Esko Ojala

Messusolun suunnittelu ja toteuttaminen

Opinnäytetyö
Kevät 2014
Tekniikan yksikkö
Automaatiotekniikka
Koneautomaatio

Seinäjoen ammattikorkeakoulu
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Koulutusohjelma: Automaatiotekniikka

Suuntautumisvaihtoehto: Koneautomaatio

Tekijä: Esko Ojala

Työn nimi: Messusolun suunnittelu ja toteuttaminen

Ohjaaja: Martti Lehtonen

Vuosi: 2014 Sivumäärä: 66

Työn tarkoituksena oli suunnitella ja valmistaa konenäköä ja robottia hyödyntävä messusolu Co-Automation Oy:lle Alihankinta 2013 -messuille Tampereelle. Messusolu sisältää ABB:n IRB120 -robotin ja Omron FQ2 -konenäköjärjestelmän.

Työssä käydään läpi messusolun suunnittelua ja mallintamista, sekä konenäön ja robotin ohjelmointia. Messusolun suunnittelun apuna käytettiin ABB RobotStudio-, Solid Edge ST2- sekä MTPro-sovelluksia. Teoriaosuudessa on kerrottu konenäön ja robottien toimintaperiaatteita ja käyttökohteita.

Työn tuloksena saatiin aikaan messusolu, joka esiteltiin Tampereella 24.-26.9.2013 järjestetyillä Alihankinta 2013 -messuilla Suupohjan yrittäjien yhteisosastolla.

Avainsanat: messusolu, konenäkö, robotiikka

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Automation Engineering

Specialisation: Machine Automation

Author: Esko Ojala

Title of thesis: Planning and making of the exhibition cell

Supervisor: Martti Lehtonen

Year: 2014 Number of pages: 66

The purpose of this thesis was to plan and produce an exhibition cell for Co-Automation Oy for Alihankinta 2013 exhibition. The cell includes the ABB IRB120 robot and the Omron FQ2 machine vision system.

The planning and modeling processes of the exhibition cell and the programming of the robot and machine vision system are gone through in this work. While planning the exhibition cell ABB RobotStudio-, Solid Edge ST2- and MTPro-applications were used. In the theory part the operating principles and applications of the computer vision and robots are handled. The result of the work was the exhibition cell which was demonstrated at Alihankinta 2013 exhibition in Tampere 24.-26.9.2013.

Keywords: exhibition cell, machine vision, robotics

SISÄLTÖ

KUVIO- JA TAULUKKOLUETTELO

KAAVIOLUETTELO

1 JOHDANTO

1.1 TYÖN TAUSTA

Opinnäytetyön kohdeyrityksenä on Co-Automation Oy. Yritys on Vaasassa sijaitseva sähkö- ja automaatioalan yritys.

Co-Automation Oy osallistui Alihankintamessuille 2013 Tampereella ja yritys tarvitsi messuille robottisolun, joka herättäisi yleisön huomion.

1.2 TYÖN TAVOITE

Opinnäytetyön tavoitteena on suunnitella ja valmistaa messusolu Co-Automation Oy:lle, sekä selvittää robotiikan ja konenäön toimintaa. Yritys halusi messusolun sisältävän ABB:n IRB120-robotin sekä Omronin konenäköjärjestelmän. Järjestelmän toiminnan on oltava selkeä ja täysin automaattinen sekä huomiota herättävä, jolloin sen esittely messuilla olisi helppoa.

1.3 TYÖN RAKENNE

Työ on jaettu kolmeen osioon, joista ensimmäinen sisältää johdantoa työn taustoista ja tavoitteista sekä yritysesittelyn. Toinen osio sisältää teoriaa roboteista ja konenäköjärjestelmistä. Kolmannessa osiossa esitellään messusolun suunnittelu ja rakennusvaiheet.

1.4 TIETOA YRITYKSESTÄ

CO-Automation Oy on Vaasassa sijaitseva 25 työntekijää työllistävä tuotantoautomaatioon, suunnittelupalveluihin sekä teollisuuden kunnossapitoon ja palveluihin erikoistunut yritys. Yritys on perustettu vuonna 2007 Sami Kiviojan ja Jani

Mäen toimesta. CO-Automation Oy:n liikevaihto oli vuonna 2013 noin 3 miljoonaa euroa. (Kivioja 2014.)

Co-Automation on auktorisoitu ensimmäisenä ABB:n ulkopuolisena yrityksenä Suomessa ABB:n uudessa Value Provider Program -ohjelmassa. ABB:n Value Provider Program -ohjelmaan kuuluvat yritykset ovat strategisia robottiyhteistyökumppaneita, jotka käyttävät ABB:n robotiikkaa toimittamissaan tuotantoautomaatioprojekteissa. (Co-Automation 2014.)

Co-Automation Oy:n tarjoamiin palveluihin kuuluvat tuotantoautomaatio, suunnittelupalvelut sekä teollisuuden palvelut (Co-Automation 2014).

Kuvio 1. Co-Automation Oy:n logo

1.4.1 TUOTANTOAUTOMAATIO

Co-Automation Oy voi toimittaa tuotantoautomaatiojärjestelmän kokonaisuudessaan suunnittelusta käyttöönottoon ja koulutukseen asti. Osaprojektina yritys voi toimittaa tuotantoautomaatiolinjan sähkö- ja ohjausjärjestelmän kokonaisuudessaan. Lisäksi yritys toteuttaa vanhojen koneiden sähkö- ja ohjausjärjestelmien uusimisia. (Co-Automation 2014.)

Tuotantoautomaatiopalveluun kuuluu projektointi, suunnittelu, ohjelmointi, keskusvalmistus, asennukset, käyttöönotto, koulutukset sekä huolto- ja korjauspalvelut. Projektien osana tai erikseen Co-Automation Oy toimittaa myös robotteja, konenäköjärjestelmiä, logiikoita, automaatiojärjestelmiä sekä kuljettimia. (Co-Automation 2014.)

1.4.2 SUUNNITTELUPALVELUT

Co-Automation Oy suorittaa prosessiteollisuuden, energia-alan ja tuotantoautomaation sähkö- ja automaatiosuunnittelua. Suunnittelupalveluihin kuuluvat sähkösuunnittelu, automaatiosuunnittelu, instrumentointisuunnittelu, automaatiojärjestelmien sovellussuunnittelu sekä logiikkasuunnittelu. Yritys suorittaa myös käyttöönotto- ja asennusvalvontapalveluita eri teollisuuden kohteissa. (Co-Automation 2011.)

Co-Automation Oy tarjoaa myös Windows SharePoint -alustan palveluita, joihin kuuluu muun muassa sivustojen, luetteloiden ja kirjastojen luontia ja muokkausta, lisäominaisuuksien rakentamista JavaScriptin avulla sekä muokkausta SharePoint Designerissa. (Co-Automation 2011.)

1.4.3 TEOLLISUUDEN PALVELUT

Co-Automation Oy suorittaa eri toimialojen kunnossapitotöitä teollisuudelle. Palveluihin kuuluu sähkö- ja automaatiolaitteiden kunnossapito sekä vianetsintä ja korjaus, sähkö- ja automaatiolaitteiden varaosatarpeiden määrittely, sähkö- ja automaatiolaitteiden huolto-ohjelman laatiminen, ohjausjärjestelmien uudelleen ohjelmointi ja ohjelmamuutokset sekä sähkölaitteistojen lämpökamerakuvaukset. (Co-Automation 2011.)

2 KONENÄKÖ

2.1 MIKÄ ON KONENÄKÖJÄRJESTELMÄ?

Konenäkö voidaan luokitella ihmisen näköaistia matkivaksi koneelliseksi aistimeksi (Soini 2011).

Ensimmäisiä konenäön sovelluksia suomessa käytettiin sahateollisuudessa tukkien halkaisijoiden mittaamiseen 1970-luvun alussa. Mittauksen jälkeen tukit esilajiteltiin sahausta varten. (Soini 2011.)

Konenäköjärjestelmän peruskomponentteja ja -toimintoja ovat kamera, kuvankäsittely, valaistus, mittausohjelmisto, ohjausjärjestelmä sekä käyttöliittymä (Voutilainen 2004).

Konenäköjärjestelmän toiminnan perusperiaatteena on, että kamera ottaa digitaalisia kuvia, jotka tallennetaan kameran muistiin. Kuvien kohdetta analysoidaan vertailemalla sitä laitteen muistissa olevaan ohjekuvaan. (Niemi 2014.)

Viimeisimpänä kehityssuuntana ovat älykamerat, joihin on sisäänrakennettu kaikki konenäössä tarvittavat elementit (Soini 2011).

2.2 KONENÄKÖJÄRJESTELMÄN KOMPONENTIT

Seuraavaksi esitellään konenäköjärjestelmän sisältämät komponentit.

2.2.1 KAMERA

Kamera on konenäköjärjestelmän ydinkomponentti, joka valitaan sovelluksen mukaan (Niemi 2011). Kamera kuvaa tuotetta jota mitataan ja se sisältää optiikkaa, jonka avulla tuotteesta heijastuva valo siirretään valoherkälle kennolle. Kenno muodostuu pikseleistä eli varausyksiköistä. Kameran tarkkuus määräytyy pikseleiden määrän perusteella. (Voutilainen 2004.)

Kameraa valittaessa liika tarkkuus ei ole järkevää, sillä tällöin prosessoritehoa hukkuu tarpeettoman kuvainformaation analysointiin. Suurin osa kameroista toimiikin vain 640 x 480 pikselin tarkkuudella, mutta tarjolla on myös tarkempia, jopa viiden megapikselin kameroita. (Niemi 2011.)

2.2.2 KUVANKÄSITTELY

Kun kuva on otettu, se siirtyy muistiin, josta sitä hyödynnetään kuvankäsittelyllä. Kuvankäsittely tuottaa tietoa mittausohjelmiston käyttöön mittausten suorittamista varten. Käsittelyn ansiosta siirrettävä informaatio vähenee ja mittausohjelmiston ja tiedonsiirron toiminta nopeutuu. (Voutilainen 2004.)

Kuvankäsittelyllä kuvasta etsitään informaatiota, joka järjestelmän toiminnan kannalta on tarpeellista. (Halinen 2007).

Kuvankäsittely voi tehdä kuvalle esimerkiksi seuraavia asioita:

- Kuvan kirkkauden säätäminen
- Kuvan värien säätäminen
- Erilaisten filttereiden eli suodattimien käyttäminen
- Kuvan kääntäminen tai siirtäminen kuvatun kappaleen asennon tai sijainnin perusteella
- Kuvan rajaaminen tietylle alueelle. (Omron 2012, 59-69.)

Kuvio 2. Harmaasävyfiltteriä voi käyttää esimerkiksi näyttämään kuvasta ainoastaan punaista väriä sisältävät alueet (Omron 2012, 69.)

2.2.3 VALAISTUS

Valaistusta pidetään konenäön kriittisimpänä tekijänä (Lempiäinen 2011). Virheettömien mittaustuloksien aikaansaamiseksi valaistuksen on oltava oikea, sillä se tuottaa olosuhteet, joissa kuvaaminen voi tapahtua (Voutilainen 2004). Kaikki prosessoitava informaatio on peräisin valosta (Hornberg 2008, 76).

Matemaattinen kuvankäsittely vaatii, että kohteen ja taustan kirkkaus eroavat toisistaan. Kontrastia, kirkkautta ja pimeyttä, varjoja, tekstuureja, heijastuksia ja viivoja tarvitaan ja kaikki nämä tuotetaan valon avulla. Systemaattinen oikean valaistuksen etsintä säästää aikaa, rahaa ja hermoja. (Hornberg 2008, 76.)

Kohteeseen vaikuttava valaistus tulee yleensä pitää muuttumattomana, jolloin kuvankäsittely helpottuu huomattavasti (Halinen 2007).

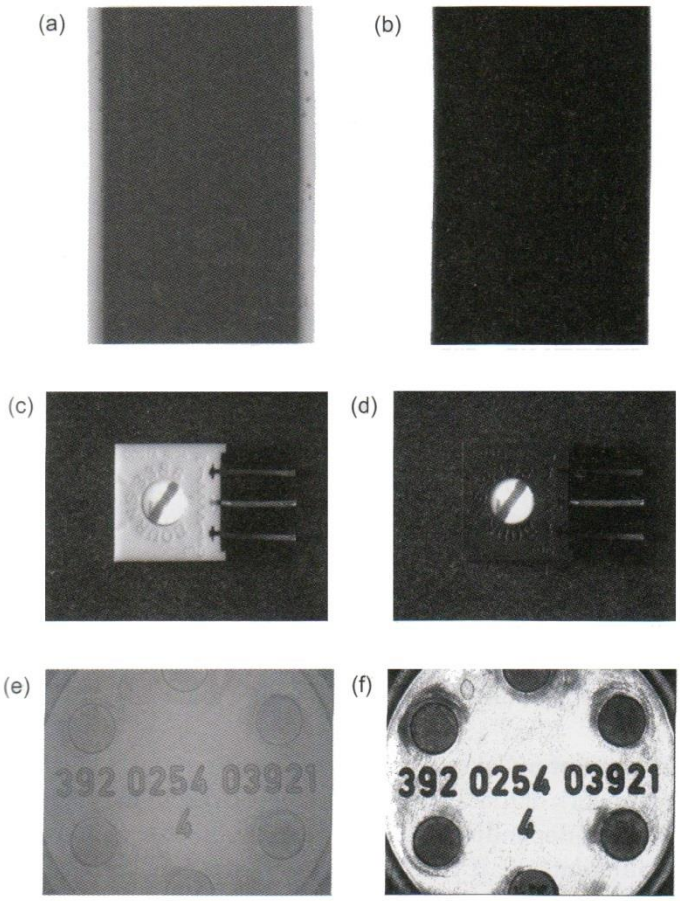
Yleisimpiä valonlähteitä ovat loisteputket, ledit, kaasupurkauslamput ja laservalo (Lempiäinen 2011).

Eri valaistusmenetelmiä ovat:

- Suora valo: Suunnatuilla valovoimaisilla lampuilla tuotetaan kirkas valo ja terävät varjot. Ei tuota tasaista valaistusta kohdepinnalle.

- Epäsuora valo: Valo heijastetaan toisen pinnan kautta kohdepinnalle. Saadaan aikaiseksi tasainen valo koko pinnalle ja vähemmän varjoja.
- Taustavalo: Heijastetaan kameraa kohti siten, että valaistava kappale on kameran ja valolähteen välissä. Saa kappaleen ulkoreunat hyvin esille.
- Salamavalo: Salamalla saadaan liikkuva kappale pysäytettyä ja estettyä muuten sumeaksi muodostuva kuva. (Halinen 2007.)
- Telesentrinen valo: Erikoistapaus suorasta valosta, jossa valonsäteet johdetaan linssin läpi. Linssi kääntää valonsäteet yhdensuuntaisiksi. Saa kappaleen pinnanmuodot hyvin esille. (Hornberg 2008, 156-157.)

Kuviosta 3 näkyy, miten eri valaistusmenetelmät vaikuttavat kuvaustulokseen: (a) Metallinen pultti epäsuoralla taustavalolla; (b) metallinen pultti telesentrisellä taustavalolla; (c) sininen potentiometri sinisessä valossa; (d) sininen potentiometri keltaisessa valossa; (e) korkki epäsuorassa valossa; (f) korkki suunnatussa valossa (Hornberg 2008, 75).



Kuvio 3. Erilaisia kappaleita eri valaistusmenetelmillä (Hornberg 2008, 75.)

2.2.4 MITTAUSOHJELMISTO

Mittausohjelmisto suorittaa ohjelman mukaiset matemaattiset laskutoimitukset kuvankäsittelyltä saatujen tietojen perusteella (Voutilainen 2004).

Mittausohjelmisto voi esimerkiksi mitata kappaleen pituutta, etsiä jotain tiettyä muotoa kuvasta tai mitata kappaleen pinta-alaa (Omron 2012, 90-91).

2.2.5 OHJAUSJÄRJESTELMÄ

Ohjausjärjestelmä hyödyntää mittaustiedon tuloksia. Ohjausjärjestelmässä päätökset tehdään mittaustulosten perusteella. Päätösten avulla suoritetaan tuotantoa ohjaavia toimenpiteitä. (Voutilainen 2004.)

2.2.6 KÄYTTÖLIITTYMÄ

Käyttöliittymän avulla hallitaan laitteiston toimintaa. Käyttöliittymän ohjelmisto sisältää toimenpiteet laitteiston seuraamiselle ja säätämiselle sekä raportoinnille. (Voutilainen 2004.)

2.3 KONENÄKÖJÄRJESTELMÄN KÄYTTÖKOHTEET

Seuraavaksi esitellään konenäköjärjestelmän erilaisia käyttökohteita.

2.3.1 YLEISET KÄYTTÖKOHTEET

Konenäköä käytetään tyypillisesti viallisten tuotteiden etsimiseen tuotantolinjalta. Konenäköä voidaan kuitenkin käyttää moneen muuhunkin tarkoitukseen, kuten päivämäärämerkintöjen, etiketin oikeellisuuden, 2D-koodien ja hitsaussaumojen tarkistamiseen sekä tuotantorobottien ohjaamiseen. (Niemi 2014.)

Yleisesti konenäköjärjestelmien sovellukset löytyvät erilaisten prosessien automatisoinnista, jolloin konenäköä käytetään ihmisoperaattorin tilalla vaikeissa, aikaa vievissä, yksitoikkoisissa, vaarallisissa tai mahdottomissa tehtävissä (Halinen 2007).

Konenäköä voidaan hyödyntää seuraavan luettelon mukaisiin tunnistus- ja tarkistustarkoituksiin. Luettelon eri alueita voidaan käyttää, ja usein käytetäänkin, eri sovelluksissa myös samanaikaisesti.

- **Koodin tunnistus:** Yleensä tunnistetaan standardisoituja viivakoodeja tai DataMatrix-koodeja, mutta voidaan myös tunnistaa kustomoituja koodeja.
- **Esineiden tunnistus:** Tunnistetaan esineiden piirteitä, kuten muotoa/geometriaa, mittoja, väriä, rakennetta/pinnanmuotoa tai tekstuuria.
- **Sijainnin tunnistus:** Tunnistetaan esineen tai esineessä sijaitsevan pisteen sijainti ja asento ennalta määritetyssä koordinaatistossa.
- **Kokoonpanon tarkistus:** Tarkistetaan onko kokoonpanossa kaikki tarvittavat osat, ja että ovatko osat oikeilla paikoillaan.

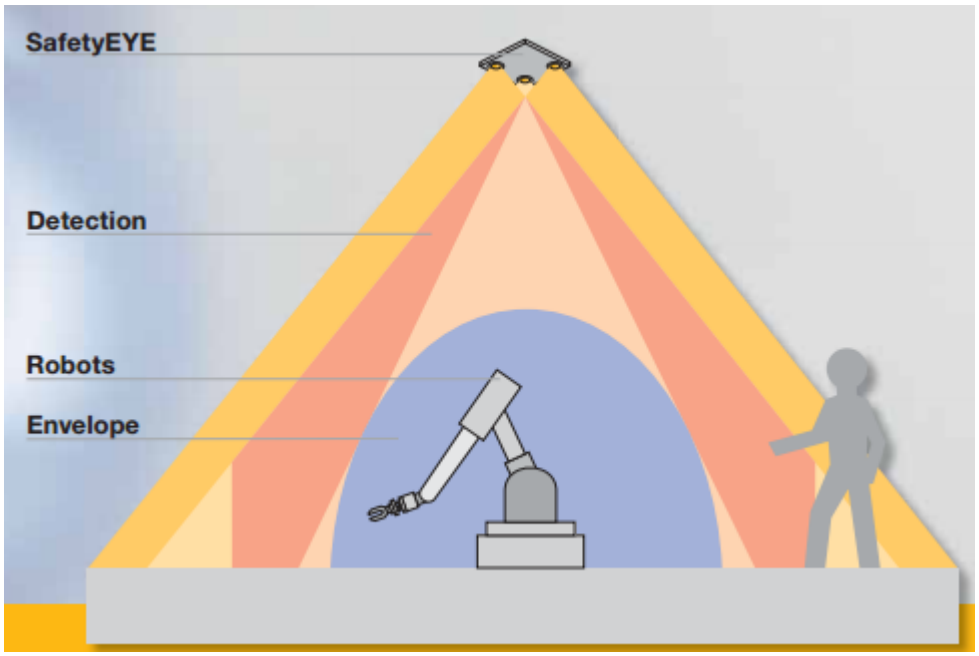
- **Muoto- ja mittatarkistus:** Mitataan tarkasti geometriset muodot ja mitat.
- **Pinnan tutkiminen:** Tutkitaan pinnan topografisia piirteitä, väriä tai tekstuuria. (Hornberg 2008, 695-696.)

2.3.2 KONENÄKÖ TURVALAITTEENA

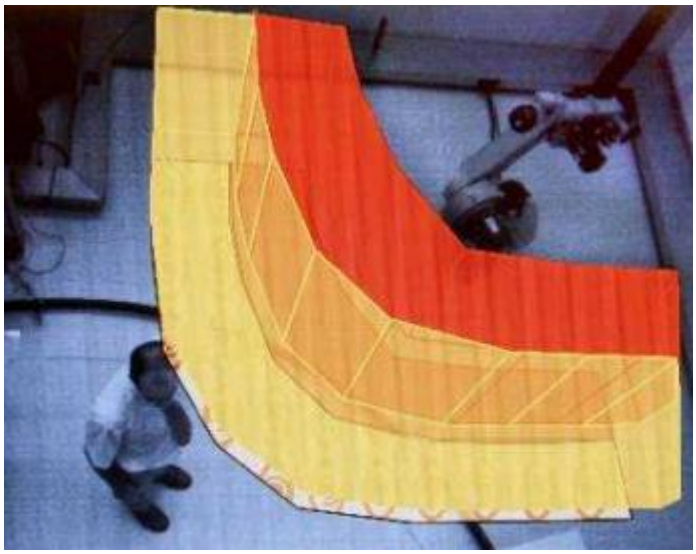
Konenäköön perustuvaa turvalaitetta ollaan standardisoimassa, mutta tekniikan nopean kehittymisen takia standardoitavan tuotteen määritys on kesken. Yhtenä turvalaitteiden perusperiaatteena on, että turvallisuus perustuu jatkuvan tiedon muuttumiseen. Kameroissa tämä periaate toimii monissa kohteissa: kamera antaa jatkuvasti kuvaa kohteesta ja muutoksiin reagoidaan. (Malm ym. 2008, 30)

Ensimmäisen 3D aluetta valvovan konenäköön perustuvan turvalaitteen, SafetyEYE:n, on valmistanut saksalainen turvalaitevalmistaja Pilz. SafetyEYE:n ohjelmistoon määritetään varoitus- ja pysäytysalue. Varoitusalueella voidaan esimerkiksi hidastaa robotin liikkeitä ja pysäytysalueella pysäyttää se kokonaan. (Pilz 2006.)

SafetyEYE-järjestelmä kehittyy nopeasti ja sille on odotettavissa kilpailijoita (Malm 2008).



Kuvio 4. SafetyEYE:n periaatekuva (Pilz 2006.)



Kuvio 5. Esimerkki SafetyEYE:n valvonta-alueista (Malm 2008)

3 ROBOTTI

3.1 MIKÄ ON ROBOTTI?

Robotti-käsite antaa yleensä oman mielikuvan jokaiselle ihmiselle. Mielikuvan on usein muokannut tieteiskirjallisuus, elokuvat tai televisiosarjat. Tekniikan ammattilaiselle robotti tuo usein mieleen teollisuusrobotin. Molemmat käsitykset ovat osaltaan oikeita, sillä robotti voi olla hyvin erilainen erilaisissa tehtävissä. (Keinänen, Kärkkäinen, Lähetkangas & Sumujärvi 2007, 259.)

Teollisuudessa robotteja käytetään pääasiassa kokoonpanossa, hitsauksessa, koneistuksen ja ruiskupuristuksen kappaleenkäsittelyssä, paketoinnissa sekä pakkauksessa. Robotteja käytetään yleisesti myös tehtävissä, jotka ovat ihmiselle liian raskaita tai vaarallisia. (Keinänen, Kärkkäinen, Lähetkangas & Sumujärvi 2007, 259.)

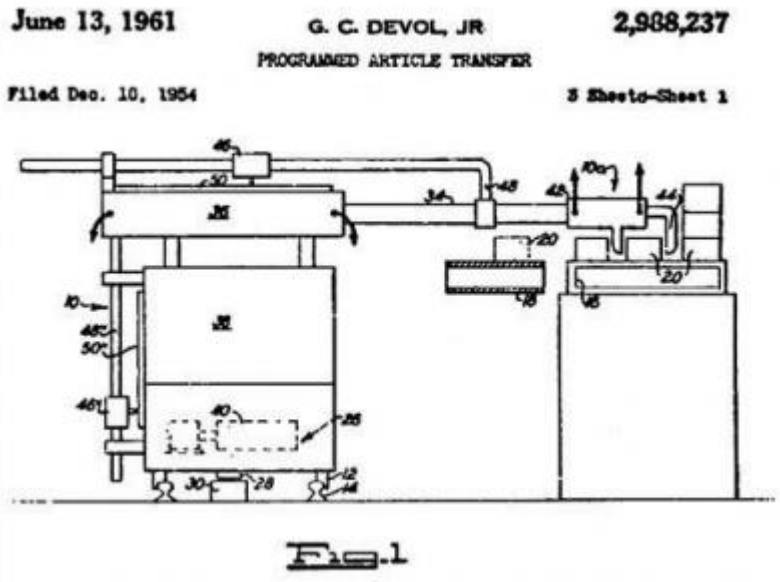
Teollisuusrobotin määritelmässä on maanosien ja eri lähteiden välillä pieniä eroja, eikä robotista ole yleisesti hyväksytty yhtä tiettyä kuvausta (Malm ym. 2008, 1). Robotin määritelmä kansainvälisen robottiyhdistyksen mukaan on seuraavanlainen: ”*robotti on uudelleenohjelmoitavissa oleva monipuolinen vähintään kolminivelinen mekaaninen laite, joka on suunniteltu liikuttamaan kappaleita, osia, työkaluja tai erikoislaitteita ohjelmoitavin liikkein monenlaisten tehtävien suorittamiseksi teollisuuden sovelluksissa*” (Aalto ym. 1999, 13).

3.2 TEOLLISUUSROBOTTI

Teollisuusrobotti on yksinkertaistettuna kone, joka siirtää työkalun kiinnityslaippaa halutulla tavalla. Robotin liikerata voidaan määritellä etukäteen, toimintaympäristön tapahtumien perusteella tai antureiden perusteella liikkeiden aikana. Robotti koostuu tukivarsista, jotka liittyvät yhteen nivelien avulla. Nivelet liikkuvat takaisinkytkettyjen servotoimilaitteiden avulla. (Aalto ym. 1999, 13.)

Nykyaikaisten teollisuusrobottien historia on alkanut vuonna 1954, jolloin amerikkalainen George C. Devol haki patenttia ohjelmoitavalle manipulaattorille, joka on

esitetty kuviossa 6. Teollisuusrobottien vallankumous alkoi 1961, kun Devolin ja Joseph F. Engelbergin kehittämä Unimate-robotti toimitettiin General Motorsin tehtaalle valukoneen avuksi. (Malm ym. 2008, 1.)

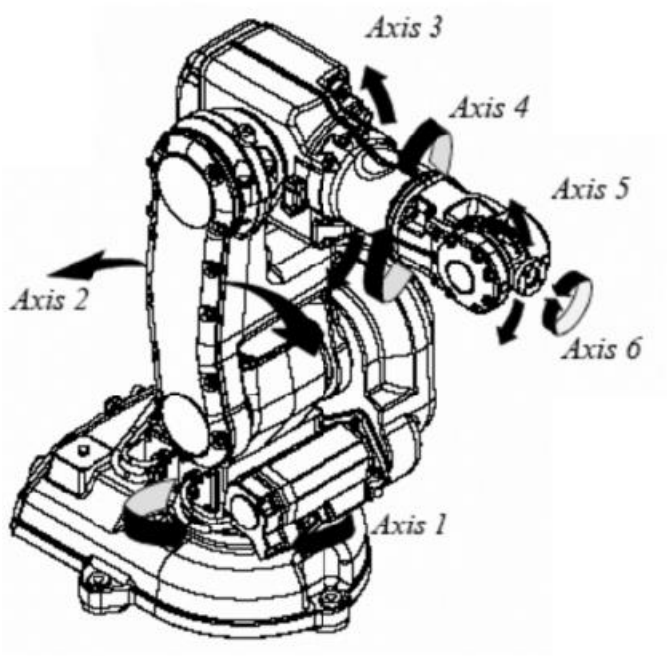


Kuvio 6. George Devolin ohjelmoitavan manipulaattorin patenttihakemus (Malm ym. 2008, 1.)

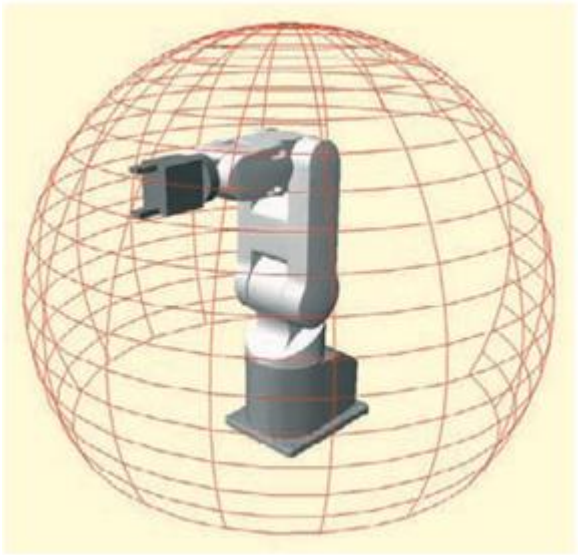
3.2.1 TEOLLISUUSROBOTIN RAKENNE

Teollisuusrobotti koostuu tukivarsista, joista kaksi liikkuu toistensa suhteen joko tietyn suoran suunnassa tai suoran ympäri. Tätä akselia kutsutaan niveleksi. Nivelten avulla tukivarret muuttavat keskinäisiä asentoja ja asemiaan. Niveliä kutsutaan vapausasteiksi (DOF, Degree of Freedom), jotka ovat kiertyviä tai suoria eli lineaarisia. Lähes kaikilla nykyaikaisilla teollisuusroboteilla on kuusi vapausastetta, joista vähintään kolme on kiertyviä. Kiertyvät nivelet ovat yleensä robotin ranteessa. Yhtä vapausastetta kohti on yleensä yksi toimilaite, esimerkiksi moottori tai sylinteri. (Aalto ym. 1999, 15.)

Kiertyvänivelisessä robotissa kaikki nivelet ovat kiertyviä ja ne ovat tavallisimpia teollisuusrobotteja (Aalto ym. 1999, 16). Kuviossa 7 on esitetty kiertyvänivelisen robotin akseleiden sijainnit ja niiden liikesuunnat.



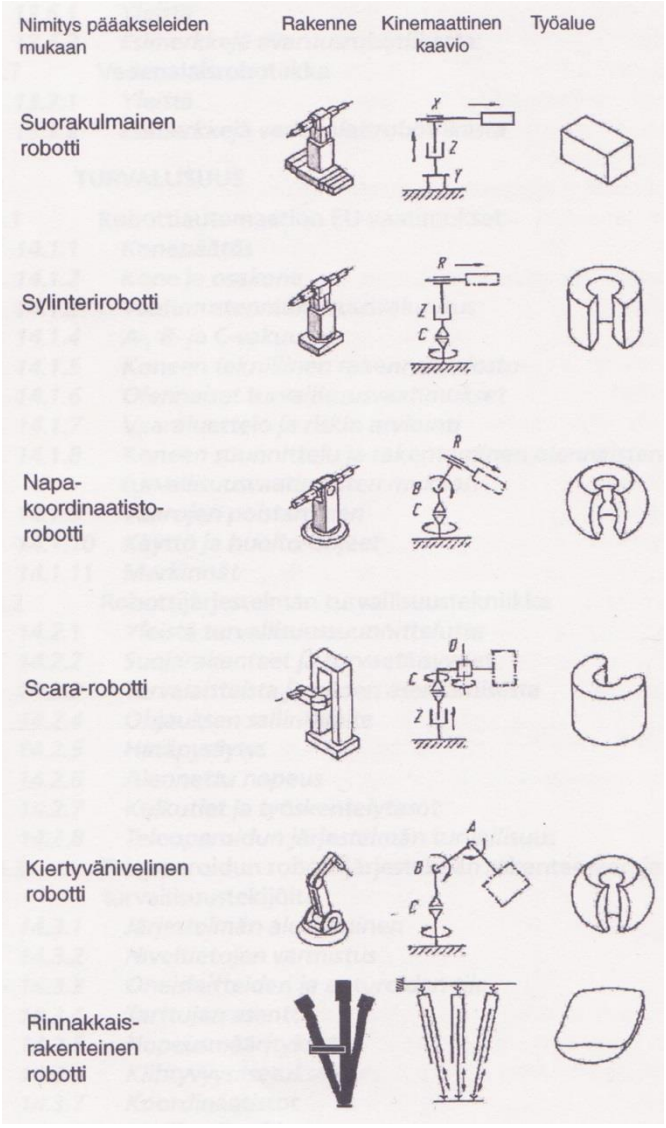
Kuvio 7. Kiertyvänivelisen robotin akselit eli vapausasteet (Lindholm 2011.)



Kuvio 8. Kiertyvänivelisen robotin työskentelyalue (Keinänen, Kärkkäinen, Lähetkangas & Sumujärvi 2007, 260.)

Erilaisia robottirakenteita saadaan kytkemällä robotin vapausasteita eri tavoin yhteen ja varioimalla niiden liikematkoja (Aalto ym. 1999, 16). Standardissa ISO

8373 on määritelty teollisuusrobottien sanastoa ja yleisimmät robottimallit mekaanisen rakenteen mukaan (Aalto ym. 1999, 12).



Kuvio 9. Erilaisten robottityyppien rakenne-esimerkkejä (Aalto ym. 1999, 12.)

3.3 ROBOTIN OHJAUS

Robotin ohjaamiseen käytetään robotin ohjainta ja käsiohjainta, jotka on esitetty seuraavaksi.

3.3.1 ROBOTIN OHJAIN

Robotin akselien moottorien virransyöttö ja liikkeiden ohjaus tapahtuvat robotin ohjaimessa. Kaikki robottiin ja sen ympäristöön kiinnitettyjen antureiden tilatiedot kulkevat ohjaimen kautta. Ohjaimessa ohjain tekee toimenpiteitä anturitietojen avulla. Tiedonvaihto muiden kokonaisjärjestelmässä olevien ohjausjärjestelmien ja tietokoneiden kanssa tapahtuu ohjaimessa. (Malm ym. 2008, 36.)

Robottien ohjausjärjestelmät ovat lähes reaaliaikaiseen toimintaan kykeneviä tehokkaita prosessitietokoneita. Ohjaimet ovat rakennettu tehdasolosuhteet huomioiden PC-arkkitehtuuriin ja komponentteihin perustuen. Ohjaimet ovat joustavia ja modulaarisia, mikä tekee mahdolliseksi saman ohjaintyypin käytön monen eri robotin ohjaimena. (Malm ym. 2008, 36.) Robottiohjaimet ovat tällä hetkellä valmistajakohtaisia, joten yleispätevää, kaikille roboteille sopivaa, ohjainta ei ole saatavilla (Keinänen, Kärkkäinen, Lähetkangas & Sumujärvi 2007, 261–262).

Kaikissa ohjaimissa käytetään TCP/IP-tiedonsiirtoprotokollaa. Useimmat robottiohjaimet ovat liitettävissä internetiin, mikä mahdollistaa tilatietojen ja asetusten muuttamisen paikasta riippumatta etäyhteyden avulla. (Malm ym. 2008, 36.)

Lähes kaikkien robottivalmistajien ohjaimilla voidaan ohjata robotin akseleiden lisäksi myös kahta ulkoista akselia. Ulkoisia akseleita voivat olla esimerkiksi kääntöpöytä tai lineaarirata. (Malm ym. 2008, 36.)

Kuvio 10. ABB IRC5 -robottiohjain (ABB 2013.)

3.3.2 KÄSIOHJAIN

Robottien käsiohjaimet ovat lähes poikkeuksetta värinäytöllä varustettuja Windows-pohjaisia pienoistietokoneita. Käsiohjaimet mahdollistavat muun muassa tehokkaan ohjelmoinnin ja ohjelmien muutoksien toteuttamisen. (Malm ym. 2008, 36.)

Esimerkiksi ABB IRC5 -ohjaimen käsiohjaimessa, FlexPendantissa (kuvio 11), käytetään Windows CE.NET -käyttöjärjestelmää. FlexPendantin kosketusnäyttö on suojattu vedeltä, kemikaaleilta ja hitsauksesta vahingossa lentäviltä roiskeilta. (Brorsson, Sjöberg & Liberg 2006, 59.)

Kuvio 11. ABB FlexPendant -käsiohjain (ABB 2013a.)

3.4 OHJELMOINTI

Robotin ohjelmointimenetelmät voidaan jakaa kahteen eri tapaan: on-line-ohjelmointi ja off-line-ohjelmointi. On-line-ohjelmoinnissa robottia ohjelmoidaan yleensä käsiohjaimella, joten robottia ei voida käyttää muuhun työhön samanaikaisesti. On-line-ohjelmointi voidaan suorittaa myös näyttämällä, jossa ei tarvita käsiohjainta, vaan robotin käsivartta liikutetaan lihasvoimin. Off-line-ohjelmoinnissa ohjelmaa tehdään ilman robottia, jolloin robotti voi tehdä samanaikaisesti muuta työtä. Off-line-ohjelmoinnin jälkeen tehdään yleensä kalibrointi ohjelmoidun ympäristön ja todellisen ympäristön välillä, jolloin ohjelmoidut pisteet osuvat tuotannossa oikeisiin kohtiin. (Malm ym. 2008, 95.)

Robottia voidaan ohjelmoida myös konenäköpohjaisesti, jolloin kohteesta otetaan ensin kuva, josta ohjelma laskee liikeradat. Konenäköpohjainen ohjelmointi soveltuu parhaiten kaksiulotteisiin liikkeisiin, kuten hitsaukseen. (Malm ym. 2008, 96.)

Käsiohjaimella ohjelmoitaessa voidaan robottia ohjelmoida opettamalla, oliopohjaisesti tai tekstipohjaisesti.

- Opettamalla ohjelmoinnissa robotti ajetaan haluttuun asemaan ja tallennetaan asema, jonka jälkeen siirrytään seuraavaan asemaan.
- Oliopohjaisessa ohjelmoinnissa käytetään käsiohjaimeen valmiiksi luotuja ikoneita.
- Tekstipohjaisessa ohjelmoinnissa ohjelmakoodi kirjoitetaan käsiohjaimella. (Malm ym. 2008, 95–96.)

Off-line-ohjelmointi voidaan toteuttaa tekstipohjaisena etäohjelmointina, oliopohjaisena etäohjelmointina, virtuaalisella käsiohjaimella, mallipohjaisena etäohjelmointina tai automaattisena etäohjelmointina.

- Tekstipohjaisessa etäohjelmoinnissa ohjelma kirjoitetaan ulkopuolisella tietokoneella. Tekstipohjainen ohjelmointi mahdollistaa monimutkaiset ohjelmarakenteet, kuten aliohjelmat.

- Oliopohjaisessa ohjelmoinnissa käytetään ulkopuolista tietokonetta ja valmiita ikoneita. Oliopohjaiseen ohjelmaan täytyy opettaa pisteet erikseen.
- Virtuaalinen käsiohjain on oikeaa käsiohjanta emuloiva PC-ohjelmisto, joten sillä ohjelmointi on samankaltaista, kuin käsiohjaimella ohjelmointi.
- Mallipohjaisessa etäohjelmoinnissa ohjelman tuottamiseen käytetään tietokoneen 3D-graafista käyttöliittymää ja robotin sekä oheislaitteiden simulointimalleja. Valmista ohjelmaa voidaan testata virtuaaliympäristössä.
- Automaattisessa etäohjelmoinnissa tietokoneohjelma analysoi sille annetun CAD-kuvan ja päättelee, mitä sille pitää tehdä. Ohjelma luo tarvittavat robotin liikkeet, jotka ihminen hyväksyy tai hylkää ja tekee tarvittavat lisäykset. Tästä eteenpäin jatketaan kuten mallipohjaisessa etäohjelmoinnissa. (Malm ym. 2008, 97–98.)

ABB:n roboteissa käytetään ohjelmointiin Rapid-ohjelmointikieltä. Rapid-ohjelma koostuu päärutiinista, alirutiinista ja ohjelmadatasta. (Keinänen, Kärkkäinen, Lähetkangas & Sumujärvi 2007, 262–263.)

3.5 ROBOTTIJÄRJESTELMÄN SIMULOINTI

Robottijärjestelmän suunnittelu simuloimalla mahdollistaa tuotantolaitteiston ominaisuuksien huomioonottamisen jo tuotteen suunnitteluvaiheessa. Tuotteen valmistusta voidaan kokeilla simulaattorilla jo ennen kuin yhtään fyysistä osaa on olemassa. (Aalto ym. 1999, 97–98.)

Simulointipohjaisella valmistuksen suunnittelulla voidaan yleensä lyhentää käyttöönottoaikaa. Parhaissa tapauksissa ensimmäinen robottiohjelman ajo suoritetaan tuotantokappaleelle. (Aalto ym. 1999, 98.)

Robottisimuloinnit voidaan jakaa kolmeen ryhmään: ulottuvuussimulointiin, prosessisimulointiin ja dynaamiseen simulointiin.

3.5.1 ULOTTUVUUSSIMULOINTI

Ulottuvuussimuloinnissa robottisolun ja tuotteen kaikki osat, sekä robotin kinematiikka mallinnetaan mahdollisimman tarkasti. Tärkeä osa ulottuvuussimulointia on myös törmäystarkastelu. Automaattinen törmäystarkastelu auttaa huomaamaan myös näkökentän ulkopuoliset törmäykset. (Aalto ym. 1999,99.)

Ulottuvuussimuloinnilla saadaan aikaiseksi robottisolun geometrinen ulottuvuustarkastelu, tuotteen ja robottisolun visualisointi sekä kiinnittimien soveltuvuuden testaaminen (Aalto ym. 1999, 99).

Ulottuvuussimuloinnin tärkeimpiä käyttökohteita ovat:

- robotin ja lisälaitteiden mitoittaminen niin, että tuotanto on mahdollista
- solun soveltuvuuden tarkastelu halutulle tuotteelle
- kiinnitinsuunnittelu. (Aalto ym. 1999, 99.)

3.5.2 PROSESSISIMULOINTI

Prosessisimuloinnissa malliin tarvitaan ulottuvuussimuloinnin lisäksi tarkka malli robotin ohjaimen toiminnasta. Prosessisimulointia käytetään etäohjelmoinnissa. (Aalto ym. 1999, 100.)

Prosessisimuloinnissa robotille luodaan tavanomaisia robottiohjelmia, jotka suorittavat tuotannonomaisia toimintoja. Ohjelmat ajetaan simuloidulla robotilla simuloidussa ympäristössä, jolloin ohjelman toimintaa voidaan tarkastella numeerisesti ja visuaalisesti. Simulaattoreissa voidaan mallintaa myös tuotantoprosessi, jolloin esimerkiksi työstössä simulaatiomallissa tapahtuu todellisuutta vastaava materiaalin poisto. (Aalto ym. 1999, 100.)

Prosessisimuloinnin tuloksia ovat:

- ohjelman toimivuuden testaus
- etäohjelmointi

- karkea tahtiaika-analyysi
- valmistettavuuden varmistaminen. (Aalto ym. 1999, 99.)

Prosessisimuloinnin käyttökohteita ovat:

- tuotannon vaatimien liikeratojen testaaminen
- törmäysvapaiden liikeratojen suunnittelu
- tuotanto-ohjelmien testaaminen
- tuotteen valmistettavuuden varmistaminen. (Aalto ym. 1999, 100.)

3.5.3 DYNAAMINEN SIMULOINTI

Dynaamisella simuloinnilla tarkoitetaan robottiin ja tuotantoprosessiin liittyvien voimien huomioon ottamista. Dynaamisen simuloinnin onnistuminen vaatii erityistä tarkkuutta robotin ohjaimen mallilta. Lisäksi on mallinnettava solun osien massat ja nivelten dynaamiset ominaisuudet. Dynaamisella simuloinnilla päästään hyvin suuriin tarkkuuksiin niin paikan, kuin ajankin suhteen. (Aalto ym. 1999, 100.)

Dynaamisella simuloinnilla saatavia tuloksia ovat:

- tarkkojen liikeratojen määritys
- tarkka tahtiaika-analyysi
- erittäin nopeita prosessiliikkeitä sisältävän ohjelman etäohjelmointi. (Aalto ym. 1999, 100.)

Dynaamisen simuloinnin käyttökohteita ovat:

- solun tahtiaikojen laskenta
- useiden robottien muodostaman solun liikkeiden synkronoinnin testaus
- suurnopeussolujen etäohjelmointi. (Aalto ym. 1999, 100.)

4 MESSUSOLUN SUUNNITTELU

4.1 TEHTÄVÄNANTO

Tehtävänä oli suunnitella ja rakentaa mahdollisimman näyttävä ja selkeä messusolu Tampereella järjestettäville alihankintamessuille. Aikaa solun suunnitteluun ja rakentamiseen oli noin puolitoista kuukautta. Solun vaatimuksina oli, että se sisältää ABB:n IRB120-robotin, sekä Omronin FQ2-konenäköjärjestelmän. Vaatimuksiin kuului myös, että solu mahtuu 700 mm x 1000 mm kokoiselle pöydälle. Muuten solun toiminta oli täysin opinnäytetyön tekijän päätettävissä.

Messusolun on tarkoitus mainostaa Co-Automation Oy:tä ja koska konenäköä käytetään teollisuudessa yhä enemmän robottien apuna, valittiin soluun robotti ja konenäköjärjestelmä.

4.2 SOLUN TOIMINNAN SUUNNITTELU

Messusolun suunnittelu aloitettiin ideoimalla erilaisia sovelluksia. Eri sovelluksideoista keskusteltiin Co-Automation Oy:n työntekijöiden kanssa ja valittiin paras toteutettavissa olevista sovelluksista.

Sovellukseksi valittiin järjestelmä, jossa konenäkö tunnistaa pelinoppien sijainnin ja silmäluvun. Tämän jälkeen robotti poimii nopat yksitellen ja lajittelee ne silmäluvun mukaan oikeaan putkeen, josta nopat putoavat kippoon. Lajittelun jälkeen robotti heittää nopat takaisin pöydälle. Tämä sovellus valittiin siksi, että sen toiminta on helppo ymmärtää, ja samaa periaatetta voidaan käyttää moneen eri tarkoitukseen.

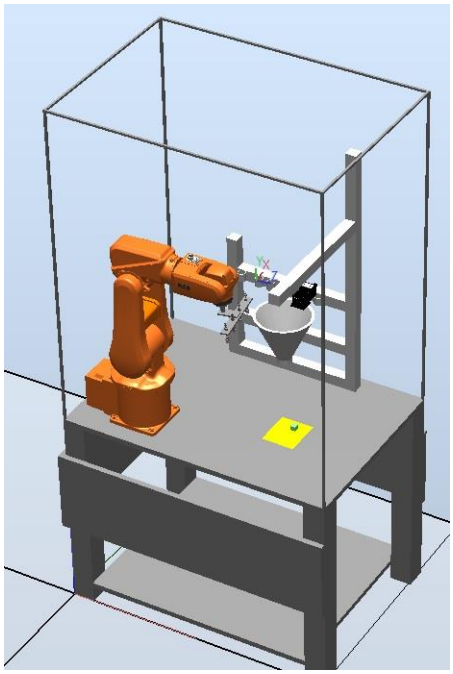
Solun toiminta pyrittiin suunnittelemaan niin, että se toimisi täysin itsenäisesti.

4.3 SOLUN SUUNNITTELU

Messusolun suunnittelu jatkui suunnittelemalla solun layout ja piirtämällä siitä 3D-malli. Malli piirrettiin ABB RobotStudio -ohjelmistolla, käyttäen apuna Solid Edge ST2 -ohjelmistoa, jolla mallinnettiin robotin ympäristön komponentit, sekä pöytä. Solid Edge -ohjelmistolla luodut mallit tuotiin RobotStudioon Iges-tiedostomuodossa. Muita ABB RobotStudion tukemia 3D-mallin tiedostomuotoja ovat:

- Acis
- Step
- Vdafs
- Catia
- Stl
- Vrml

Luomalla malli RobotStudioon saatiin selville, miten solulle varattu pöytätila tulee käyttää, jotta robotti yltää poimimaan nopat ja pudottamaan ne oikeaan paikkaan. Myös solun ohjelmointi helpottui huomattavasti layoutin ollessa valmiina. Solun 3D-layout on kuvattu kuviossa 12.



Kuvio 12. Messusolun layout

4.4 SOLUN SIMULOINTI

Ennen kuin solua alettiin rakentaa, se simuloitiin ja ohjelmoitiin ABB RobotStudiolla. Simuloinnin avulla voitiin liikuttaa robottia tietokoneen näytöllä, jolloin saatiin selville mahdolliset ongelmapaikat. Valmis ohjelma voitiin simuloinnin jälkeen siirtää suoraan robotille, jolloin tarvitsi ainoastaan kalibroida workobjektit vastaamaan oikeaa ympäristöä.

Seuraavissa kappaleissa käsitellään virtuaalisen solun luomista ja ohjelmointia.

4.4.1 VIRTUAALISEN ROBOTTIKONTROLLERIN LUOMINEN

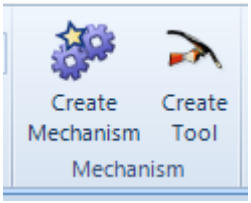
ABB RobotStudio -ohjelmistoon luotiin uusi virtuaalinen robottikontrolleri, joka sisälsi soluun tarvittavat optiot. Optiot ovat robotin kontrolleriin asennettavia lisäohjelmistoja, joiden avulla saadaan lisätoimintoja robottiin. Tarpeellisia optioita solun kannalta olivat PC Interface ja FlexPendant Interface. PC Interface -option avulla saadaan yhteys konenäköjärjestelmän ja robottikontrollerin välille. FlexPendant Interface mahdollistaa oman käyttöliittymän käyttämisen FlexPendant-paneelilla. Uuden kontrollerin ominaisuudet on esitetty kuviossa 13. Oikean robotin kontrollerissa täytyy myös olla samat optiot, mikä tulee ottaa huomioon robottia hankkiessa, sillä optioiden lisääminen jälkeenpäin on kallista.

```
<?xml version="1.0" encoding="UTF-8"?>
- <SystemProperties>
    <SystemName>Messurobotti</SystemName>
    <SerialNo>VIRTUAL_USE</SerialNo>
  - <UsedMedia>
      <Media version="5.15.1091.01" name="RobotWare" path="C:\PROGRAM FILES (X86)\ABB INDUSTRIAL IT\ROBOTICS IT\MEDIAPOOL\ROBOTWARE_5.15.1091"/>
    </UsedMedia>
  - <ControlModule>
      <Key>haKKKKRJJvTiCKKKKKKKKKh</Key>
      <SignatureNr>177</SignatureNr>
      <Category descr="OS">RobotWare OS and English</Category>
      <Category descr="Languages">644-11 Finnish</Category>
      <Category descr="Options">748-1 DeviceNet Lean</Category>
      <Category descr="Options">616-1 PC Interface</Category>
      <Category descr="Options">617-1 FlexPendant Interface</Category>
      <Category descr="Languages">645-8 Swedish</Category>
      <Option descr="RW Control module key"/>
      <Option descr="RobotWare OS and English"/>
    - <Option descr="644-11 Finnish">
        <SubOption descr="645-8 Swedish"/>
      </Option>
      <Option descr="748-1 DeviceNet Lean"/>
      <Option descr="616-1 PC Interface"/>
      <Option descr="617-1 FlexPendant Interface"/>
    </ControlModule>
  - <DriveModule>
      <Name>DriveModule1</Name>
      <Key>B0ZbnLDDpn</Key>
      <SignatureNr>170</SignatureNr>
      <Category descr="Manipulators">ABB standard manipulator</Category>
      <Category descr="Manipulators">Drive System 04 120/140/1400</Category>
      <Category descr="Manipulators">IRB 120</Category>
      <Category descr="Variants">120-0.58/3</Category>
      <Category descr="Drives">RC1 No add drive</Category>
      <Option descr="RW Drive module 1 key"/>
    - <Option descr="ABB standard manipulator">
      - <SubOption descr="IRB 120">
          <SubOption descr="120-0.58/3"/>
        </SubOption>
      </Option>
    - <Option descr="Drive System 04 120/140/1400">
        <SubOption descr="RC1 No add drive"/>
      </Option>
    </DriveModule>
</SystemProperties>
```

Kuvio 13. Virtuaalisen robottikontrollerin asetukset ja optiot

4.4.2 TYÖKALUN LUOMINEN

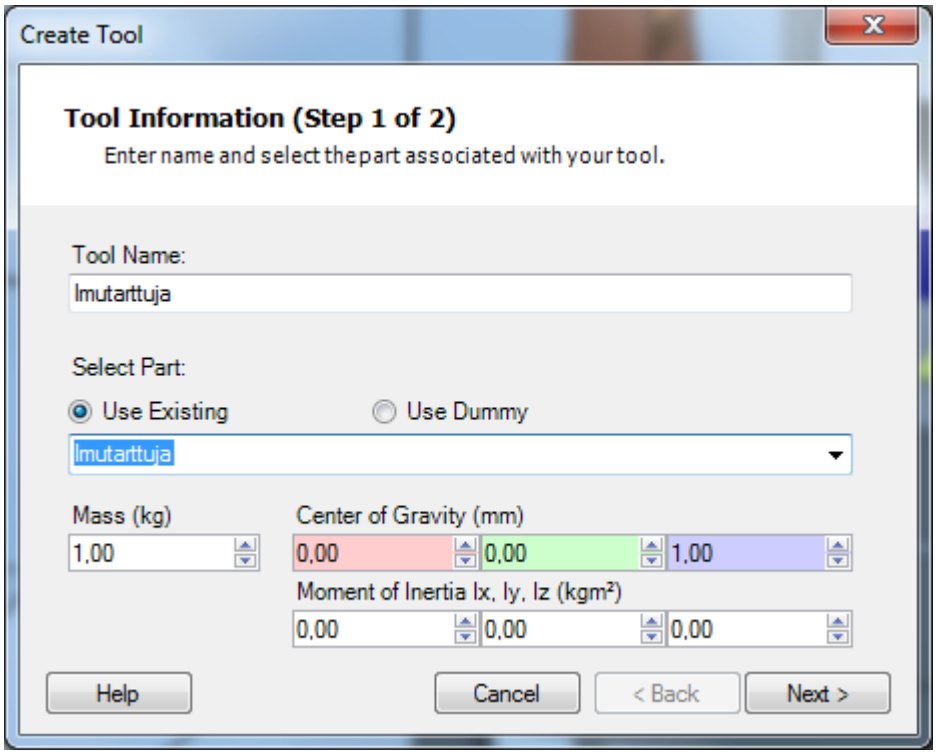
Työkalun malli on tuotu jo aikaisemmin robotin laippaan kiinni, joten se täytyi muuntaa työkaluksi, jonka työkalupisteitä robotin kontrolleri osaa ohjata. Työkalun luominen ABB RobotStudiolla tapahtuu Modeling-välilehdellä olevalla Create Tool -painikkeella (kuvio 14).



Kuvio 14. Create Tool -painike

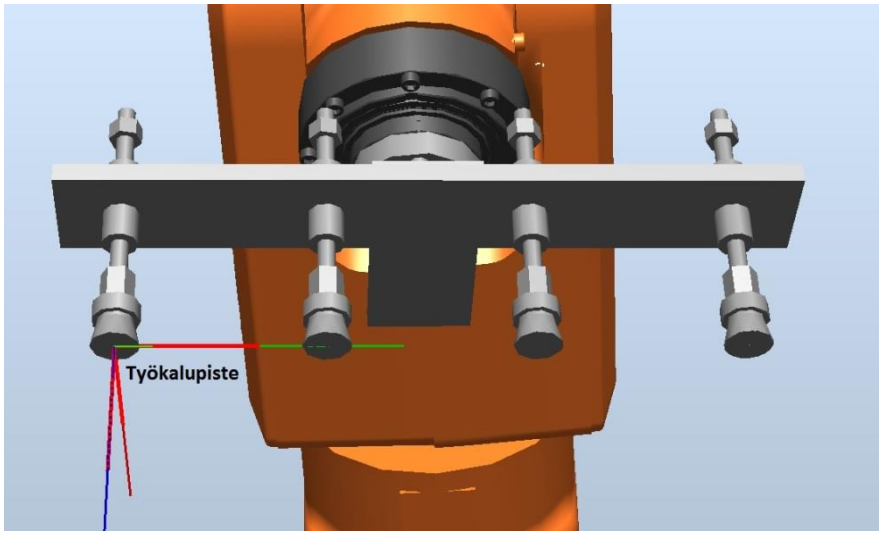
Painike avaa uuden ikkunan, jossa kysytään työkalun nimeä, massaa ja massakeskipistettä. Ikkunassa voidaan myös valita kappale, joka toimii työkaluna valit-

semalla Use Existing ja alla olevasta valikosta valitsemalla työkalun 3D-malli. Työssä valittiin aikaisemmin tuotu tarttujan 3D-malli työkaluksi ja annettiin työkalulle nimeksi imutarttuja. Massa ja massakeskipiste pidettiin alkuperäisen mukaisina (kuvio 15).

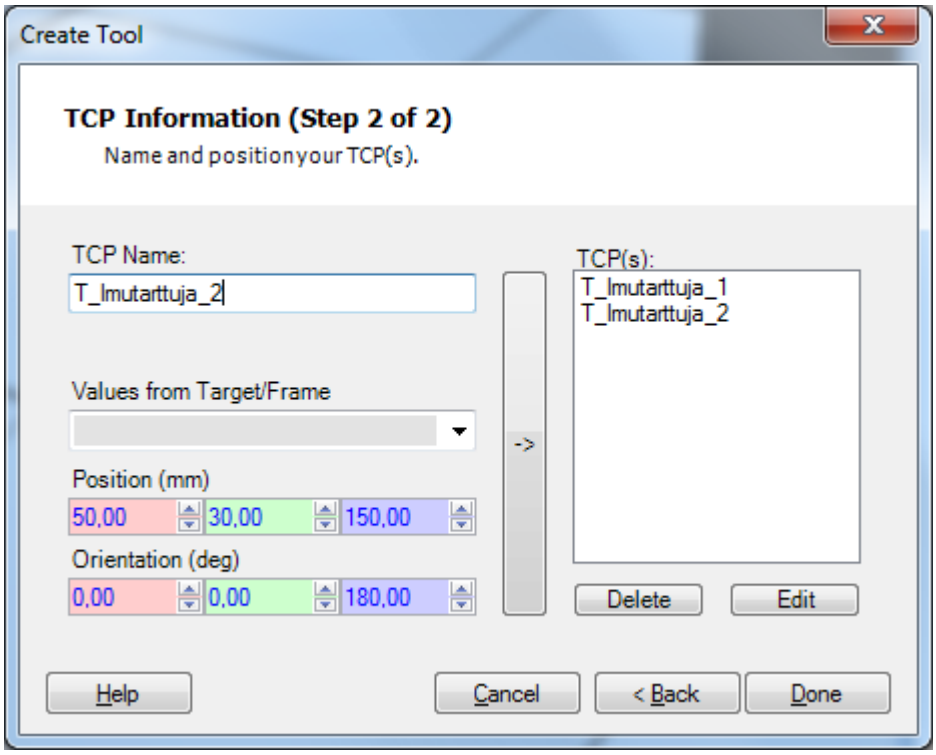


Kuvio 15. Työkalun luonti-ikkuna

Seuraavassa ikkunassa määritetään työkalulle työkalupisteet. Työssä käytettävässä työkalussa on 2 imukuppitarttujaa, joten niille molemmille määritettiin omat työkalupisteet. Työkalupisteen sijainnin ja asennon voi joko kirjoittaa kenttiin, tai sen voi myös valita hiirellä layoutin 3D-mallista. Työssä käytettiin jälkimmäistä tapaa. Kuviossa 16 on esitetty ensimmäisen työkalupisteen sijainti ja kuviossa 17 työkalupisteiden asetusikkuna valmiina.



Kuvio 16. Työkalupisteen sijainti imukupissa 1

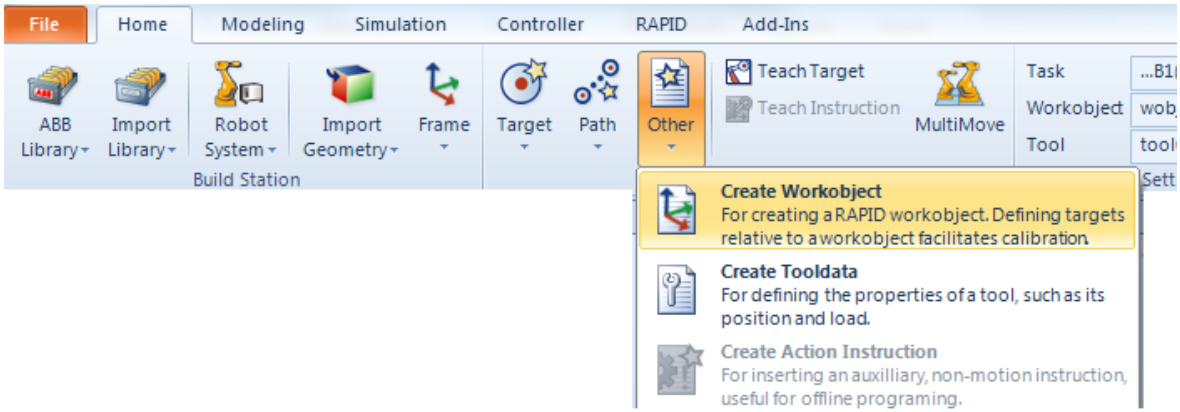


Kuvio 17. Työkalupisteiden luonti-ikkuna

4.4.3 WORKOBJECT-KOORDINAATISTO

Workobject-koordinaatiston eli työkohde-koordinaatiston avulla voidaan määrittää työskentelypisteitä työalueelle. Jos työalueen tai robotin sijainti muuttuu myöhemmin, tarvitsee ainoastaan Workobject-koordinaatiston sijainti päivittää, jolloin työskentelypisteet päivittyvät niihin yhdistetyn Workobject-koordinaatiston mukana. Workobject-koordinaatiston luominen helpottaa myös ohjelman kalibrointia siirryttäessä simulaatiosta oikeaan ympäristöön.

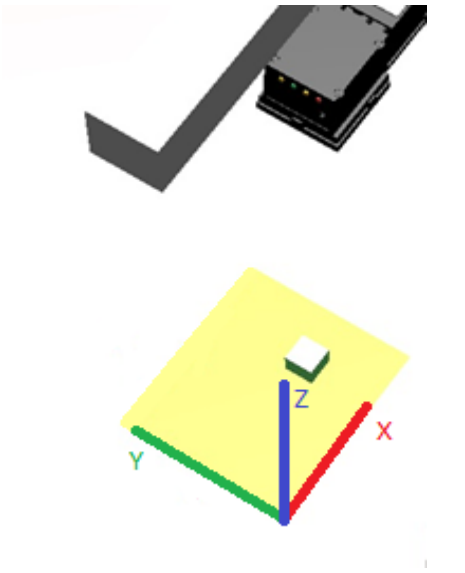
Workobject-koordinaatisto luodaan valitsemalla Home-välilehdeltä Other-painikkeen alta Create Workobject -painike (kuvio 18).



Kuvio 18. Create Workobject -painike

Painikkeesta avautuu ikkunan vasempaan reunaan ikkuna, johon annetaan koordinaatiston tiedot. Koordinaatiston sijainti ja asento annetaan kenttään User Frame. Position x, y, z -kenttään annetaan koordinaatiston origon sijainti ja Rotation rx, ry, rz -kenttään koordinaatiston asento.

Työn solulle määritettiin kolme Workobject-koordinaatistoa, jotka sijaitsivat kameran kuvausalueen kulmassa, pudotusputkien kiinnitysprofiilin kulmassa, sekä noppakipon kulmassa. Kuvassa 19 on esitetty kameran kuvausalueen Workobject-koordinaatiston sijainti. Kameran kuvausalueen Workobject-koordinaatisto sijaitsee samassa paikassa ja sen asento on sama, kuin myöhemmin esiteltävä konenäköjärjestelmän kalibrointikoordinaatisto.



Kuvio 19. Kuvausalueen workobject

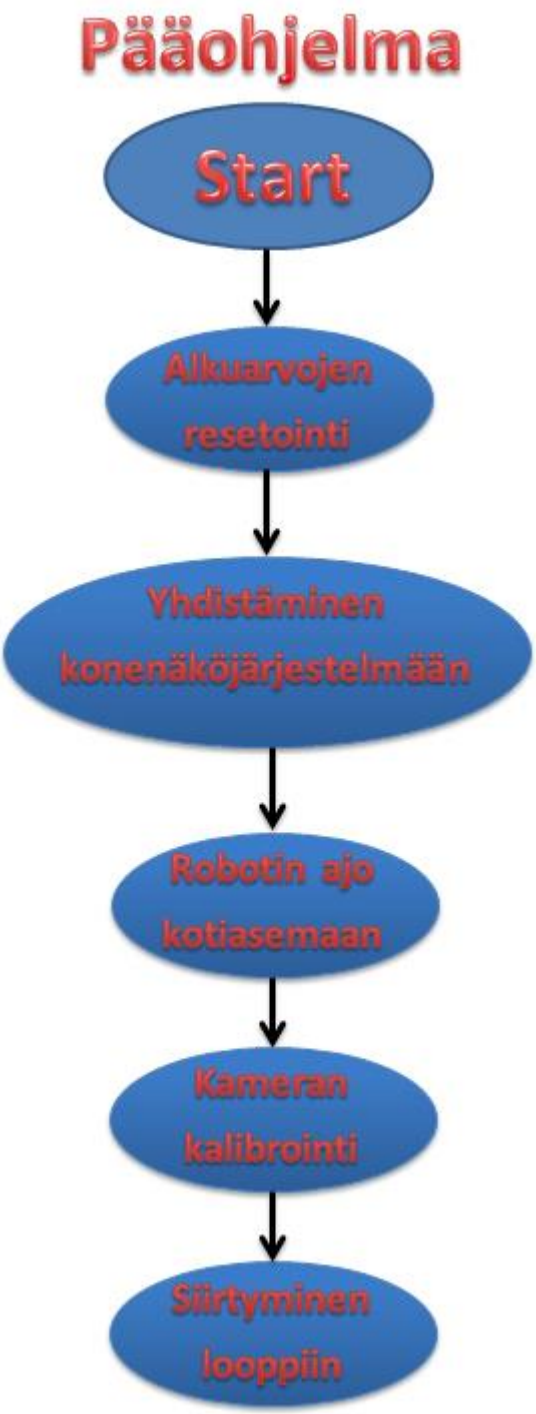
4.4.4 OHJELMOINTI

Robotin ohjelmointi voidaan toteuttaa usealla eri tavalla, esimerkiksi ABB RobotStudioon sisältyvällä virtuaalisella tai robotin omalla FlexPendant-paneelilla, kirjoittamalla RAPID-koodia käsin tekstinkäsittelyohjelmalla tai RobotStudion omalla RAPID-editorilla. Työssä käytettiin pääasiassa RAPID-editoria ja tehtiin pieniä muokkauksia FlexPendant-paneelilla. Ohjelmassa käytetyt työpisteet luotiin RobotStudion layoutnäkymässä.

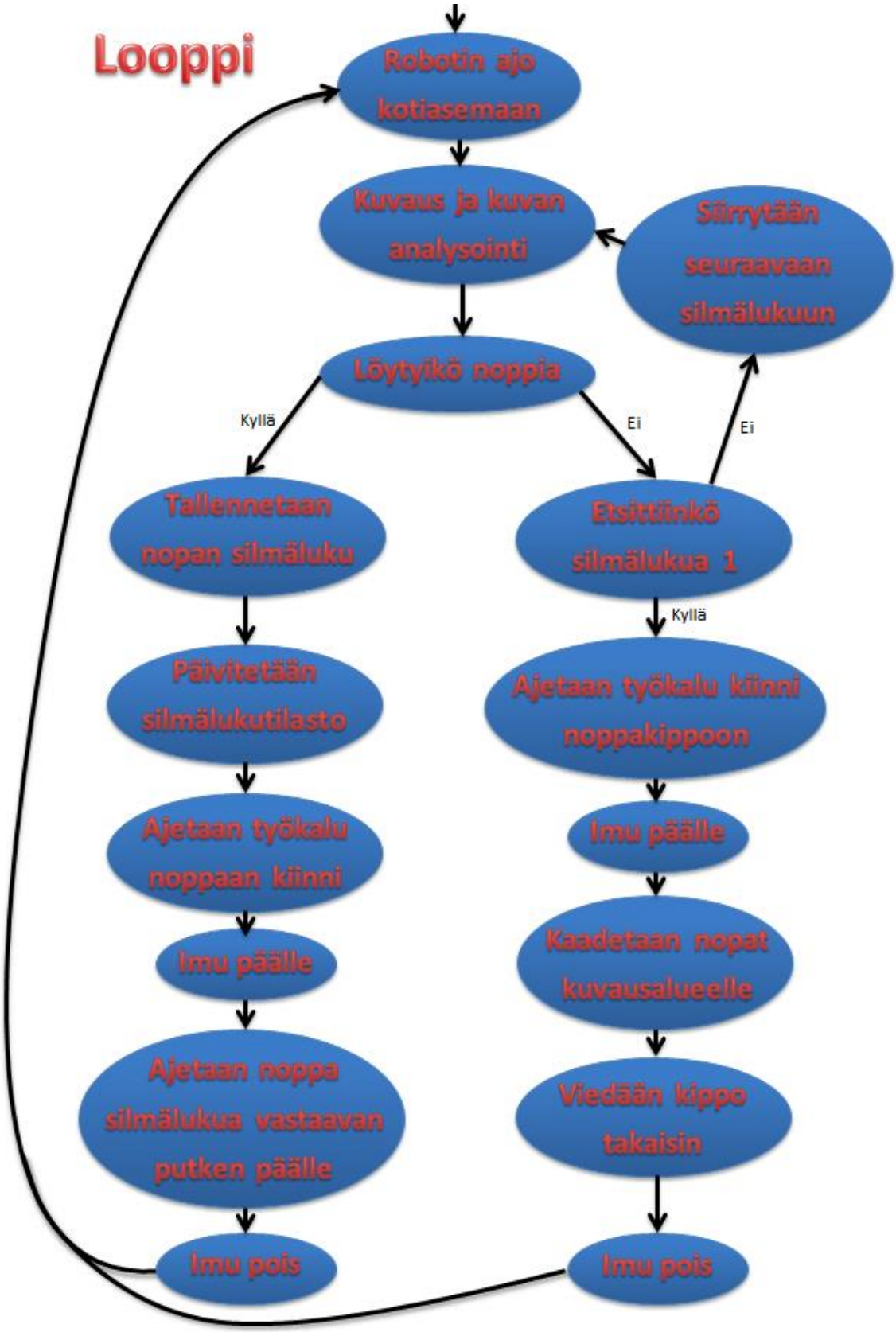
Kuviossa 20 on osa robotin pääohjelmasta, jossa ajetaan robotti ensin kotiasemaan. Tämän jälkeen otetaan kuva ja selvitetään onko noppia jäljellä ja jos on, selvitetään nopan silmäluku ja tallennetaan se muuttujaan nSilmaluku. Tämän lisäksi kasvatetaan nKutoset-muuttujaa yhdellä silmäluvun ollessa kuusi, jolloin saadaan paneelin näytölle tilastoa eri silmäluvuista. Seuraavaksi nopan löytyessä suoritetaan aliohjelma, jossa robotti hakee nopan ja vie sen oikean pudotusputken päälle. Jos noppia ei löydy, tilanteesta riippuen joko kuvataan seuraavaa silmälukua tai suoritetaan aliohjelma, jossa robotti heittää nopat uudelleen ja palaa tämän jälkeen alkuun. Kaavioissa 1 on esitetty pääohjelman toiminta ja kaaviossa 2 on esitetty loopin toiminta.

```
PROC Looppi()
    ikiluuppi:
            MoveL pKotiasema,v500,fine,Imutarttuja1; ! Ajetaan kotiasemaan
            rTuote:=OtaKuva(nScene); ! Otetaan kuva
            IF rTuote.tulos=TRUE THEN ! Jos kuvasta löytyy noppia suoritetaan toiminta sen mukaan mikä silmäluku löytyy.
                incr nNoppa; ! nNoppa kertoo, miten monta noppaa on löydetty. Luku näkyy käyttöliittymässä.
                IF nScene=0 THEN ! Jos Scene on 0. on löydetty silmäluku 6.
                    nSilmaluku:=6; ! asetetaan silmäluku nSilmaluku-muuttujaan
                    Incr nKutoset; ! kasvatetaan ko. silmäluvun muuttujaa yhdellä

                ELSEIF nScene=1 THEN
                    nSilmaluku:=5;
                    Incr nVitoset;

                ELSEIF nScene=2 THEN
                    nSilmaluku:=4;
                    Incr nNeloset;

                ELSEIF nScene=3 THEN
                    nSilmaluku:=3;
                    Incr nKolmoset;

                ELSEIF nScene=4 THEN
                    nSilmaluku:=2;
                    Incr nKakkoset;

                ELSEIF nScene=5 THEN
                    nSilmaluku:=1;
                    Incr nYkkoset;

                ENDIF
            ELSEIF rTuote.tulos=FALSE THEN
                IF nScene>=5 THEN   ! kun scene on 5 tai suurempi, noppia ei enää löydy, joten heitetään nopat uudelleen ja palataan sceneen 0.
                    Heitto;
                    nNoppa:=0;
                    nScene:=0;
                ELSE
                    ! Jos kaikkia silmälukuja (scenejä) ei ole testattu nostetaan scenen numeroa yhdellä.
                    incr nScene;
                ENDIF
                GOTO ikiluuppi; ! palataan loopin alkuun
            ENDIF
            ! Asetetaan nopan sijainti pNoutoReferenssi -muuttujaan
            pNoutoReferenssi.trans.x:=rTuote.koordinaatti.x; ! X sijainti
            pNoutoReferenssi.trans.y:=rTuote.koordinaatti.y; ! Y sijainti
            ! pNoutoReferenssi.rot:=rTuote.kiertyma;         ! Kiertymä
            Poiminta; ! suoritetaan poiminta
            Pudotus;! Suoritetaan pudotus
            GOTO ikiluuppi; ! Palataan loopin alkuun
ENDPROC
```

Kuvio 20. Ohjelmakoodia



Kaavio 1. Pääohjelman toimintakaavio



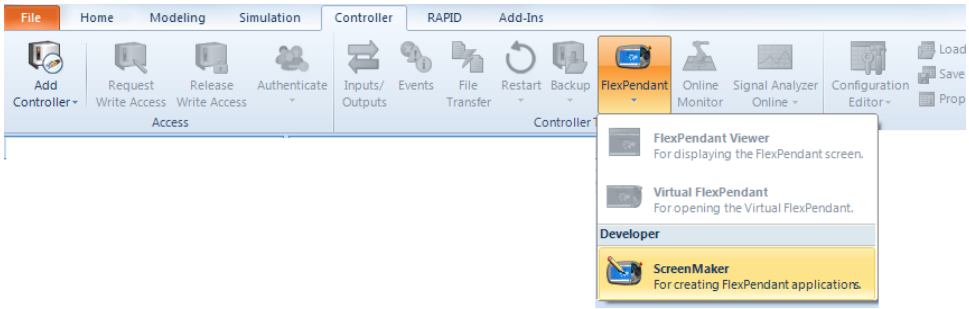
Kaavio 2. Loopin toimintakaavio

Kun robottiin on ohjelmoitu jokin liikekäsky, voidaan se simuloida aikaisemmin luodussa ympäristössä. Näin nähdään ovatko ohjelmoidut liikkeet mahdollisia. Simuloimalla liikkeet ensin tietokoneen näytöllä vältytään mahdollisilta vahingoilta, jotka aiheutuvat robotin törmäämisistä ja voidaan tarkistaa, että robotti yltää joka paikkaan.

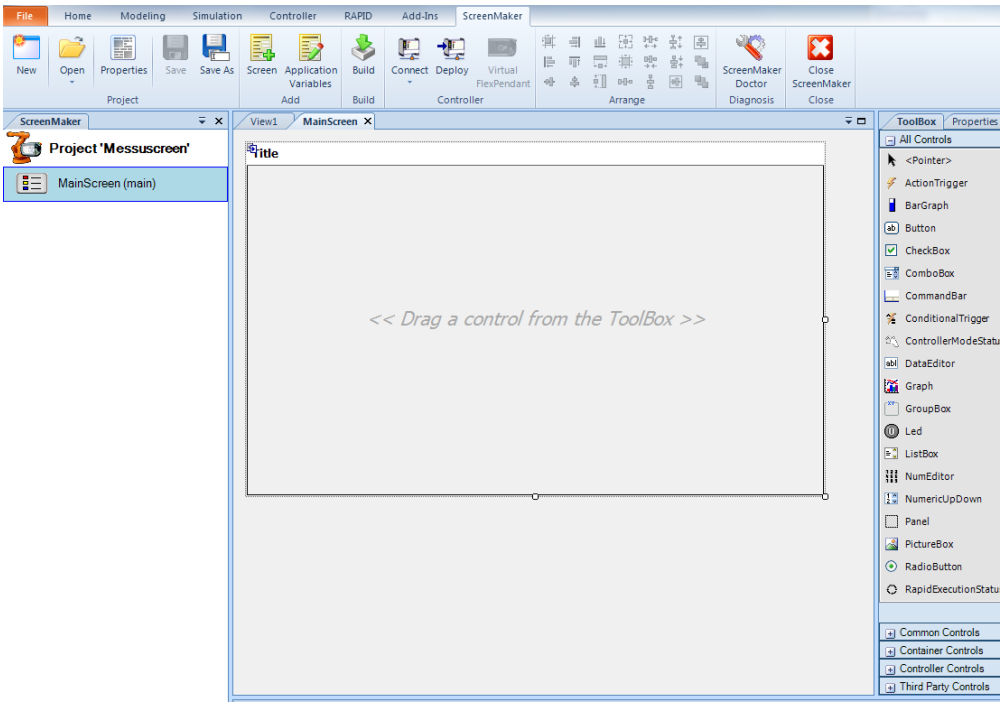
Myös konenäköjärjestelmän ohjaus ja sieltä saatujen tulosten lukeminen onnistuu virtuaalikontrollerilla, kunhan siihen on asennettu PC Interface -optio. Konenäköjärjestelmää testattiinkin ensin simuloidun robotin avulla, vaikkakaan näin tehtäessä ei voitu olla varmoja siitä, että konenäköjärjestelmän ilmoittamat pisteet ohjasivat robottia oikeisiin sijainteihin.

4.5 KÄYTTÖLIITTYMÄN LUOMINEN

FlexPendant-paneelille voidaan tehdä oma käyttöliittymä RobotStudioon kuuluvalla ScreenMaker-ohjelmistolla (kuvio 22). Ohjelmisto toimii ainoastaan RobotStudion 32-bittisessä versiossa. ScreenMaker-ohjelmisto käynnistetään RobotStudion Controller-välilehdeltä FlexPendant-painikkeen alla olevalla ScreenMaker-painikkeella (kuvio 21).



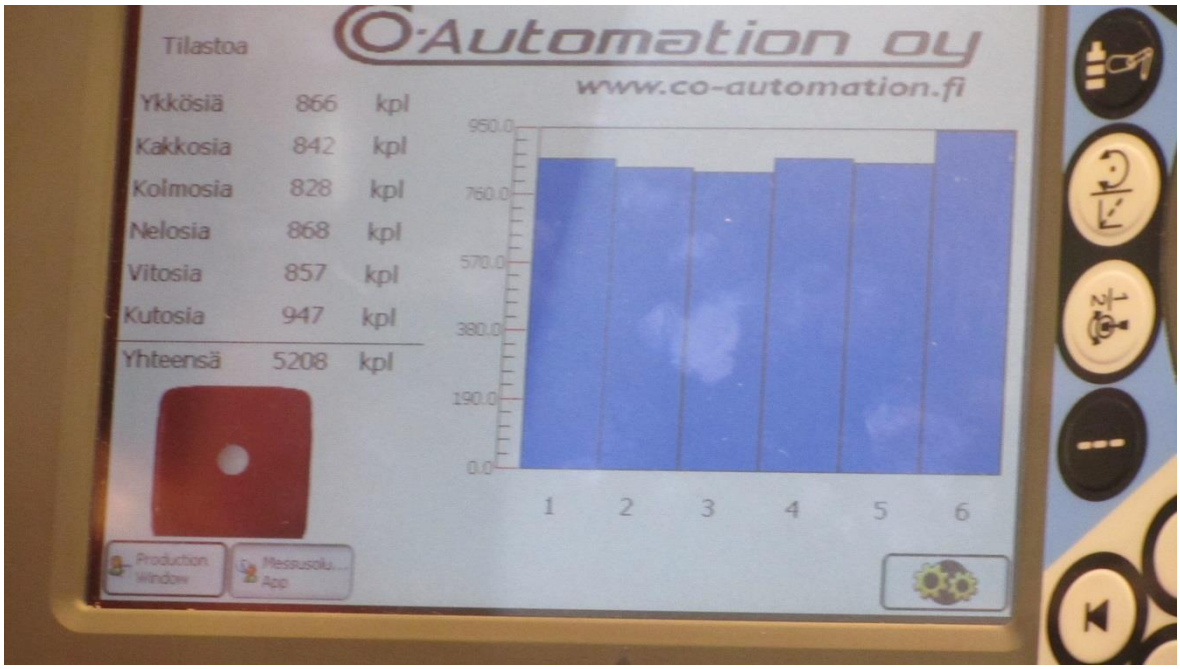
Kuvio 21. ScreenMaker-painike



Kuvio 22. ScreenMaker-ohjelmiston aloitusnäkymä

ScreenMaker-ohjelmistolla voidaan luoda käyttöliittymä, joka voi sisältää esimerkiksi kuvia, tietoa robotin sijainnista, numerodataa, ohjauspainikkeita ja -arvoja.

Tässä työssä käyttöliittymässä esitettiin eri silmälukujen määrä sekä määrää kuvaava pylväsdiagrammi. Käyttöliittymässä näkyi myös, mitä robotti tekee milloinkin. Vaiheet esitettiin kuvilla, joissa on esimerkiksi sen nopan silmäluku, jota robotti parhaillaan käsitteli, tai kameran kuva robotin odottaessa kameran tuloksia. Kuvissa 23 on esitetty kuva käyttöliittymästä kolmen messupäivän jälkeen.



Kuvio 23. Käyttöliittymä kolmen messupäivän jälkeen

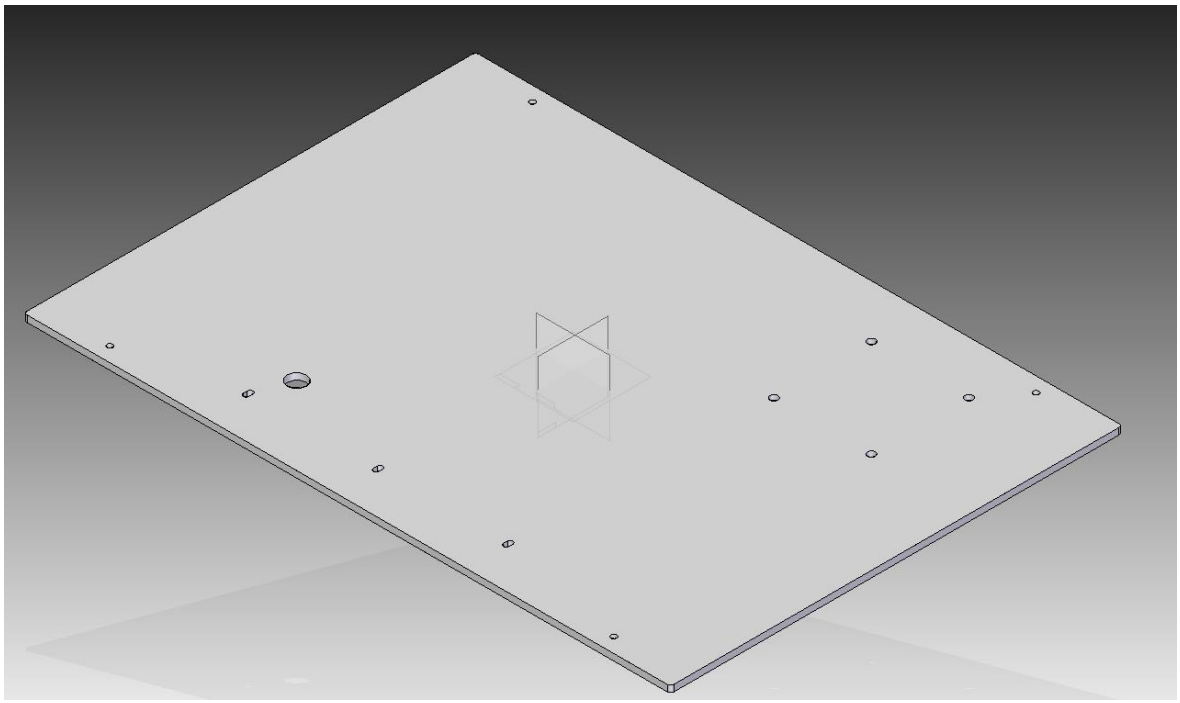
5 MESSUSOLUN PÖYDÄN RAKENTAMINEN

5.1 PÖYDÄN SUUNNITTELU

Co-Automation Oy:llä oli valmis pöytä edellisiltä messuilta, mutta se oli liian kevyttä tekoa, joten sitä täytyi uudistaa. Vanha pöytä koostui alumiinisesta pöytärungosta, kahdesta lastulevystä ja pleksisuojasta. Pöytärunko oli rakennettu ohuista alumiiniprofiileista, jotka täytyi vaihtaa vahvempaan, koska robotin liikkuessa pöytä heilui holtittomasti ja heilutti näin ollen konenäkökameraa. Kameran heiluessa sen tarkkuus heikkeni, mikä aiheutti liian paljon virhetilanteita. Pöytälevynä oli lastulevy, joka ei enää ollut erityisen edustavassa kunnossa, joten sen tilalle suunniteltiin uusi levy. Vanhasta pöydästä jätettiin uuteen pöytään ainoastaan pleksisuoja ja robotin kontrollerin alustana toimiva lastulevy.

Uuden pöydän runko valmistettiin Bosch Rexroth Oy:n alumiiniprofiileista ja se suunniteltiin Bosch Rexroth Oy:n omalla MTpro-ohjelmistolla. MTpro-ohjelmisto on 3D CAD -suunnitteluohjelmisto, joka sisältää Bosch Rexroth Oy:n katalogin. Ohjelmiston avulla luodaan 3D-malli, joka sisältää tarvittavien alumiiniprofiilien lisäksi myös tarvittavat kiinnitystarvikkeet. Mallin perusteella ohjelma luo automaattisesti ostoslistan, jonka voi lähettää suoraan Bosch Rexroth Oy:n edustajalle.

Uusi pöytälevy suunniteltiin Solid Edge ST2 -ohjelmistolla ja materiaaliksi valittiin 10 mm paksu teräslevy, jonka päälle sijoitettiin vielä 3 mm paksu rosterilevy koristeeksi. Molemmat levyt valmistettiin mallin mukaiseksi vesileikkaamalla. Pöytälevyn 3D-malli on esitetty kuviossa 24.



Kuvio 24. Uuden pöytälevyn 3D-malli

5.2 PÖYDÄN RAKENTAMINEN

Kun uuden pöydän osat oli vastaanotettu, voitiin pöytä rakentaa ja siirtää solun komponentit uudelle pöydälle. Kameran ja pudotusputkien telineet valmistettiin Co-Automation Oy:ltä ylimääräiseksi jääneistä alumiiniprofiileista. Uusi pöytä on esitetty kuviossa 25.

Kuvio 25. Uusi pöytä

6 KONENÄKÖJÄRJESTELMÄN ASENNUS JA OHJELMOINTI

Työssä käytettiin Omronin FQ2-konenäköjärjestelmää, joka on niin sanottu älykamera. Älykameraan on sisällytetty kaikki konenäköjärjestelmään tarvittava laitteisto yhteen pakettiin.

Seuraavissa kappaleissa kerrotaan Omron FQ2 -konenäön asentamisesta ja ohjelmoinnista.

6.1 KONENÄKÖKAMERAN ASENNUS

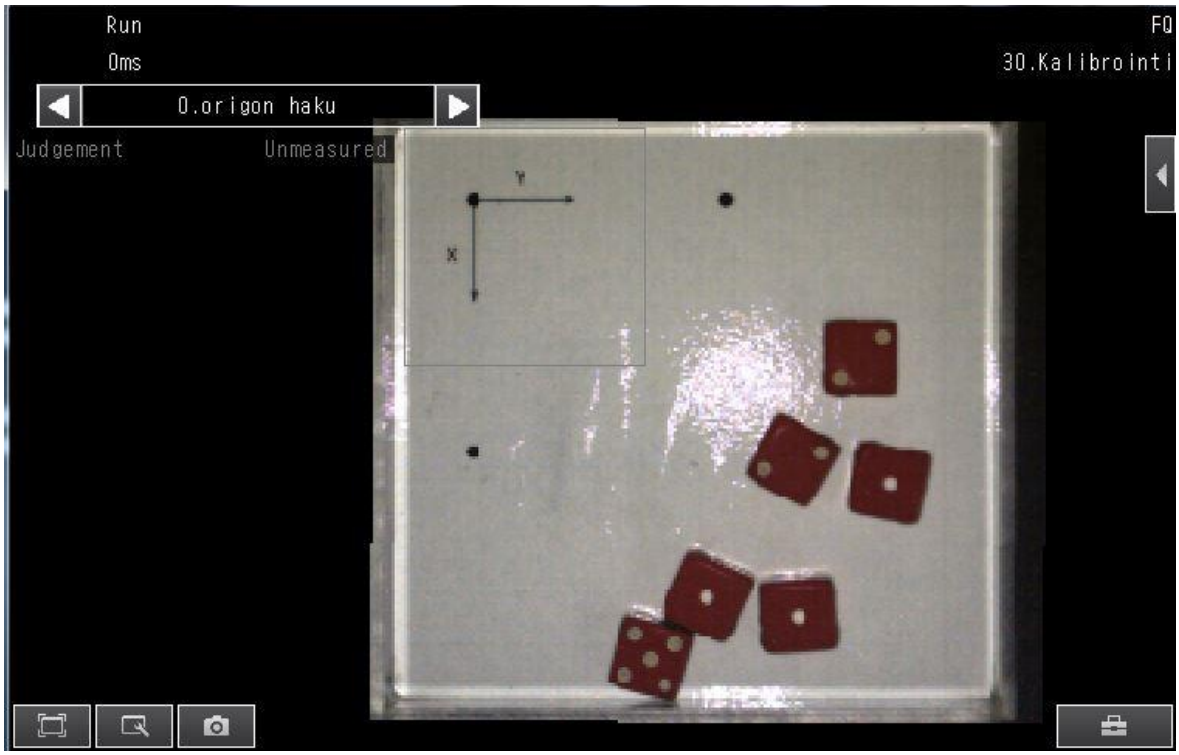
Konenäkökameralle rakennettiin alumiiniprofiilista teline, jolle se asennettiin. Kamera asetettiin korkeudelle, jossa kuvausalueen koko on noin 150 mm x 150 mm.

Omronin FQ2-konenäkökamerassa on kaksi liitäntää, joista toinen on Ethernet-liitäntä ja toisessa on virta- ja signaaliliitännät. Konenäköjärjestelmä liitettiin ohjelmointivaiheessa PC:n Ethernet-liitäntään, jolloin kameraa pystyttiin ohjaamaan tietokoneelle asennettavalla Omronin TouchFinder PC -ohjelmistolla. Virtaa kameralle annettiin aluksi tasavirtalähteellä, joka on esitetty kuviossa 26. Myöhemmin konenäköjärjestelmä liitettiin robotin kontrolleriin, jolloin robotti pystyi ohjaamaan kameran toimintoja. Signaaliliitäntöjä ei työssä tarvittu, koska kaikki järjestelmästä saadut tulokset ohjattiin Ethernet-liitännän kautta robotin kontrollerille.

Kuvio 26. Tasavirtalähde

6.2 KONENÄKÖJÄRJESTELMÄN OHJELMOINTI

Konenäköjärjestelmää voidaan ohjelmoida joko tietokoneelle asennettavalla TouchFinder PC -ohjelmistolla tai järjestelmään saatavalla kosketusnäytöllä. Työssä käytettiin ainoastaan PC-ohjelmistoa, sillä näyttöä ei ollut saatavilla. TouchFinder PC -ohjelmiston käyttöliittymä on juuri samanlainen kuin konenäköjärjestelmään kuuluvan kosketusnäytön käyttöliittymä. Kuviossa 27 on esitetty TouchFinder PC -ohjelmiston näkymä RUN-tilassa, kun sceneksi on valittu kalibrointi-scene.

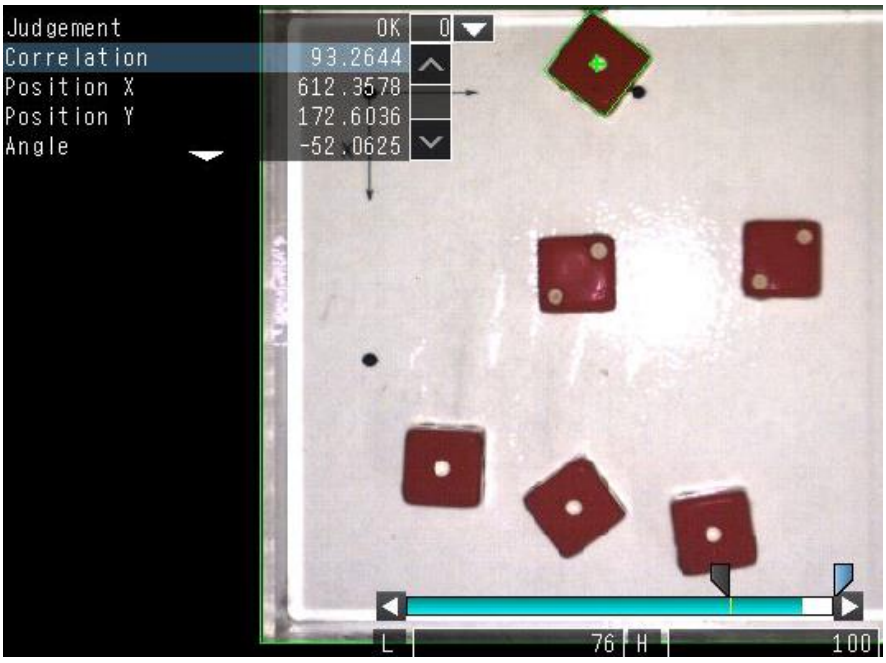


Kuvio 27. TouchFinder PC -ohjelmiston näkymä RUN-tilassa

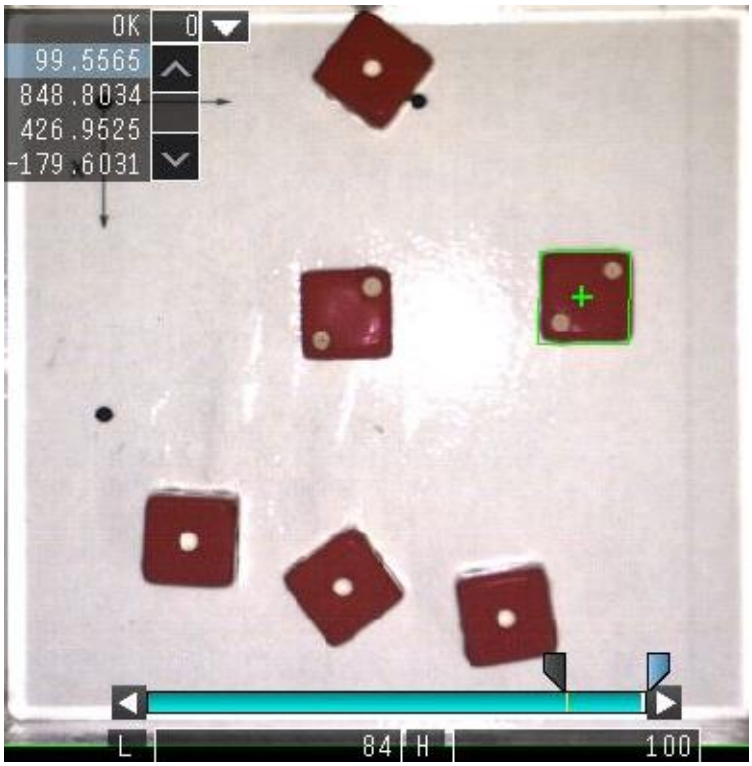
Konenäköjärjestelmän ohjelmoinnissa hankalinta oli saada järjestelmä tunnistamaan noppien eri silmäluvut, koska nopan kaikki tahkot ovat samanmuotoisia ja noppiin muodostui kameran salamasta heijastumia, jotka aiheuttivat häiriöitä pisteiden tunnistukseen. Kameran asetuksia säätämällä, sekä lisäämällä Enhance edges -filtteri saatiin kuitenkin järjestelmä tunnistamaan nopat oikein.

Konenäköjärjestelmän sisältäessä salamavalon oletettiin, että valaistus on joka tilanteessa lähes sama, joten valaistukseen ei salamavalon voimakkuuden säädön lisäksi kiinnitetty juurikaan huomiota.

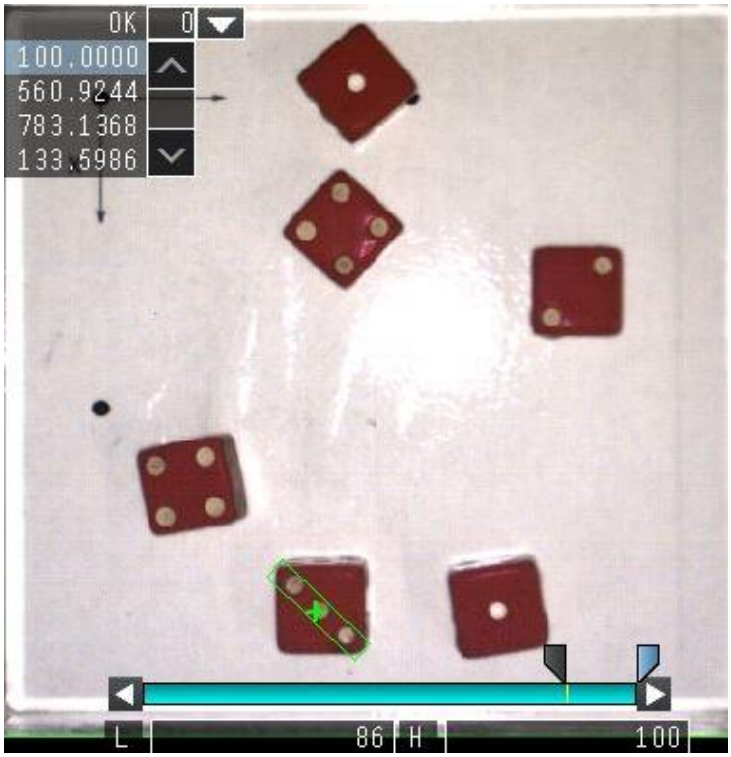
Järjestelmään luotiin kuusi eri sceneä, jotka kaikki tunnistavat eri silmäluvun. Näin pystyttiin ohjelmoimaan robotti siten, että se poimii nopat suuruusjärjestyksessä, alkaen suurimmasta silmäluvusta. Sceneihin luotiin Shape Search II -Inspection, joka etsii kuvattavasta alueesta ennalta määritettyjä muotoja. Hakuun voidaan määrittää alue, jolta muotoja etsitään. Mitä pienempi hakualue on, sitä nopeammin järjestelmä toimii. Kuvioissa 28–33 on esitetty miten kamera tunnistaa eri silmäluvut. Tunnistettu alue on esitetty vihreällä reunuksella. Kuvista selviää myös, miten hyvin tunnistettu kuvio vastaa opetettua kuviota, missä kuvio sijaitsee, sekä kuvion asento. Kuvien alareunassa olevaan palkkiin on annettu vastaavuuden raja-arvot, joiden sisällä järjestelmä hyväksyy kuvion.



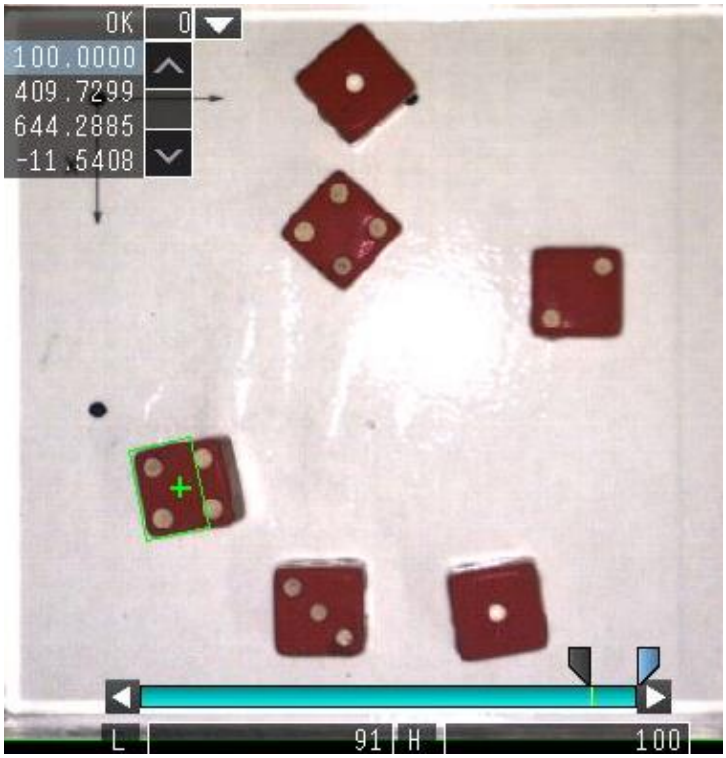
Kuvio 28. Ykkösen tunnistus



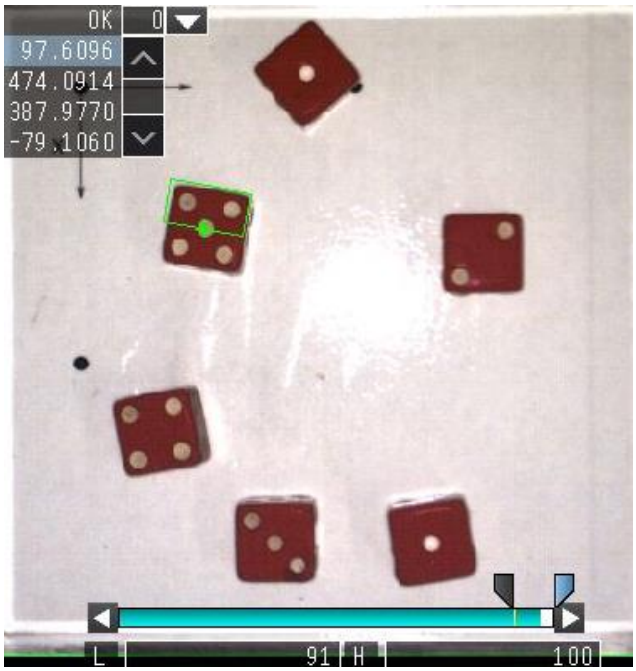
Kuvio 29. Kakkosen tunnistus



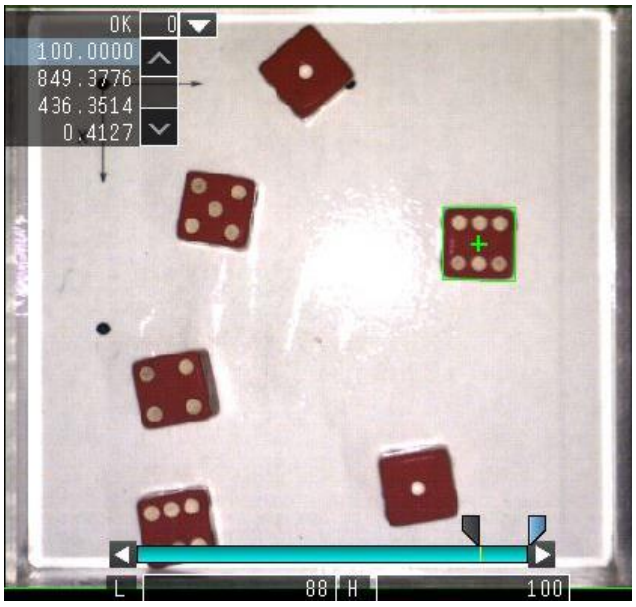
Kuvio 30. Kolmosen tunnistus



Kuvio 31. Nelosen tunnistus

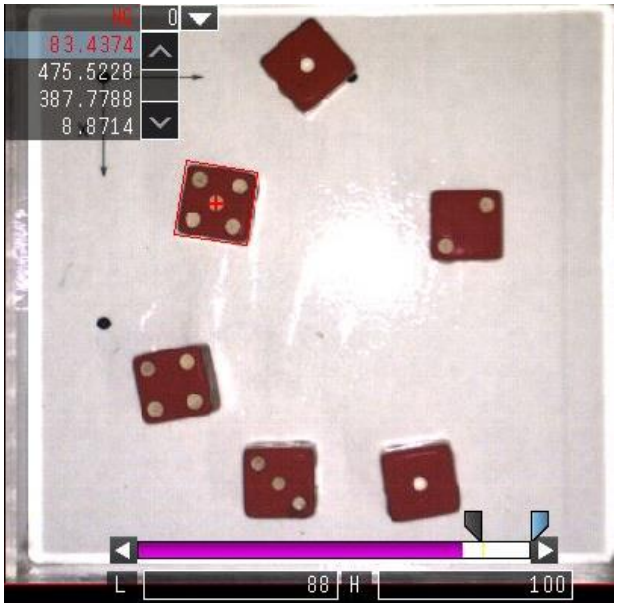


Kuvio 32. Viitosen tunnistus



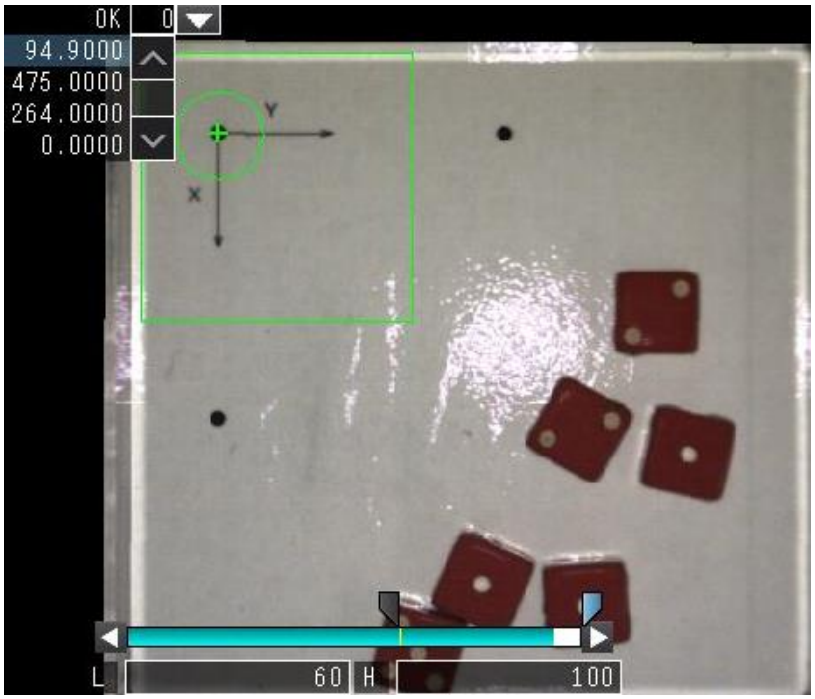
Kuvio 33. Kuutosen tunnistus

Kuviosta 34 näkyy, miten kamera löytää silmäluvun viisi, vaikka sceneksi on valittu kuutonen. Järjestelmä hylkää silti tuloksen eli asettaa tuloksen NG-tulokseksi (Not Good), koska vastaavuus on liian pieni. Järjestelmä etsii tuloksia vastaavuusjärjestyksessä, jolloin ensin löytyy parhaiten opetettua vastaavat kuviot. Joten tilannetta, jossa esimerkiksi kuutosia olisi vielä pöydällä, mutta kamera löytää ensin jonkun muun silmäluvun, ei tule.

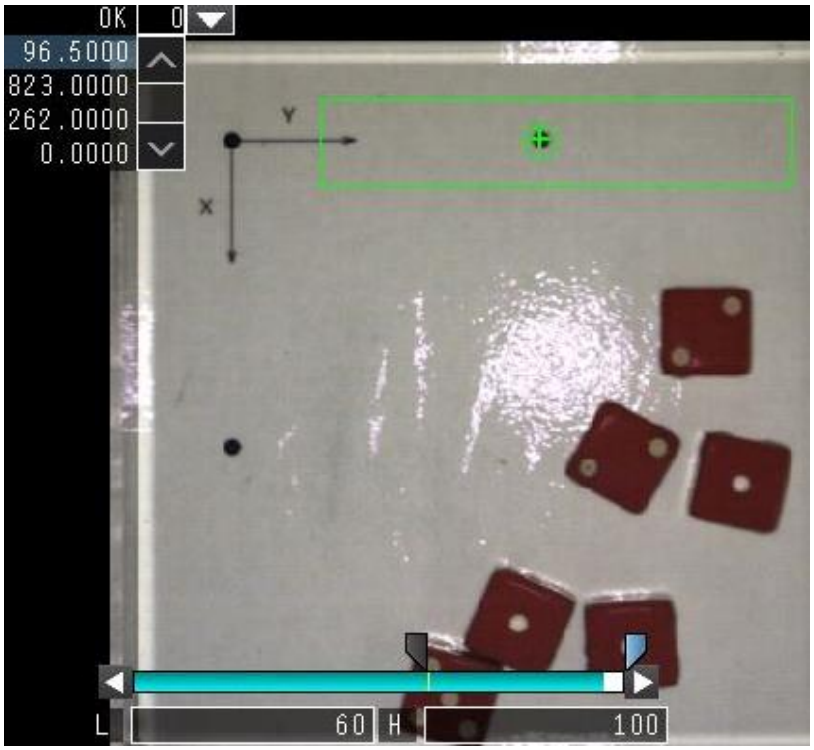


Kuvio 34. NG-tulos

Järjestelmään luotiin scene myös kameran kalibroinnille. Kalibrointiscenessä järjestelmä laskee kuvasta kalibrointikoordinaatiston Y-akselin pituuden pikseleinä, sekä koordinaatiston asennon. Pikselit muunnetaan myöhemmin robotin kontrollerilla millimetreiksi. Järjestelmä kääntää kuvan automaattisesti siten, että Y-akselin suunta osoittaa oikealle ja X-akselin alas. Kääntäminen on määritetty ShapePos Comp -työkalulla, johon on opetettu koordinaatiston kuva oikeassa asennossa. Kuviossa 35 näkyy, miten järjestelmä löytää origon, ja kuviossa 36 miten se löytää Y-akselin.

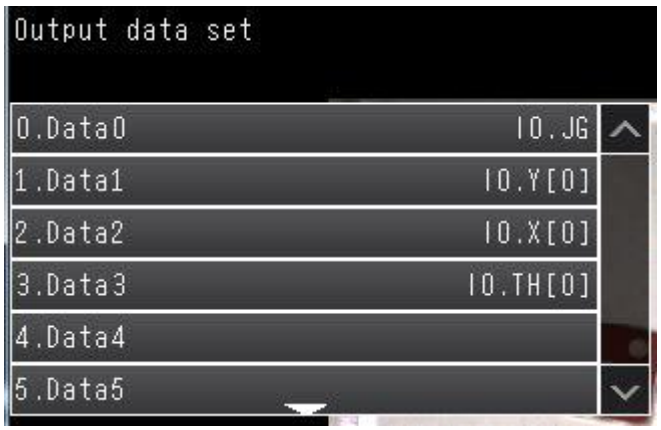


Kuvio 35. Origon sijainti



Kuvio 36. Y-akselin sijainti

Kun järjestelmä on löytänyt etsittävän kuvion, se lähettää tiedot eteenpäin. Eteenpäin lähetettäviä tietoja voidaan luoda ja muokata I/O setting -valikossa. Kuviossa 37 esitetään nopan sijainnin ulostulotiedot, jossa Data0 on judgement eli tieto onko noppaa löydetty vai ei. Data1 on nopan Y-sijainti ja Data2 X-sijainti. Data3 kertoo nopan kiertymän, mutta tätä tietoa ei työssä käytetty, koska robotti käsittelee noppia imukuppitarttujalla, joten nopan kiertymällä ei ole merkitystä.



Kuvio 37. Nopan tietojen output-taulukko

Kuviossa 38 on esitetty kalibrointiscenen ulostulotaulukko. Taulukossa Data0 on origonhaun judgement eli onko origo löydetty. 1. data on origon X-sijainti, 2. on origon Y-sijainti, 3. on origon kiertymä asteina, 4. on y-akselilla sijaitsevan pisteen judgement eli onko pistettä löytynyt ja 5. on origon ja Y-akselin pisteen välinen etäisyys pikseleinä.

Kuvio 38. Kalibroinnin output-taulukko

7 ROBOTIN ASENNUS JA OHJELMOINTI

7.1 ROBOTIN ASENNUS

Robotti asennettiin pöytälevyyn tehtyihin reikiin pulteilla ja johdot liitettiin robotin ja kontrollerin välille. Johdot vietiin kontrollerille pleksisuojan oveen tehdyn reiän kautta.

Robotin käynnistyessä ensimmäistä kertaa täytyi sen kierroslaskurit (revolution counters) päivittää. Päivitys tapahtuu siten, että liikutetaan robotin akselit kotiasemiin, jotka on merkitty robotin runkoon viivoilla. Tämän jälkeen valitaan paneelilta update revolution counters ja valitaan akselit, joiden laskurit halutaan päivittää. Tässä tapauksessa päivitettiin kaikki akselit. Jos laskureita ei päivitetä, ei robottia voida ajaa kuin käsiajolla akseleittain, eikä sitä voida ohjelmoida, koska robotti ei tiedä missä sen akselit sijaitsevat. Laskurien päivitys täytyy suorittaa myös, jos robotti on liikkunut virrattomana, esimerkiksi kuljetuksen aikana aiheutuvien äkillisten tärähdysten takia. Tämän takia robotti kannattaakin aina ajaa kuljetuksen ajaksi esimerkiksi pöydällä olevaa pehmustetta vasten, jolloin robotti ei pääse liikkumaan.

Kierroslaskurien päivityksen jälkeen robotti oli valmis ohjelmointia varten.

7.2 TARTTUJAN VALMISTAMINEN

Co-Automation Oy:llä oli vanhasta projektista jäänyt ylimääräiseksi imukuppitarttujan runko ja neljä imukuppia, joten niitä käytettiin työssä. Koska noppia kuljetetaan aina yksi kerrallaan ja kipon nostamiseen tarvitaan kaksi imukuppia, riittää että työkalussa on kaksi imukuppitarttujaa. Näin ollen rungosta katkaistiin toinen puoli pois.

Vanhasta projektista oli jäänyt ylimääräiseksi myös imukuppien ohjaukseen tarkoitetut SMC:n tyhjiöejektorit (kuvio 39), joten ne asennettiin pöydän alle riviliittimien läheisyyteen (kuvio 40).

Kuvio 39. SMC-tyhjiöejektori

Kuvio 40. Ejektorit ja riviliittimet asennettuna pöydän alle

Solussa käytettävien noppien pinta oli lähes samankokoinen kuin imukuppien tartunta-ala, mikä aiheutti virhetilanteita tartunnassa. Tästä syystä hankittiin pienemmät imukupit vanhaan runkoon. Uudet imukupit olivat juuri sopivan kokoiset ja niiden ansiosta tarttuja sai aina kiinni nopasta. Kuviossa 41 on esitetty valmis tarttuja.

Kuvio 41. Imukuppitarttuja

7.3 I/O-OHJAUSKORTIN AKTIVOINTI

Koska tyhjiöejektoreita ohjataan robotin kontrollerin digitaalisella output (DO) käskyllä, täytyi I/O-ohjauskortille tuoda ohjausjännite, jota ei ollut tehty tehtaalla valmiiksi.

I/O-ohjauskortti oli kytketty kontrollerin PCI-liitäntään, joten itse kortti sai virtansa suoraan emolevyltä, mutta DO-ohjaukseen vaadittavaa jännitettä ei saatu kortin liittimiltä ulos. Tämän seurauksena kontrolleri jouduttiin avaamaan ja tarkistamaan kortin liitännät. Huomattiin, että jännitettä ei ollut liitetty kortille ja selvitettiin, että se saadaan toimimaan yhdistämällä liittimen XS10 pinnit 1 ja 4, sekä 2 ja 3 yhteen. Pinniin 1 on liitetty 24 voltin jännite ja pinniin 2 0 V. Pinnissä 4 on I/O-kortin 24 V ja pinnissä 3 kortin 0 V.

Yhdistämisen jälkeen annettaessa robotin paneelilta käsky laittaa jokin DO-signaali päälle, saatiin vastaavalta liittimeltä jännitekäsky. Imukuppitarttujaa pystyttiin siis ohjaamaan suoraan robotilta.

Kuvio 42. Robottikontrollerin sisältö

7.4 ROBOTIN OHJELMOINTI

Robotin ohjelmointi suoritettiin jo simulointivaiheessa, joten ohjelma täytyi ainoastaan ladata robotin kontrolleriin ja muokata Workobject-koordinaatistot oikeisiin sijainteihin. Robotin ja kameran välisen rajapinnan ohjelmoinnissa käytettiin hyväksi Co-Automation Oy:n aiemmin valmistamaa rajapintaa. Rajapinta oli kuitenkin työn tekijälle ennestään tuntematon, joten siihen täytyi perehtyä syvemmin sitä valmistaneen työntekijän kanssa. Rajapintaa myös muokattiin jonkin verran aiheeseen sopivammaksi. Seuraavassa on esitetty rajapinnan keskeisimmät toiminnot.

7.4.1 KALIBROINTI

Kalibroinnin aikana saatuja tietoja käytetään hyväksi noppien sijainnin selvittämiseen. Kalibroinnissa selvitetään pikselikerroin, eli miten monta pikseliä vastaa yhtä millimetriä.

Konenäköjärjestelmän kalibroinnin aikana saadut tulokset lähetetään robotin kontrollerille. Tulokset paloitellaan ja eri arvot tallennetaan omiin muuttujiinsa. Arvoja ovat: origon X- ja Y-sijainti, koordinaatiston kiertymä sekä origon ja Y-akselin pisteiden etäisyys toisistaan. Saadut sijainti- ja etäisyysarvot ovat pikseleinä, joten niistä lasketaan miten monta pikseliä vastaa yhtä millimetriä robotilla. Tämä on mahdollista, kun tiedetään origon ja Y-akselin pisteen välinen todellinen etäisyys millimetreinä.

Ohjelma on kirjoitettu niin, että kalibrointi tarvitsee suorittaa vain kerran. Kuitenkin niin, että jos solulta katkaistaan virta, on kalibrointi suoritettava uudelleen. Ennen kalibrointia robotti ajetaan aina kotiasemaansa, jolloin se ei ole kameran kuvausalueella. Ennen kalibrointia suoritetaan myös yhteyden avaus konenäköjärjestelmään sekä asetusten nollaus, johon kuuluu muun muassa kalibrointikoordinaatiston origon ja Y-akselin pisteen välisen etäisyyden antaminen kontrollerille.

7.4.2 NOPPIEN SIJAINNIN TUNNISTAMINEN

Konenäköjärjestelmä lähettää tiedot robotin kontrollerille, joka paloittelee saadut tiedot oikean mittaisiin tietoihin ja analysoi, onko noppia löydetty. Jos noppa löytyy, laskee kontrolleri nopan etäisyyden origosta, ja näin ollen myös kameran workobjektista, ja muuttaa saadut pikseliarvot millimetreiksi. Kontrolleri siirtää saadut nopan X- ja Y-koordinaatit robtarget-muuttujaan.

Kuviossa 43 on esitetty konenäköjärjestelmästä saadun tuloksen pilkontavaihetta, jossa ensin etsitään konenäköjärjestelmästä saadusta merkkijonosta (strSaatuVastaus) pilkku, joka erottaa eri tulokset toisistaan. Tämän jälkeen otetaan talteen pilkkua edeltävä merkkijono, joka kuvaa nopan X-sijaintia, ja tallennetaan se strTuoteX-nimiseen tekstimuuttujaan. StrTuoteX-muuttuja muunnetaan numeroar-

voksi ja tallennetaan nSceneXPiks-nimiseen muuttujaan. Seuraavaksi konenäköjärjestelmästä saatua merkkijonoa lyhennetään siten, että poistetaan siitä nopan X-sijaintia kuvaava merkkijono. Tämän jälkeen voidaan aloittaa sama toimenpide alusta eri muuttujilla, jolloin saadaan tallennettua kaikki tarvittavat arvot omiin muuttujiinsa. Tätä samaa periaatetta käytetään myös kalibroinnin yhteydessä origon ja Y-akselin välistä pituutta laskettaessa sekä origon sijaintia selvitettäessä.

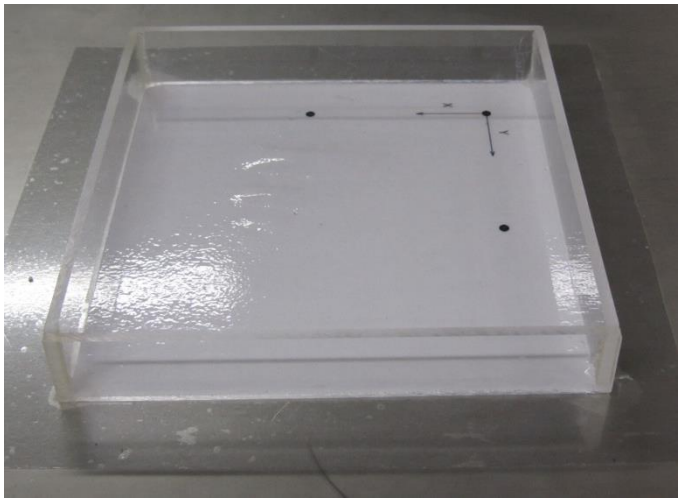
```
!Haetaan tuoteen X
nPilkunpaikka1 := StrFind(strSaatuVastaus,1,",");
strTuoteX:=strPart(strSaatuVastaus,1,nPilkunpaikka1-1);
bMuunnosOK:=strtoval(strTuoteX,nSceneXPiks);
nTuloksenpituus:=StrLen(strSaatuVastaus)-nPilkunpaikka1;
strSaatuVastaus:=strPart(strSaatuVastaus,nPilkunpaikka1+1,nTuloksenpituus);
```

Kuvio 43. Nopan X-sijainnin pilkkominen konenäköjärjestelmästä saadusta tuloksesta

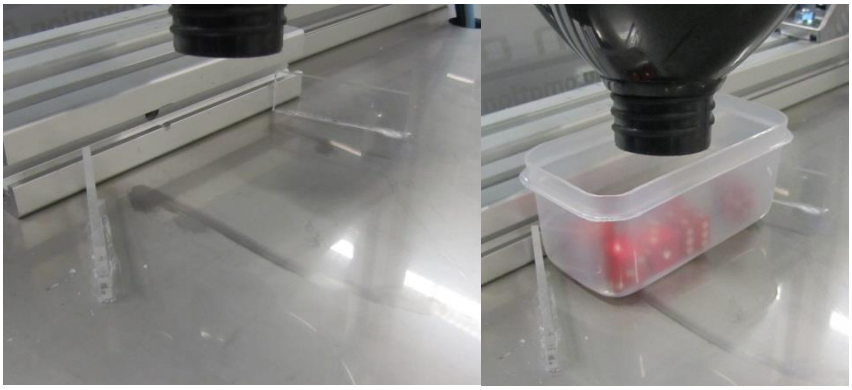
8 MESSUSOLUN VIIMEISTELY

Kun kaikki komponentit saatiin asennettua pöydälle, viimeisteltiin solu tekemällä konenäkökameran kuvausalueen kokoinen kaukalo nopille, sekä ohjurit noppakipolle. Noppakipon ohjureiden tarkoitus on pitää kippo putkien alla ja robotin tartunta-alueella. Kaukalo (kuvio 44) valmistettiin neljästä toisiinsa liimatuista pleksinpalasta. Kaukalon alle asennettiin kontaktimuovilla kalibrointikoordinaatisto. Kalibrointikoordinaatisto sijaitsee samassa paikassa ja asennossa, kuin kameran kuvausalueen Workobject. Tällöin noppien sijainti on sama sekä kalibrointi- että Workobject-koordinaatistossa.

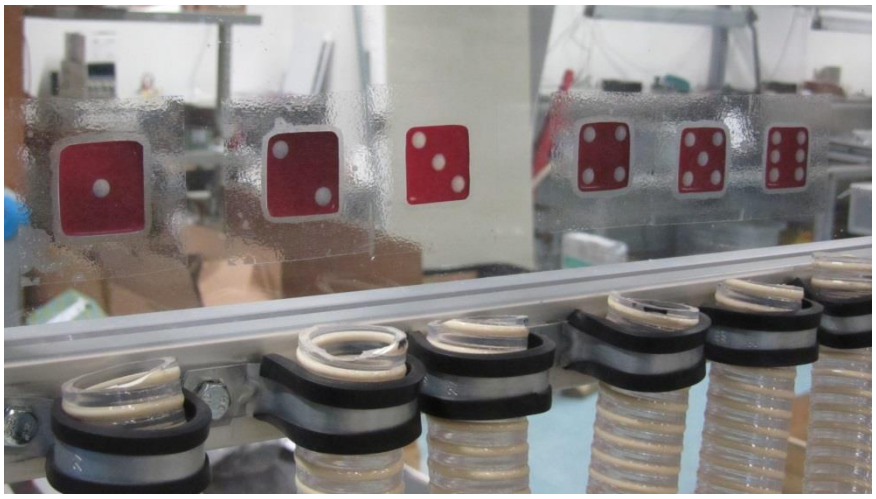
Noppakipon ohjurit (kuvio 45) valmistettiin myös pleksistä, ja ne liimattiin pöytään kiinni. Myös noppien pudotusputkien taakse valmistettiin taustapleksi (kuvio 46), johon liimattiin noppien silmälukujen kuvat kyseisen silmäluvun pudotusputken päälle. Viimeisenä asennettiin pleksisuoja solun ympärille. Kuva valmiista solusta messupaikalla on esitetty kuviossa 47.



Kuvio 44. Noppakaukalo ja referenssikoordinaatisto



Kuvio 45. Noppakipon ohjurit



Kuvio 46. Taustapleksi ja pudotusputket

Kuvio 47. Valmis messusolu

9 POHDINTA

Työ oli erittäin mielenkiintoinen ja juuri sopivan laajuinen opinnäytetyöksi, vaikka aluksi ajattelinkin sen olevan suppeahko. Työstä teki mielenkiintoisen käytännön työ, jossa pääsi toteuttamaan itse suunnittelemansa messusolun. Solun rakentaminen onnistuikin ilman suurempia ongelmia.

Valmis messusolu esiteltiin Tampereella alihankintamessuilla 24. - 26.9.2013 Suupohjan seudun yrittäjien yhteisosastolla. Vaikka soluun vaihdettiin pienemmät imukupit vasta messualueella, toimi se muutamien säätöjen jälkeen hienosti koko messujen ajan, muutamaa nopan pomppaamista kuvausalueen ulkopuolelle lukuun ottamatta. Solu herätti hyvin yleisön ja muiden esillepanijoiden huomion ja sitä onkin esitelty myös messujen jälkeen Co-Automationin tiloissa käyville vierailijoille.

Messusolua suunniteltaessa ja ohjelmoitaessa sain käyttää jo koulussa oppimaani tietoa, mutta opin myös paljon uutta. Esimerkiksi ABB:n RobotStudio-ohjelmistoon olen tutustunut jo koulussa, mutta käyttäessäni sitä simulointiin ja ohjelmointiin opin paljon uusia asioita, jotka varmasti ovat tulevaisuudessa erittäin hyödyllisiä. Konenäkö oli minulle ennestään täysin tuntematon asia, mutta työn aikana minulle selvisi, miten konenäkö toimii ja miten sitä voidaan hyödyntää moneen eri tarkoitukseen.

Konenäön ohjelmoinnin ollessa minulle uutta, on solun ohjelma melko yksinkertainen. Jos aikaa solun valmistamiseen olisi ollut enemmän, olisin voinut toteuttaa konenäön ohjelmoinnin siten, että se tunnistaa kaikki tietyn silmäluvun nopat yhdellä kuvauskerralla. Aikataulu oli kuitenkin tiukka, ja kyseisen toiminnan toteuttaminen olisi vaatinut lisää konenäön ja robotin välisen rajapinnan muokkaamista.

Messujen aikana sain muutaman parannusidean soluun. Yksi parannus oli lisätä käyttöliittymään kenttä, joka ilmoitti käsiteltyjen noppien määrän, kenttä lisättiinkin käyttöliittymään messujen aikana. Myös messuyleisöltä tuli muutamia kehitysideoita soluun. Pääosin olen kuitenkin tyytyväinen soluun, eikä se mielestäni vaatisi mitään suurempia muutoksia.

LÄHTEET

Aalto, H., Heilala, J., Hirvelä, T., Kuivanen, R., Laitinen, M., Lehtinen, H., Lempiäinen, J., Lylynoja, A., Renfors, J., Selin, K., Siintoharju, T., Temmes, J., Tuovila, T., Veikkolainen, M., Vihinen, J. & Virtanen, A. 1999. Robotiikka. Vantaa: Talentum Oyj/MetalliTekniikka.

ABB. 2013a. ABB Production Screen. [www-dokumentti]. ABB Robotics. [Viitattu 18.12.2013]. Saatavissa: http://new.abb.com/products/robotics/application-software/production-screen

ABB. 2013b. IRC5. [www-dokumentti]. ABB Robotics. [Viitattu 16.12.2013]. Saatavissa: http://new.abb.com/products/robotics/controllers/irc5

Brorsson, I., Sjöberg, R. & Liberg, A. 2006. Do-it-yourself robotics. [pdf-julkaisu]. ABB Robotics. [Viitattu18.12.2013]. Saatavissa: http://search.abb.com/library/Download.aspx?DocumentID=9AKK101130D3033&LanguageCode=en&DocumentPartId=&DocumentRevisionId=&Action=Launch

Co-Automation. 2014. Co-Automation yritysesittely. [www-dokumentti]. Co-Automation Oy. [Viitattu 16.1.2014]. Saatavissa: http://co-automation.fi/fi/etusivu

Halinen, M. 2007. Konenäkö robotin ohjauksessa. [pdf-julkaisu]. Aalto-yliopisto. [Viitattu 14.1.2014]. Saatavissa: http://automation.tkk.fi/attach/AS-0-2230/lab3c_teoria.pdf

Hornberg, A. 2008. Handbook of machine vision. Suhl: Wiley-vch.

Keinänen, T., Kärkkäinen, P., Lähetkangas, M. & Sumujärvi, M. 2007. Automaatiojärjestelmien logiikat ja ohjaustekniikat. Helsinki: WSOY Oppimateriaalit Oy.

Kivioja, S. 2014. Toimitusjohtaja. Co-Automation Oy. Henkilökohtainen tiedonanto 3.2.2014.

Lempiäinen, M. 2011. Konenäkö tutuksi viikossa. [www-dokumentti]. [Viitattu 10.1.2014]. Saatavissa: http://konenako.blogspot.com/

Lindholm, M. 2011. ABB IRB 140 robottisolun käyttöönotto. AMK-opinnäytetyö. Tampereen ammattikorkeakoulu. [Viitattu 12.12.2013]. Saatavissa: http://urn.fi/URN:NBN:fi:amk-201104084044

Malm, T. 2008. Robottijärjestelmien uudet turvallisuustekniikat. [pdf-julkaisu]. VTT. [Viitattu 28.12.2013]. Saatavissa: http://www.vtt.fi/inf/julkaisut/muut/2008/RobUudetTurv.pdf

Malm, T., Viitaniemi, J., Marstio, I., Toivonen, S., Koskinen, J., Venho, O. & Salmi, T. 2008. Vuorovaikutteisen robotiikan turvallisuus. Helsinki: Suomen robotiikkayhdistys ry.

Niemi, P. 2011. Konenäkö – lyhyt oppimäärä. [www-dokumentti]. Orbis Oy. [Viitattu 10.1.2014]. Saatavissa: http://www.orbis.fi/konenako-lyhyt-oppimaara

Omron. 2012. Omron FQ2 User's Manual. [pdf-julkaisu]. Omron Oy. [Viitattu 14.1.2014]. Saatavissa: http://downloads.industrial.omron.fi/IAB/Products/Sensing/Vision%20Sensors%20and%20Systems/Easy%20Vision%20Sensors/FQ2/Z326/FQ2_manual_E.pdf

Pilz. 2006. SafetyEYE®. Perfect 3D Protection. [pdf-julkaisu]. Pilz GmbH. [Viitattu 28.12.2013]. Saatavissa: http://www.componente-automatizari.ro/pilz/doc/SafetyEYE.pdf

Soini, A. 2011. Konenäkö. [pdf-julkaisu]. Suomen automaatioseura ry. [Viitattu 10.1.2014] Saatavissa: http://www.automaatioseura.fi/index/tiedostot/Konenako.pdf

Voutilainen, P. 2004. Konenäkö. [www-dokumentti]. Opetushallitus. [Viitattu 10.1.2014]. Saatavissa: http://www03.edu.fi/oppimateriaalit/puutuoteteollisuus/automaatio/konenako/index.html