUTOSIZE)

#Mikäli kameraa ei löydy tulostetaan viesti ja poistutaan ohjelmasta
if( not pCapture ):

        print("failed to initialize video capture\n")
        sys.exit(0)

#koetetaan pääohjelmaa
try:
        #while true
        while(1):
        #haetaan kuva kamerasta
        #voidaan käyttää ”helpompaa” QueryFramea tai grap-retrive
```

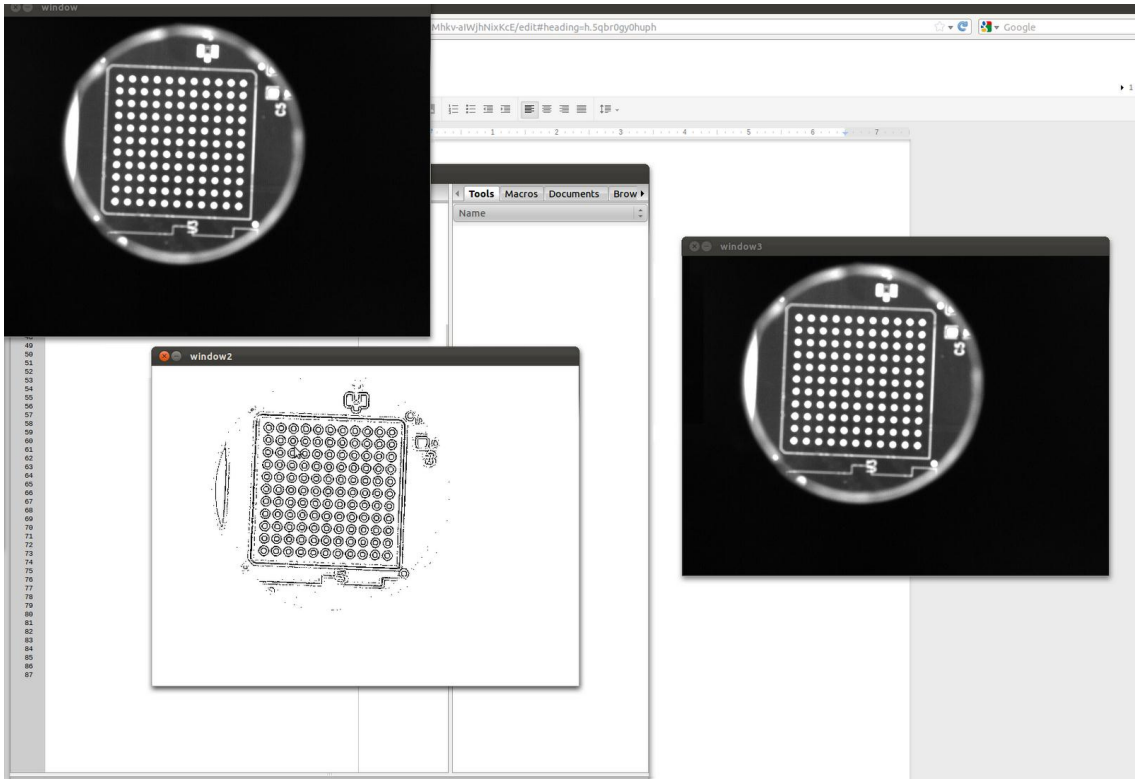
```
        #pVideoFrame = cv.QueryFrame(pCapture)
        cv.GrabFrame(pCapture)
        pVideoFrame = cv.RetrieveFrame(pCapture)

        #Luodaan  pVideoFrame:n kokoinen kuva, 8bit and 1 channel
        gray = cv.CreateImage(cv.GetSize (pVideoFrame),8 , 1)

        #tehdään kuva mustaksi
        cv.SetZero(gray)

        #kopioidaan kuva
        cv.Copy(pVideoFrame, gray)

        #näytetään kuva ikkunassa
        cv.ShowImage("window", gray)

        #odotetaan näppäinkomentoa 50ms
        key = cv.WaitKey(50)

        #jos näppäin on ”esc” niin poistutaan ohjelmasta ja tuhotaan ikkuna
        if (key == 1048603): # tai ==27 riippuen ympäristöstä
                print("die")
                cv2.destroyWindow("window")
                break
#Jos jotain menee vikaan, niin tulostetaan syitä.
except:
        print (sys.exc_info()[0])
        print (sys.exc_info()[1])
        print (sys.exc_info()[2])
```

Kuvia Pythonilla tehdyistä kuvankäsittelyistä

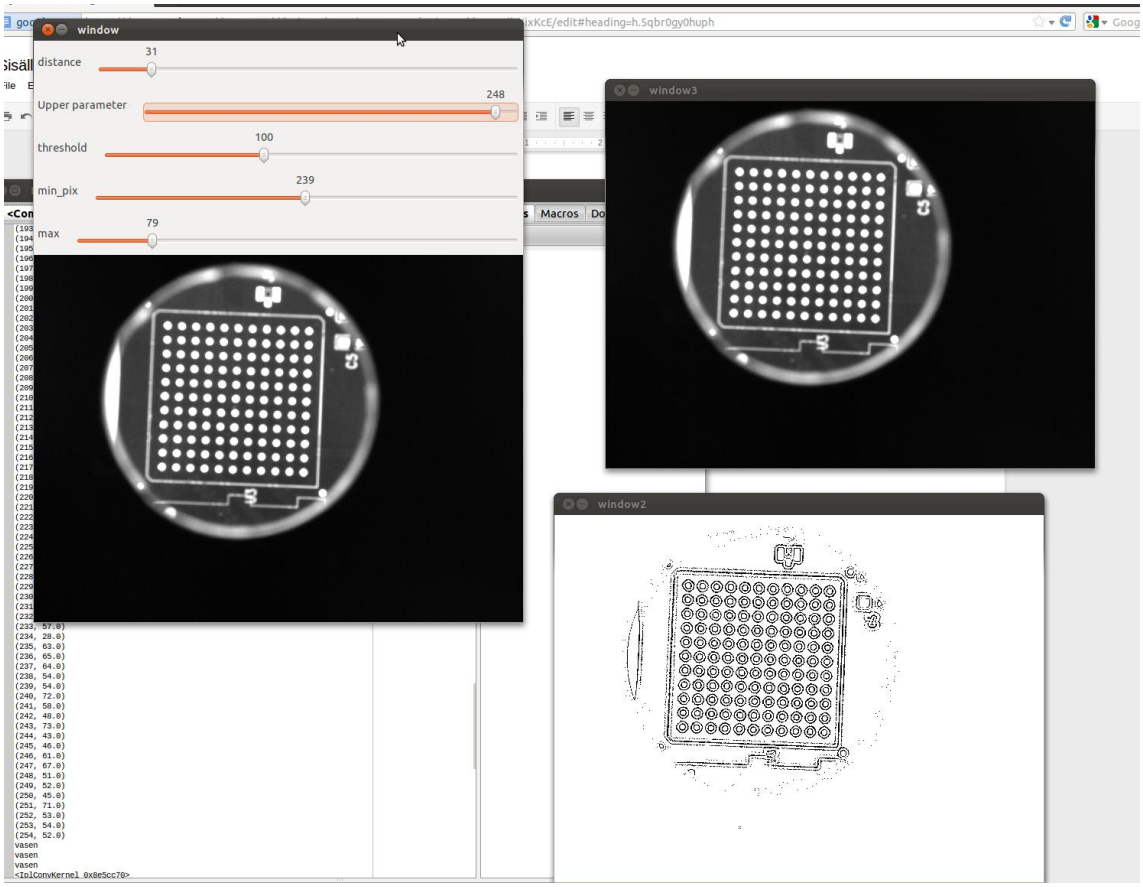
Tässä liitteessä on esitetty kuvankaappauksia erillaisista kuvankäsittelyistä, mitä testattiin Pythonilla. Kuvassa 143 on morphologinen muunnos, kuvassa 144 threshold eli kynnystys, kuvassa 145 OpenCV:n omat liukukytkimet sekä kuvassa 146 HoughCircles.



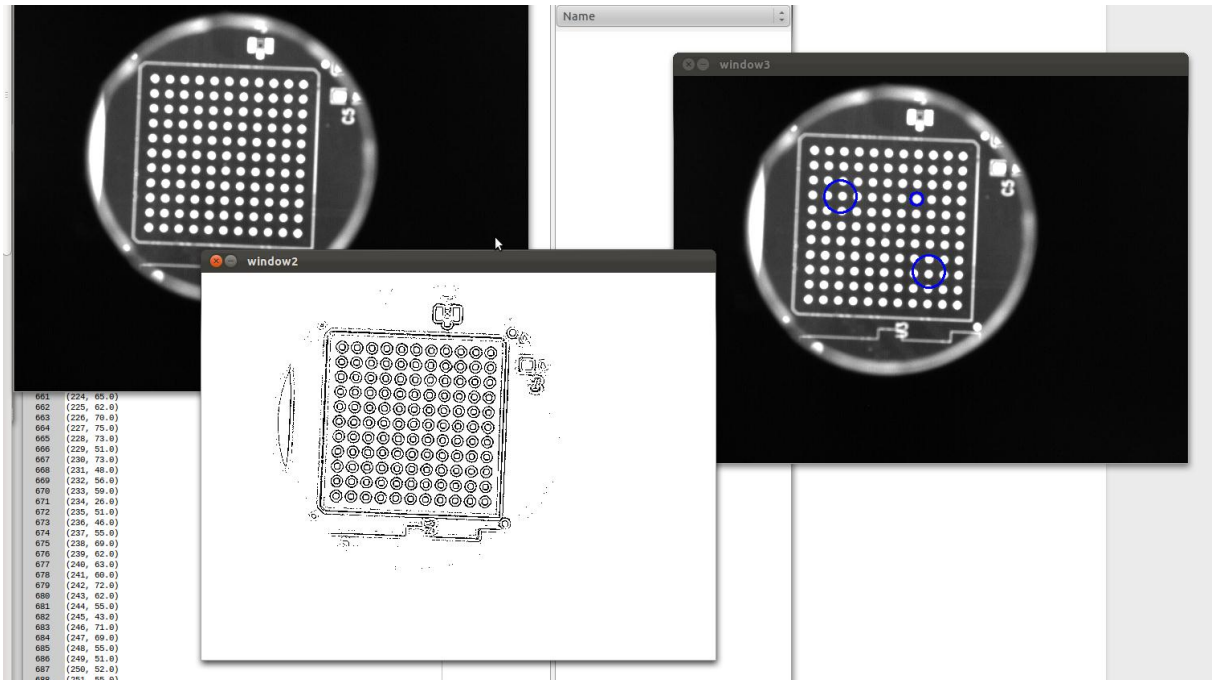
Kuva 143. Morphologi muunnos



Kuva 144. Threshold eli kynnystys



Kuva 145. OpenCV:n liukukytkimet



Kuva 146. Pythonilla testattu HoughCircles löytää yhden ympyrän

Qt esimerkkejä

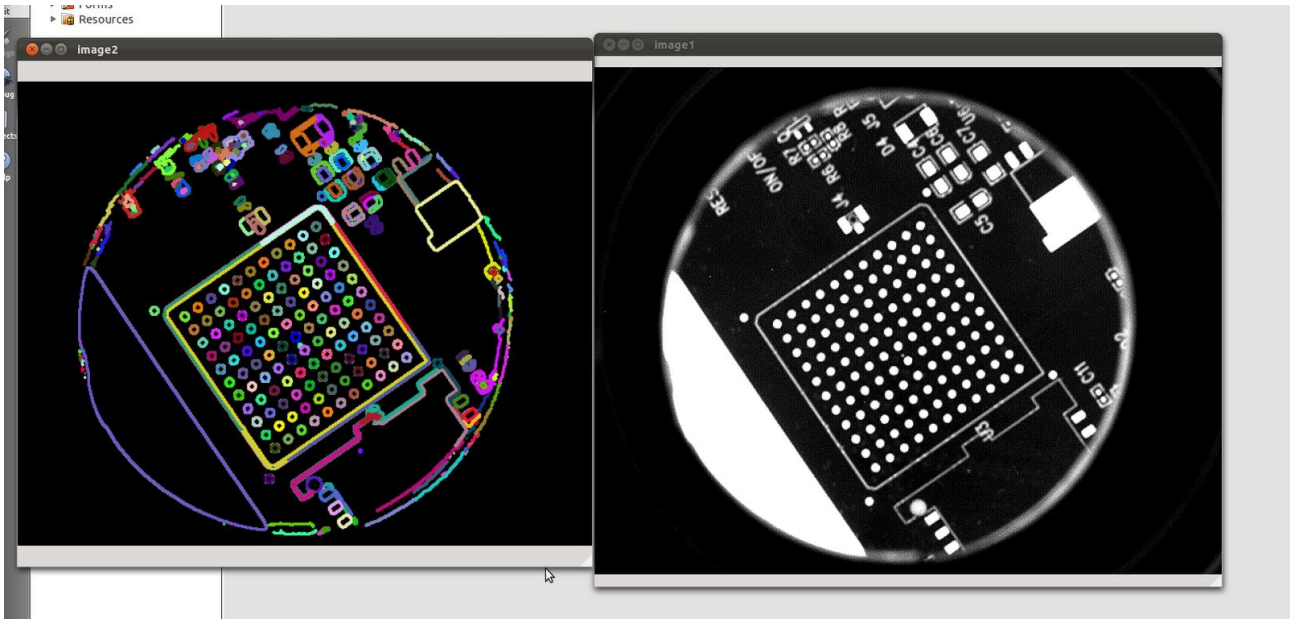
Tässä liitteessä esitellään Qt:llä toteutettuja esimerkkejä. Kuvassa 147 Canny-algoritmi, 148 sobel,

```
void MainWindow::on_pushButton_clicked() // avaa kuvan tiedostosta
{
	QString fileName = QFileDialog::getOpenFileName(this, "open Image",".",
	 "Image Files(*.png *.jpg *.jpeg *bmp)");

	image=cv::imread(fileName.toAscii().data());
	cv::namedWindow("image1",CV_WINDOW_FREERATIO );
	cv::imshow("image1",image);
}

void MainWindow::on_pushButton_8_clicked() //tallentaa “image”:ssa olevan kuvan. // kuvan perään täytyy kirjoittaa tyyppi, esim “*.png”
{
	QString fileName = QFileDialog::getSaveFileName(this, "Save File");
	cv::imwrite(fileName.toAscii().data(),image);
}

void MainWindow::on_pushButton_2_clicked()// muuntaa kuvan harmaasävyiseksi
{
	//muunnetaan kuva harmaakuvaksi
	cv::cvtColor(image, image, CV_BGR2GRAY);
	cv::namedWindow("image1",CV_WINDOW_FREERATIO );
	cv::imshow("image1",image);
}
```

```
void MainWindow::on_pushButton_3_clicked() // vaihtaa kuvatyypin 8-bittiseksi
{
	double minVal, maxVal;
	cv::minMaxLoc(image, &minVal, &maxVal); //find minimum and maximum
	//intensities
	cv::Mat draw;
	image.convertTo(draw, CV_8U, 255.0/(maxVal - minVal), -minVal);

	cv::namedWindow("image1",CV_WINDOW_FREERATIO );
	cv::imshow("image1", draw);
}

void MainWindow::on_pushButton_4_clicked() //blur, sumentaa kuvaa, alipäästösuodatin
{
	//luodaan kerneli
cv::Mat kernel(3,3,CV_32F,cv::Scalar(0));
	kernel.at<float>(1,1) = 5.0;
	kernel.at<float>(0,1) = -1.0;
	kernel.at<float>(2,1) = -1.0;
	kernel.at<float>(1,0) = -1.0;
	kernel.at<float>(1,2) = -1.0;

	cv::filter2D(image,image,image.depth(),kernel);

	cv::namedWindow("image1",CV_WINDOW_FREERATIO );
	cv::imshow("image1", image);
}
```

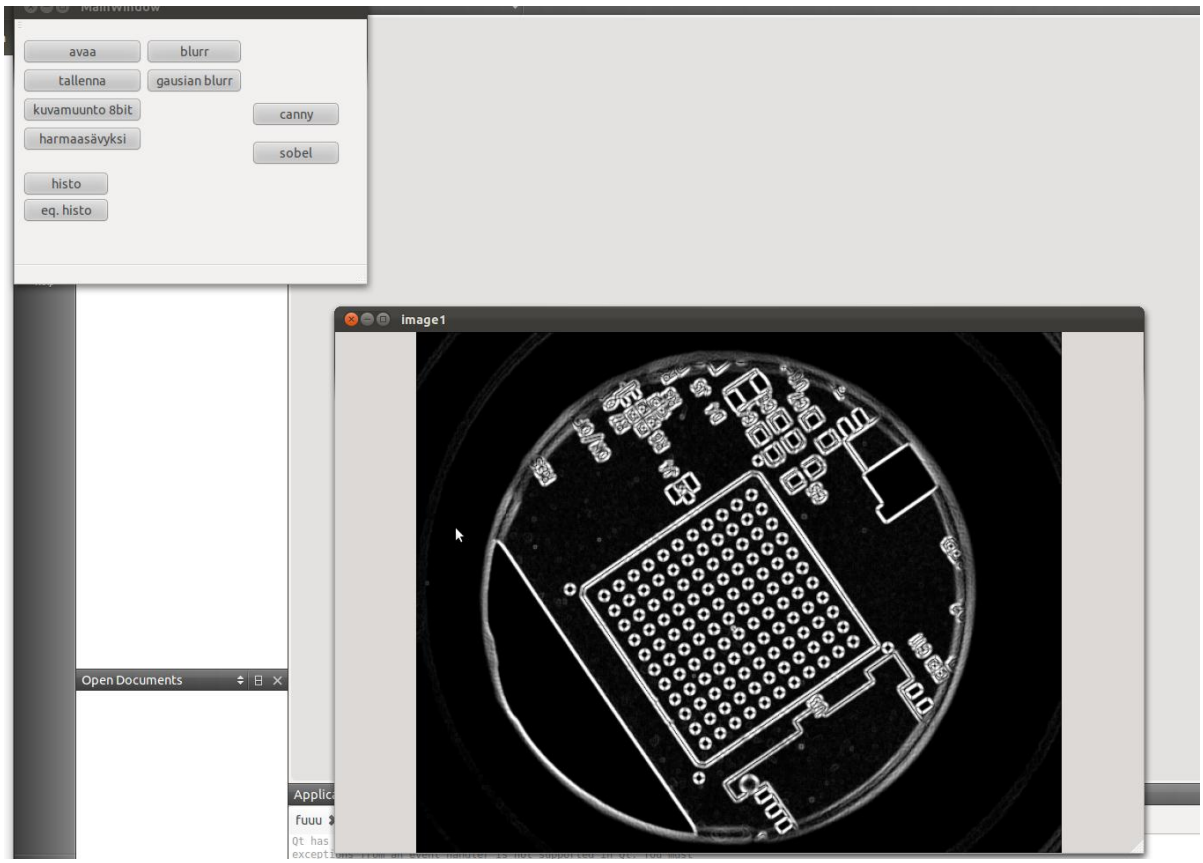
```
void MainWindow::on_pushButton_6_clicked()// gaussian-blur, sumentaa kuvan
{
  cv::GaussianBlur( image, image, cv::Size(3,3), 0, 0, cv::BORDER_DEFAULT );
  cv::namedWindow("image1", CV_WINDOW_FREERATIO );
  cv::imshow("image1", image);
}

void MainWindow::on_pushButton_5_clicked() //canny funktio, piirtää reunat satunnaisilla
//väreillä
{
        //create image and vectors
        cv::Mat xxx;

        std::vector<std::vector<cv::Point> > contours;
        std::vector<cv::Vec4i> hierarchy;

        //convert to gray image
        cv::cvtColor(image, image, CV_BGR2GRAY);

        //blur and canny
        cv::blur(image, image, cv::Size(3,3));
        cv::Canny(image, xxx, 100, 200, 3);

        //etintä
cv::findContours( xxx, contours, hierarchy, CV_RETR_TREE,
CV_CHAIN_APPROX_SIMPLE, cv::Point(0, 0) );
```

```
//piirto
cv::Mat drawing= cv::Mat::zeros(xxx.size(), CV_8UC3);

//luodaan satunnaislukugeneraattori
QTime time = QTime::currentTime();
qsrand((uint)time.msec());

//( qrand() % ((255 + 1) - 1) + 1) tämä arpoo pikselin luvut 255:n asti
for( int i = 0; i< contours.size(); i++ )
{
cv::Scalar color = cv::Scalar(( qrand() % ((255 + 1) - 1) + 1),
( qrand() % ((255 + 1) - 1) + 1),( qrand() % ((255 + 1) - 1) + 1));
cv::drawContours( drawing, contours, i, color, 2, 8, hierarchy, 0, cv::Point() );
}
cv::namedWindow("image2", CV_WINDOW_FREERATIO );
cv::imshow("image2", drawing);
}
```



Kuva 147. Canny-algoritmi ja satunnaiset värit

```
void MainWindow::on_pushButton_7_clicked()//Sobel-funktio
{
        cv::Mat src_gray;
        cv::Mat grad;

        cv::Mat grad_x, grad_y;
        cv::Mat abs_grad_x, abs_grad_y;

        cv::GaussianBlur( image, image, cv::Size(3,3), 0, 0, cv::BORDER_DEFAULT );
        cv::cvtColor(image, src_gray, CV_BGR2GRAY);

        /// Gradient X
        cv::Sobel( src_gray, grad_x, CV_16S, 1, 0, 3, 1, 0, cv::BORDER_DEFAULT );
        cv::convertScaleAbs( grad_x, abs_grad_x );
        /// Gradient Y
        cv::Sobel( src_gray, grad_y, CV_16S, 0, 1, 3, 1, 0, cv::BORDER_DEFAULT );
        cv::convertScaleAbs( grad_y, abs_grad_y );

        cv::addWeighted( abs_grad_x, 0.5, abs_grad_y, 0.5, 0, image );

        cv::namedWindow("image1", CV_WINDOW_FREERATIO );
        cv::imshow("image1", image);

}
```



Kuva 148. Sobel

```
//Värikanavien histogrammi. Lasketaan jokaisesta kanavasta oma histogrammi.
void MainWindow::on_pushButton_9_clicked()
{
        cv::Mat histo;
        /// Separate the image in 3 places ( R, G and B )
        std::vector<cv::Mat> rgb_planes;
        cv::split(image, rgb_planes);

        /// Establish the number of bins
        int histSize = 255;
        /// Set the ranges ( for R,G,B) )
        float range[] = { 0, 255 } ;
```

```
const float* histRange = { range };
bool uniform = true; bool accumulate = false;
cv::Mat r_hist, g_hist, b_hist;

cv::calcHist( &rgb_planes[0], 1, 0, cv::Mat(), r_hist, 1, &histSize,
 &histRange, uniform,
accumulate );
cv::calcHist( &rgb_planes[1], 1, 0, cv::Mat(), g_hist, 1, &histSize,
 &histRange, uniform,
accumulate );
cv::calcHist( &rgb_planes[2], 1, 0, cv::Mat(), b_hist, 1, &histSize,
 &histRange, uniform,
accumulate );
int hist_w = 400; int hist_h = 400;
int bin_w = cvRound( (double) hist_w/histSize );

cv::Mat histImage( hist_w, hist_h, CV_8UC3, cv::Scalar( 0,0,0) );

/// Normalize the result to [ 0, histImage.rows ]
cv::normalize(r_hist, r_hist, 0, histImage.rows, cv::NORM_MINMAX, -1, cv::Mat() );
cv::normalize(g_hist, g_hist,0, histImage.rows, cv::NORM_MINMAX, -1, cv::Mat() );
cv::normalize(b_hist, b_hist, 0, histImage.rows, cv::NORM_MINMAX, -1, cv::Mat()
);
```

```
/// Draw for each channel
        for( int i = 1; i < histSize; i++ )
        {
        cv::line( histImage, cv::Point( bin_w*(i-1), hist_h - cvRound(r_hist.at<float>(i-1)) ) ,
                cv::Point( bin_w*(i), hist_h - cvRound(r_hist.at<float>(i)) ),
                cv::Scalar( 0, 0, 255), 2, 8, 0  );

        cv::line( histImage, cv::Point( bin_w*(i-1), hist_h - cvRound(g_hist.at<float>(i-1)) ) ,
                cv::Point( bin_w*(i), hist_h - cvRound(g_hist.at<float>(i)) ),
                cv::Scalar( 0, 255, 0), 2, 8, 0  );
        cv::line( histImage, cv::Point( bin_w*(i-1), hist_h - cvRound(b_hist.at<float>(i-1)) ) ,
                cv::Point( bin_w*(i), hist_h - cvRound(b_hist.at<float>(i)) ),
                cv::Scalar( 255, 0, 0), 2, 8, 0  );
        }

        cv::namedWindow("image3", CV_WINDOW_FREERATIO );
        cv::imshow("image3", histImage);

}
```

```
void MainWindow::on_pushButton_10_clicked()// Tasoitettu histogrammi
// Tasoittaa sävyjen jakaumaa.
{
	cv::Mat histo;

	std::vector<cv::Mat> rgb_planes;

	/// Establish the number of bins
	int histSize = 255;

	/// Set the ranges ( for R,G,B) )
	float range[] = { 0, 255 } ;
	const float* histRange = { range };

	bool uniform = true; bool accumulate = false;

	cvtColor( image, image, CV_BGR2GRAY );
	cv::equalizeHist( image, image );
	cv::Mat r_hist;
	cv::calcHist( &image, 1, 0, cv::Mat(), r_hist, 1, &histSize, &histRange,
	uniform, accumulate );

	int hist_w = 400; int hist_h = 400;
	int bin_w = cvRound( (double) hist_w/histSize );

	cv::Mat histImage( hist_w, hist_h, CV_8UC3, cv::Scalar( 0,0,0) );

	/// Normalize the result to [ 0, histImage.rows ]
	cv::normalize(r_hist, r_hist, 0, histImage.rows, cv::NORM_MINMAX, -1, cv::Mat() );
```

```
        /// Draw for each channel
        for( int i = 1; i < histSize; i++ )
        {
        cv::line( histImage, cv::Point( bin_w*(i-1), hist_h - cvRound(r_hist.at<float>(i-1)) ) ,
                cv::Point( bin_w*(i), hist_h - cvRound(r_hist.at<float>(i)) ),
                cv::Scalar( 0, 0, 255), 2, 8, 0  );
        }

        cv::namedWindow("image3", CV_WINDOW_FREERATIO );
        cv::namedWindow("image1", CV_WINDOW_FREERATIO );
        cv::imshow("image1", image);
        cv::imshow("image3", histImage);
}
```