Teemu Hirvonen

ROBOTTISOLUN JA PALETOINTISOVELLUKSEN PÄIVITYS

Automaatiotekniikan koulutusohjelma
2014

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

ROBOTTISOLUN JA PALETOINTISOVELLUKSEN PÄIVITYS

Hirvonen, Teemu
Satakunnan ammattikorkeakoulu
Automaatiotekniikan koulutusohjelma
Syyskuu 2014
Ohjaaja: Suvela, Timo
Sivumäärä: 29
Liitteitä: 2

Asiasanat: automaatiojärjestelmät, robotiikka, konenäkö

Opinnäytetyön tarkoituksena oli testata ja päivittää robottisolu. Solu sijaitsee Satakunnan ammattikorkeakoulun tekniikka Porin automaatiolaboratoriossa, jossa se on osana harjoitustyöjärjestelmää. Robotille oli aikaisemmin luotu sovellus, joka paletoi erivärisiä kappaleita älykameran ja pulssianturin avulla. Tehtävänä oli 1) testata sovelluksen toiminta, 2) muokata sovellus siistimmäksi, ja 3) vaihtaa solun älykamera uudempaan malliin.

Vanhaa paletointisovellusta ei oltu käytetty noin neljään vuoteen, joten sen toiminnasta ei oltu varmoja. Tänä aikana robottiohjain oli nollautunut, joten osa sen asetuksista oli tehdasarvoissa. Koodista haluttiin yksinkertaisempi, jotta ohjelmaan perehtymätön voisi helpommin ymmärtää sen toiminnan. Hihnakuljettimen seurantaominaisuuden käyttöönotto mahdollistaa jatkossa siihen liittyvien laboratoriotöiden tekemisen. Älykameran vaihtaminen uudempaan malliin teki solusta modernimman ja mahdollistaa parempien harjoitustöiden tekemisen jatkossa.

Työssä käytettiin erityisen paljon RT ToolBox 2-ohjelmistoa, jolla hallittiin robottia. Toinen tärkeä ohjelmisto oli In-Sight Explorer 4.9.0, jolla luotiin älykameran sovellus. Lopullinen sovellus sisälsi näillä ohjelmistoilla rakennetut ohjelmat sekä niiden välisen Ethernet-kommunikoinnin.

Lopputuloksena saatiin modernisoitu älykameralaitteisto sekä selkeämpi ohjelmakoodi sovellukseen. Se sisältää esimerkit kuljettimen seurantaominaisuuden käytöstä, ohjelmistojen välisestä kommunikoinnista ja kappaleiden tunnistamisesta. Näiden tulosten avulla voidaan jatkossa kehittää esimerkiksi laboratorioharjoituksia, joiden avulla opiskelijat voivat kehittää taitojaan.

UPDATING OF A ROBOT CELL AND PALLETIZING APPLICATION

Hirvonen, Teemu
Satakunta University of Applied Sciences
Degree Programme in Automation Engineering
September 2014
Supervisor: Suvela, Timo
Number of pages: 29
Appendices: 2

Keywords: automation systems, robotics, machine vision

The purpose of this thesis was to test and update a robot cell. The cell is located at the automation laboratory of Satakunta university of applied sciences, where it is a part of an exercise system. In the past, one had created an application for the robot which palleted pieces of different color, with the help of a machine vision camera and a pulse encoder. The assignment was to 1) test the function of the application, 2) clean up the application, and 3) change the machine vision sensor of the cell to a later model.

The old palletizing application had not been used in circa four years, so there was no certainty of its function. During this time the robot controller had been reseted, so some of its parameters were in factory values. The code was desired to be more simple, so that a person who was not educated with the program, could more easily understand its function. The implementation of tracking function for the belt conveyor will enable tracking related laboratory exercises to be done for it in the future. The replacement of the machine vision sensor for a later model made the cell more modern and will enable better exercises to be done later.

The most important software during this project was RT ToolBox 2, it was used to control the robot. Another major software was In-Sight Explorer 4.9.0, which was used to create the application for machine vision sensor. The final application included the programs created with these softwares and the Ethernet communication between them.

The final result was a modernized machine vision system and a clearer code for the application. It includes examples of the use of conveyor tracking function, communication between softwares and identification of objects. These results can be used, for example, to create laboratory exercises, which will help students to advance their skills.

SISÄLLYS

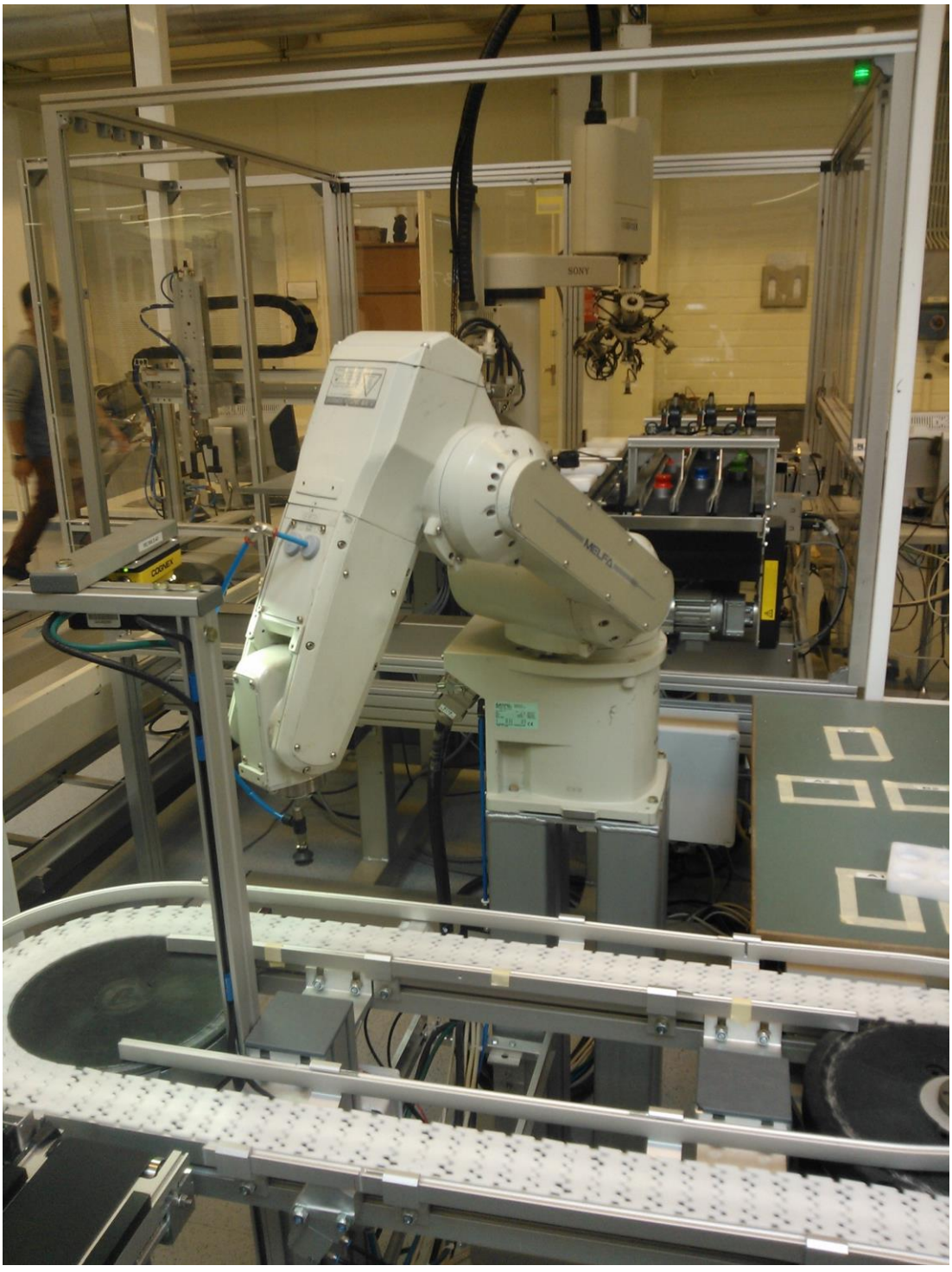
1 JOHDANTO

Opinnäytetyön tarkoituksena oli päivittää Satakunnan ammattikorkeakoulun tiedepuiston kampuksella sijaitseva robottisolu. Päivitykseen kuului 4 vuotta vanhan ohjelman toimivuuden testaus ja uudelleenrakentaminen järkevämmäksi. Ohjelman avulla käsivarsirobotti poimi älykameran ja pulssianturitiedon perusteella erivärisiä alumiinisia kappaleita liikkuvalta kuljettimelta ja siirsi sekä lajitteli ne edelleen seuraavalle kuljettimelle, josta seuraava robottisolu alkoi käsitellä niitä.

Lisäksi järjestelmän älykamera vaihdettiin uudempaan malliin. Tämä tarkoitti myös kameran ohjelmiston vaihtamista ja kappaleen tunnistusohjelman uudelleenluontia uudessa ohjelmointiympäristössä.

Raportissa perehdytään työn eri vaiheisiin ja ongelmatilanteisiin. Tarkoituksena on selostaa tärkeät ja mahdollisesti vaikeuksia tuottavat tilanteet sellaisella tavalla, että lukija pystyy vastaavanlaisessa tilanteessa ratkaisemaan ongelman tämän raportin avulla.

2 JÄRJESTELMÄN ESITTELY



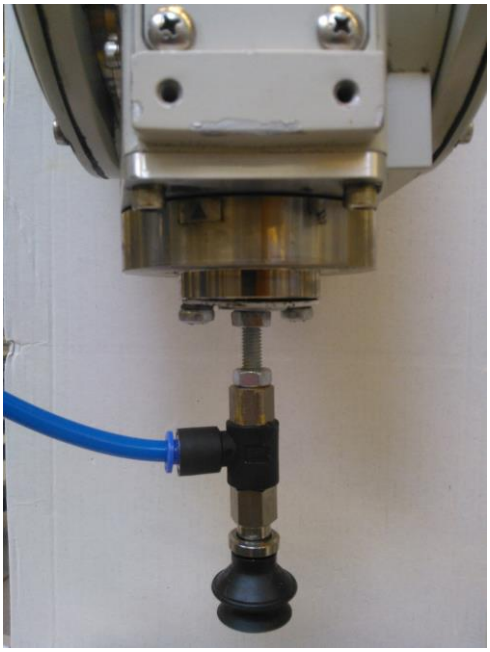
Kuva 1. Robottisolu

Käsiteltävä robottisolu sijaitsee Satakunnan ammattikorkeakoulun tiedepuiston kampuksen automaatiolaboratoriossa. Se on osa suurempaa harjoitussolujärjestelmää,

jossa paletteja käsitellään. Solut toimivat kuitenkin pääsääntöisesti itsenäisesti, toisistaan riippumatta. Tämän vuoksi esittelen vain solun, johon työ kohdistui. Solun toimintatarkoitus tässä työssä esitellään kohdassa 3.1.

2.1 Käsivarsirobotti

Solun mekaanisen työn suorittaa kuusiakselinen yleisrobotti, Melfa RV-6S. Sen ulottuvuussäde on 258-696 mm, jossa se kykenee liikkumaan maksimissaan 9300 mm/s:n vauhdilla. Robotilla käsitellään tässä työssä satojen grammojen painoisia paletteja, mutta se pystyy liikuttamaan kappaleita aina kuuteen kilogrammaan asti (Hakola 2010, 7.) Robotin työkaluna on imukuppitarttuja, joka tarttuu kappaleisiin alipaineen avulla.



Kuva 2. Käsivarsirobotin imukuppitarttuja

2.2 Robotin ohjainyksikkö ja käsiohjain

Kuva 3. Käsiohjain (vasemmalla) ja ohjainyksikkö

Robotin aivot näkyvät kuvassa 3 oikealla. Ohjaimen tyyppi on Melfa CR2B-574. Kaikki robotin käskyt ja asetukset prosessoidaan ohjaimessa. Lisäksi laajennusominaisuudet, kuten pulssianturikortti asennetaan ohjainyksikköön. Ohjaimeen vaikutetaan lähinnä PC:n tai käsiohjaimen kautta, joten se ei itsessään sisällä juurikaan kytkimiä. Ainoat tarvittavat kytkimet ovat virta- sekä tilanvalintakytkin. Lisäksi ohjaimessa on näyttö, joka näyttää robotin tilan.

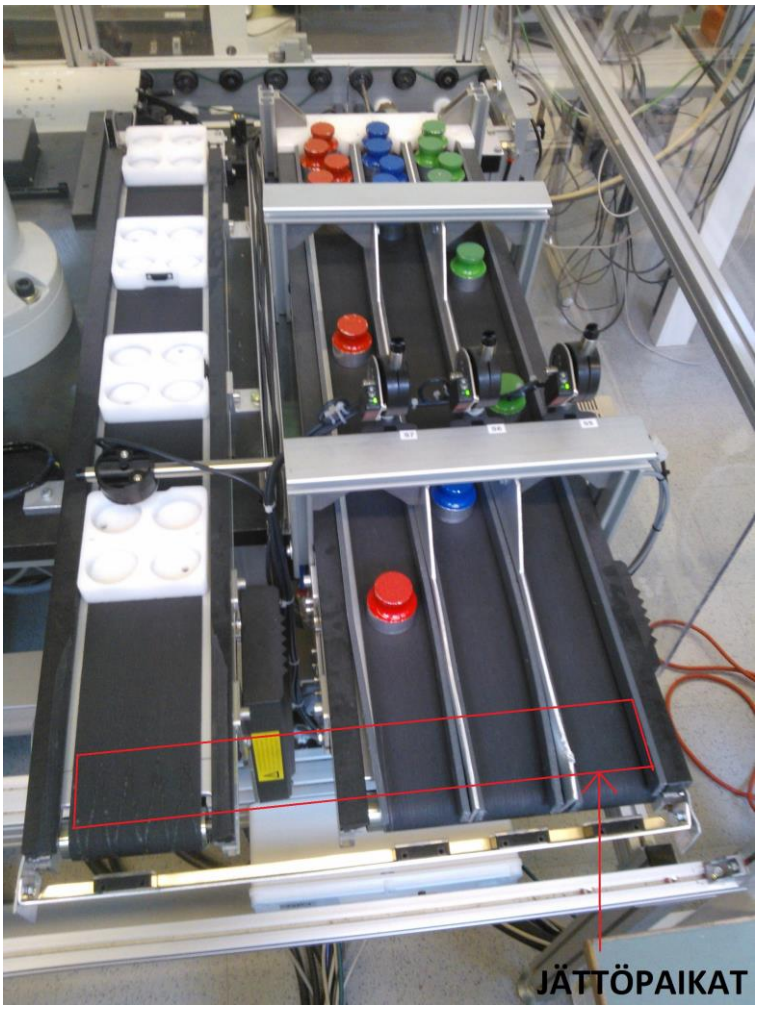
Käsiohjain on tyyppiä R46TB. Se on liitetty ohjainyksikköön kaapelin avulla. Sillä voidaan hallita robottia lähes yhtä laajasti kuin PC:llä. Esimerkiksi koodin muokkaus onnistuu käsiohjaimen kautta, mutta parhaiten se soveltuu asetusten muokkaamiseen sekä ennen kaikkea paikkapisteiden opettamiseen.

2.3 Ulkoinen ohjaustaulu

Kuva 4. Ulkoinen ohjaustaulu

Ohjaustaulun tärkeimmät ominaisuudet ovat valoverho ja kameran valo. Robotin työalueelle mentäessä valoverho laukaisee hälytyksen, joka vie robotin vikatilaan. Tämä hälytys kuitataan Valoverho Start/Reset -napista. Valoverhon tila kannattaa tarkastaa aina robottia käynnistettäessä. Kameran valo -kytkimestä asetetaan älykameraan asennettu valaisin päälle. Kytkin kannattaa pitää aina päällä, kun kameralla otetaan kuvia. Valon unohtaminen vaikuttaa kuvien tummuuteen, mikä voi vaikuttaa kameraohjelman toimivuuteen. Valoa ohjataan pelkästään ohjaustaulusta, joten se tulee sammuttaa aina työskentelyn lopettamisen yhteydessä.

2.4 Jättökuljetin

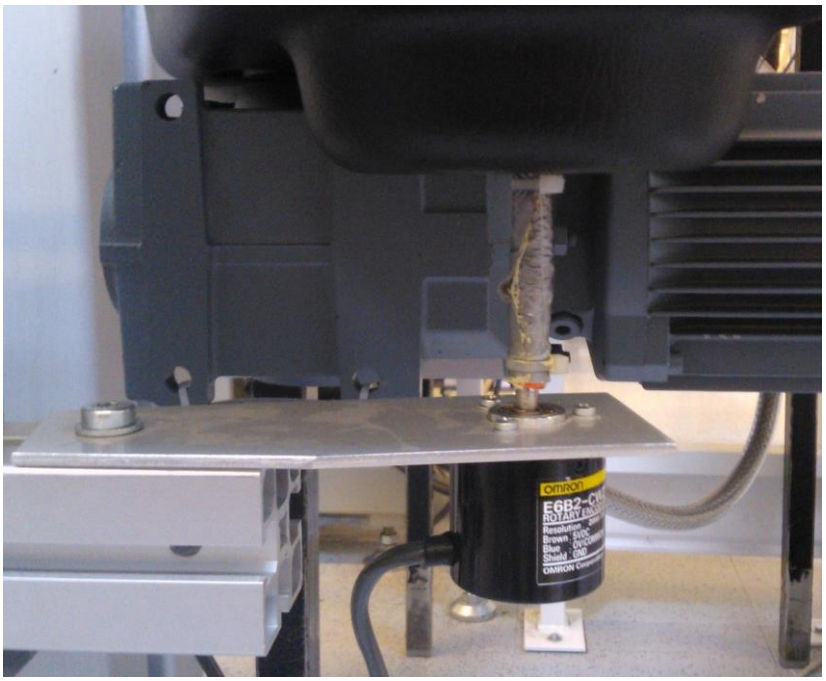


Kuva 5. Jättökuljetin

Jättökuljetin toimii rajapintana työstettävän- ja seuraavan robottisolun välillä. Se sisältää neljä hihnakuljetinväylää, joita pitkin paletit sekä alumiinikappaleet ohjataan uuteen soluun. Jokaisella väylällä on oma valokenno seuraamassa, onko jättöpaikalle saapunut tavaraa. Valokennon aktivoituminen ohjaa kuljettimen päälle ja kappaleet siirtyvät kohti seuraavaa solua.

2.5 Muut komponentit

Järjestelmään kuuluu myös hihnakuljettimia, joiden avulla paletit kuljetetaan ensin soluun ja solussa robotin toiminta-alueelle. Robotin toiminta-alueella sijaitsevaan kuljettimeen on kiinnitetty pulssianturi, joka mahdollistaa paletin seurannan.



Kuva 6. Pulssianturi kiinnitettynä kuljettimen akseliin

Anturi on tyyppiä Omron E6B2-CWZ1X. Se on kiinnitetty kuljettimen akseliin, jolloin se pyörii samalla nopeudella kuin hihnakuljetin. Kuljettimen liikkuessa anturilta saatu lukuarvo kasvaa, jolloin voidaan seurata paletin sijaintia reaaliajassa. Järjestelmään kuuluu lisäksi konenäkökamera, josta kerrotaan luvussa 4.

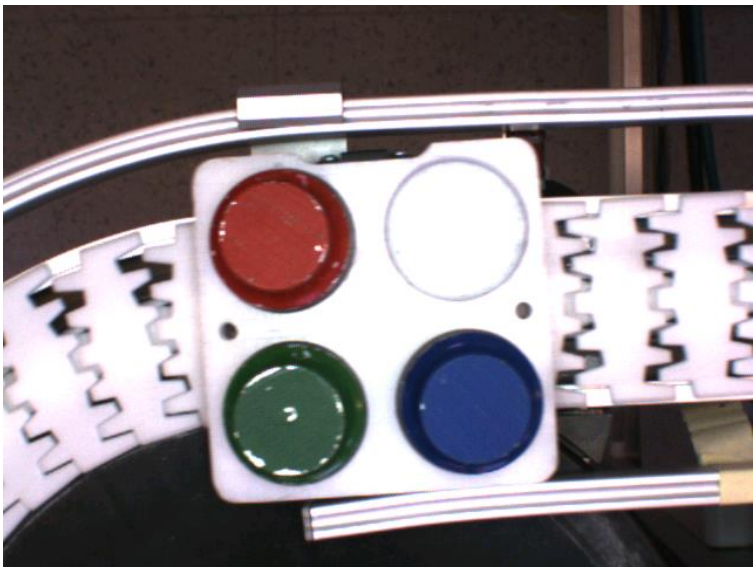
3 VARASTOINTIOHJELMAN TESTAUS

3.1 Varastointiohjelman toiminta

Työn ensimmäinen vaihe oli testata vanha varastointiohjelma, jota oli käytetty viimeksi vuonna 2010. Sen jälkeen robottiohjaimen paristo oli tyhjentynyt, mikä tarkoitti osan ohjaimen parametreista olevan tehdasasetuksilla. Robottia oli tänä aikana käytetty, muttei sellaisissa töissä, missä käytettäisiin apuna älykameraa tai pulssianturia.

Sovelluksen tarkoituksena on siirtää käsivarsirobotin avulla alumiinisia kappaleita ja niihin kuuluvaa palettia liikkuvalta kuljettimelta toiselle kuljetinradastolle, joka toimii rajapintana seuraavalle robottisolulle.

Muoviselle, 100 mm x 100 mm -kokoiselle paletille mahtuu neljä kappaletta. Kappaleita on kolmea eri väriä: sinisiä, punaisia sekä vihreitä. Kuljettimen yläpuolella sijaitseva älykamera ottaa paletista kuvan tämän saapuessa robotin toiminta-alueelle. Tästä kuvasta tunnistetaan kameran ohjelman avulla, onko paletti tyhjä vai onko siinä kappaleita. Jos kappaleita löytyy, niiden sijainti paletilla, sekä väri tunnistetaan. Kameran ohjelmasta lähetetään tieto robottiohjelmaan, jonka perusteella siirrytään oikeaan aliohjelmaan.



Kuva 7. Paletti

Pulssianturitiedon perusteella robotti alkaa seurata palettia ja poimii kappaleen liikkeestä. Ensin varastoidaan kaikki alumiinikappaleet omille kuljettimilleen ja lopuksi tyhjä paletti. Kameralta saadun tiedon perusteella kappale ohjataan oikealle jättökuljettimelle, jolle asetettu anturi tunnistaa kappaleen saapumisen. Tämä tieto käynnistää jättökuljettimen ja siirtää kappaleen kohti seuraavaa robottisolua. Lopuksi robotti palaa kotipisteeseen odottamaan seuraavan kappaleen saapumista.

3.2 Ongelmatilanteet

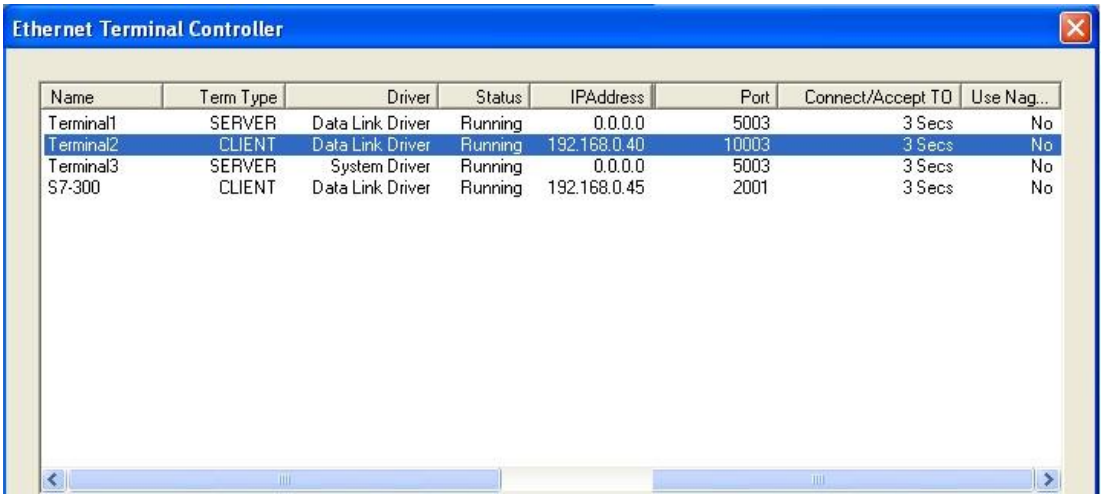
Ohjelman käyttöönotto ei sujunut ongelmattomasti. Pitkän tauon aikana robottiohjaimen parametrit sekä Ethetnet-asetukset olivat vaihtuneet, mikä aiheutti valtaosan ongelmatilanteista. Edes testausvaiheen lopussa, missä luulin korjanneeni kaiken tarpeellisen, en ollut saanut erästä parametria oikein. Tämä johti koko ohjelman uudelleenrakentamiseen siinä luulossa, että alkuperäisessä ohjelmakoodissa olisi vikaa. Vasta projektin viime metreillä ongelma ratkesi ja järjestelmä saatiin toimimaan halutulla tavalla.

Seuraavissa luvuissa kerrotaan kohdatuista vaikeuksista ja pyritään näyttämään niille ratkaisut vastaavien tilanteiden varalta.

3.2.1 Ethernet-asetukset

Jotta älykameralta voidaan lähettää tietoa robottiohjaimelle, tulee ethernet-asetusten olla kunnossa. Asetusten pitää olla oikein kameran ja robotin ohjelmissa, sekä robotin ohjelmointikoodissa. Alkutilanteessa asetukset eivät olleet oikein, joten robotti ei saanut tarvitsemiaan tietoja älykameralta.

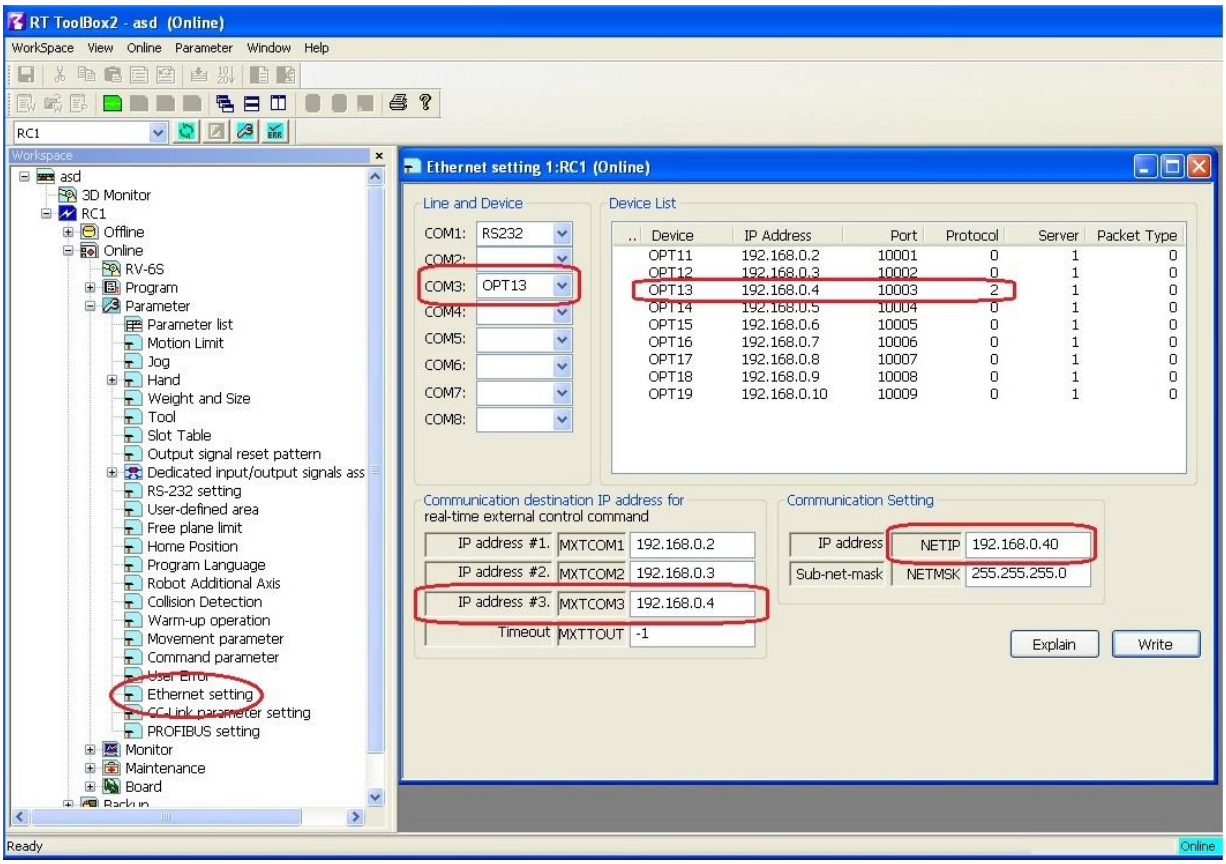
Aloitin valitsemalla Simatic Spectations -ohjelman I/O-valikosta Ethernet Terminal Controller.



Kuva 8. Ethernet Terminal Controller -valikko

IPAddress-kohtaan valitaan robottiohjaimen osoite. Port-kohdan 10003 määräytyy robottiohjaimen asetusten mukaan.

Robottiohjaimen asetukset muutin RT Toolbox 2 -ohjelman avulla. Ethernet-asetuksiin pääsee online-tilassa valitsemalla Parameter-valikosta Ethernet setting (kuva 9).



Kuva 9. Ethernet-asetukset RT ToolBox 2 -ohjelmassa

NETIP-kohtaan valitaan robottiohjaimen osoite. Valitsin linjaksi COM3: OPT13, jolloin käytetään automaattisesti porttia 10003. Tämä portti asetettiin myös Simatic Spectations-ohjelmassa. Linjaa COM3 käytetään, kun portti avataan robotin ohjelmakoodissa. Protocol-kohtaan asetetaan numero 2 (2=Data link, 0=ei proseduuria) (Mitsubishi electric 2002, 19). Lopuksi uudet asetukset kirjoitetaan robottiohjaimelle Write-napista.

Robotin ohjelmakoodissa porttia käytetään kuvan 10 mukaisesti.

```
380 OPEN "COM3:" AS #1   'COM3: is set to port 10003
390 WAIT VKENNO = 1
400 M_ENC(1) = 0
410 INPUT #1,M1,M2,M7,M3
420 CLOSE #1
```

Kuva 10. Ethernet-portin käyttö vanhassa robottiohjelmassa

Rivillä 380 portti avataan, minkä jälkeen rivillä 410 luetaan kameralta saadut tiedot muistipaikkoihin M1, M2, M3 ja M7. M1-M3:en talletetaan noutopisteen x-, y- ja z-koordinaatit, M7 määrittää, mikä aliohjelma kutsutaan tiputusta varten.

3.2.2 Robottiohjelman momenttirajoitukset ja astelukujen muuttuminen

Yhteysasetusten korjaamisen jälkeen pääsin ajamaan robottiohjelmaa. Kuitenkin jo ensimmäiseen pisteeseen siirryttäessä ohjain meni virhetilaan. Ongelma ratkesi suurentamalla robotin akseleille asetettuja momenttirajoituksia, jotka oli asetettu hyvin pieniksi.

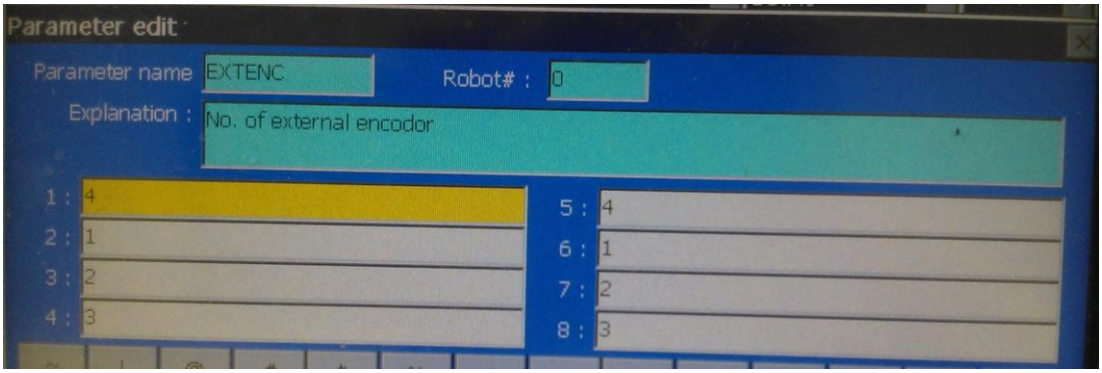
Seuraavaksi erääseen pisteeseen siirryttäessä robotin 2 akselia alkoi kiertyä ääriasentoihin ja ohjain meni jälleen virhetilaan. Monitoroinnissa selvisi, että robotti pyrki kiertämään akseleita tuhansia asteita.

Ongelma liittyi jälleen parametrien tehdasasetuksiin. Tehdasasetuksilla robotti käsittelee sille annettujen pisteiden kiertokulmat radiaaneina. Ongelma korjattiin asettamalla PRG_MDEG-parametri arvoon 1 (1=asteet, 0=radiaanit).

3.2.3 Pulssianturi

Suurimmaksi ongelmaksi muodostui pulssianturin toimimattomuus. Poimintaohjelman seurantaosuudessa käytettiin hyväksi pulssianturitietoa, minkä perusteella kappaleen sijainti mitattiin ja se osattiin poimia oikeasta paikasta. Pulssianturin muuttujan arvo ei kuitenkaan päivittynyt lainkaan.

Ensimmäinen epäilys oli, että robottiohjaimen paristonvaihto olisi nollannut joitain tarpeellisia parametreja, kuten aikaisemmissakin ongelmissa. Maahantuojan avustuksella totesimme tämän olevan totta, löysimme EXTENC-nimisen parametrin, jolla määritetään robotin pulssianturit. Tässä parametrissa on kahdeksan alkiota, tehdasasetuksilla ne on määritelty olemaan (1,2,3,4,1,2,3,4) (Halme sähköposti 14.9.2007).

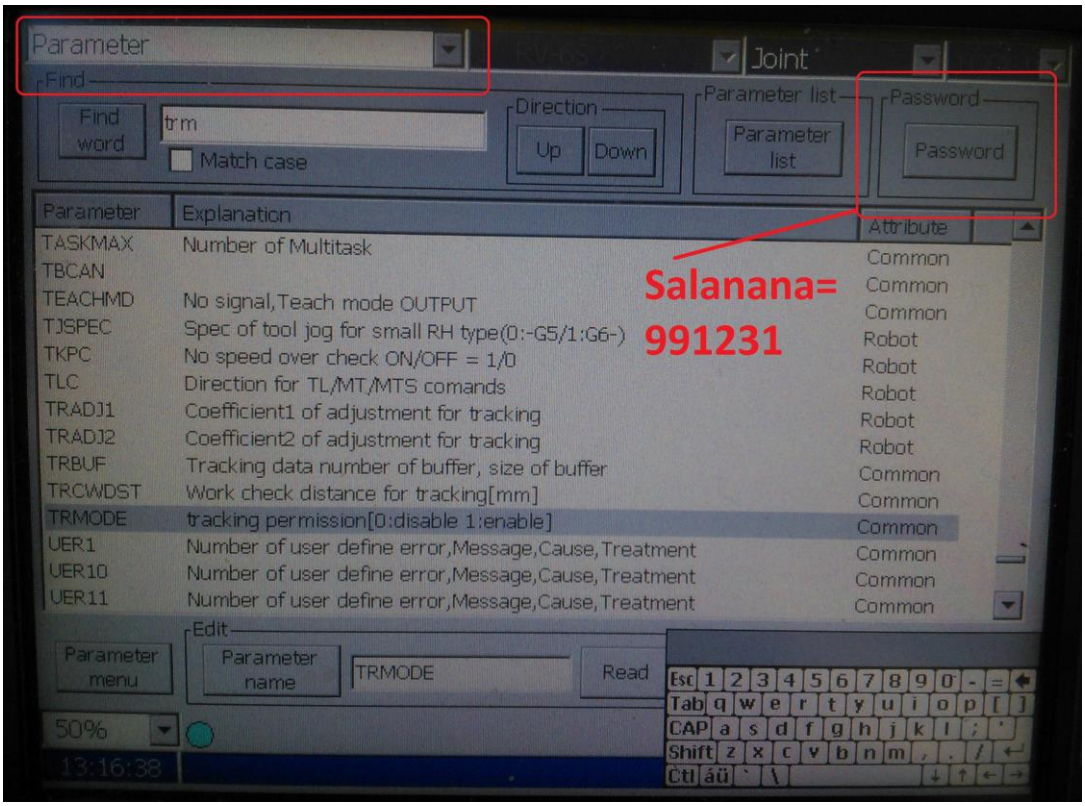


Kuva 11. EXTENC-parametrin asettaminen

Koitin useita eri vaihtoehtoja EXTENC-parametrille, mutta ongelma ei vieläkään ratkennut. Jätin parametrin tehdasasetukselle, jolloin pulssianturin tiedot tallentuvat tässä tapauksessa muuttujaan M_ENC(3). Muuttujan numeron määrää pulssianturin korttipaikka sekä kanava (Halme sähköposti 14.9.2007). Käytössä olevat muuttujat ovat M_ENC, sekä M_ENC(1)-M_ENC(8).

Tässä vaiheessa tiedossa ei ollut muita parametreja, joita pitäisi asettaa. Siksi siirryin testaamaan pulssianturia sähköisesti yleismittarin avulla. Mittausten perusteella anturi näytti olevan kunnossa.

Lopulta vanhasta sähköpostikeskustelusta Beijerin maahantuojan kanssa löytyi lisää asetettavia parametreja. Jotta seuranta saataisiin toimimaan, tulee asettaa TRMECH- ja TRMODE-parametrit, jotka löytyvät vain käsiohjaimen kautta (Halme sähköposti 14.9.2007). TRMODE-parametrin arvoksi tulee asettaa 1. TRMECH-parametrin näkyminen vaatii salasanaparametrin asettamista, ohjeet kuvassa 12.



Kuva 12. TRMECH-parametrin asettaminen

TRMECH-parametrin arvo asetetaan robotin akselien lukumäärän mukaan. Jos käytössä on neljä- tai kuusiakselinen robotti, parametrin arvoksi asetetaan 0. Viiden akselin robotissa arvoksi tulee 1. (Mitsubishi electric 2006, 166.)

Tässä kohtaa pulssianturi alkoi toimia, mutta reaaliaikainen seuranta ei. Muuttuja M_ENC(3) päivittyi, ja robotti tiesi, missä kohtaa kuljetinta paletti kulki, mutta ei lähtenyt seuraamaan kuljetinta, kun seuranta asetettiin päälle. Tästä muodostui koko työn suurin ongelma, sillä kaikki näytti olevan kunnossa, mutta seuranta ei vain toiminut halutulla tavalla.

Tästä johtuen aloin rakentaa seurantaohjelmaa alusta asti uudestaan. Sain pulssianturitietoa hyväksikäyttäen aikaan lopulta kaksi erilaista ohjelmaa, joista molemmat toimivat siinä määrin, että kappale saatiin poimituksi kuljettimelta ja prosessi toimi reaaliaikaista seurantaa lukuun ottamatta niin kuin piti. Vaatimuksena kuitenkin oli seurannan virheetön toimiminen, joten jouduin selvittämään ongelman.

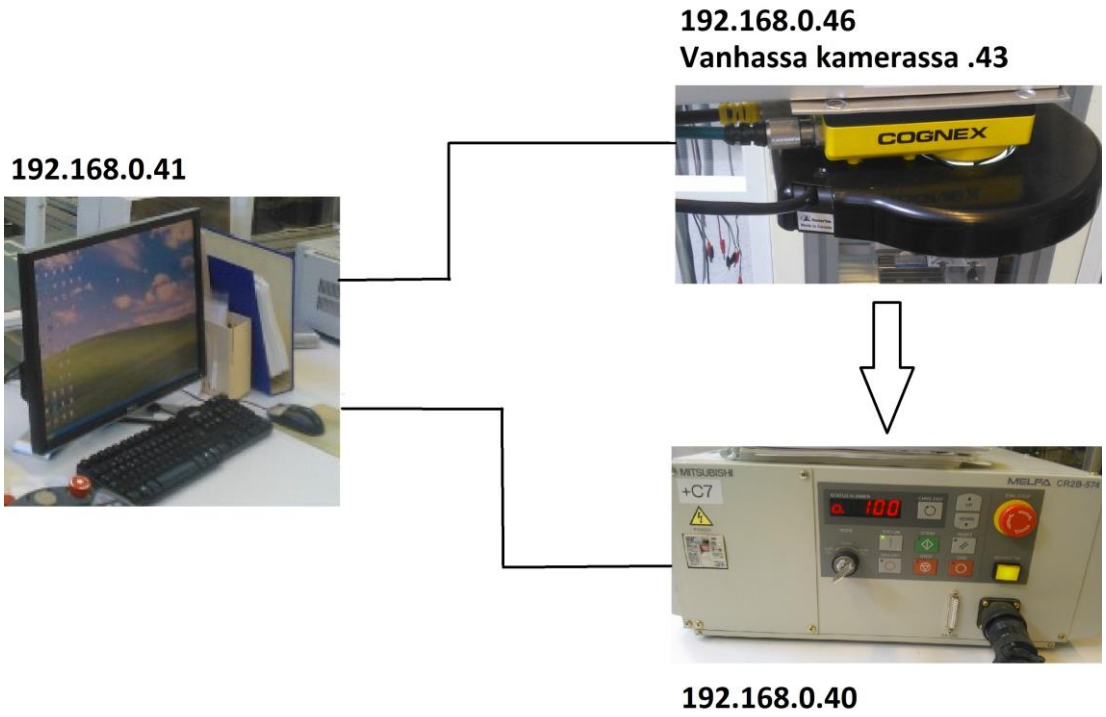
Työskentelin ongelman parissa todella kauan ja kokeilin useita eri käskyjä ja keinoja asian ratkaisemiseksi. Mikään ei kuitenkaan toiminut, joten palasin takaisin alkuun tarkastelemaan EXTENC-parametria (kuva 11). Asetin jokaisen alkion arvoksi 3, jolloin pulssit tallentuvat kaikkiin M_ENC-muuttujiin. Tätä asetusta voidaan käyttää, jos käytössä on ainoastaan yksi pulssianturi (Halme sähköposti 14.9.2007). Seuranta alkoi toimia välittömästi. Jostain syystä parametri piti asettaa tällä tavalla, jotta reaaliaikainen kuljettimen seuraaminen toimisi.

3.3 Yhteenveto asetuksista

Ensimmäisenä suosittelen tarkastamaan robottiohjaimen parametrit. Lähes kaikki parametrit saa muutettua robottiohjelmasta, esim. RT ToolBox 2, valitsemalla online-valikosta parameter-kohdasta parameter list. Osa parametreista löytyy vain käsiohjaimen kautta ja salasanan syöttämisen jälkeen (katso 3.2.3).

PRG_MDEG-parametri asetetaan arvoon 1, jolloin robotti käsittelee koordinaattipisteisiin asetetut kulmat asteina. TRMODE asetetaan arvoon 1, tämä sallii tracking-toiminnon. TRMECH asetetaan robotin akselien mukaan, neljän ja kuuden akselin roboteissa arvoksi tulee 0. Jos käytössä on vain yksi pulssianturi, EXTENC-parametrin jokaiseen alkioon asetetaan 3, jolloin kaikki M_ENC-muuttujat osoittavat käytettävään pulssianturiin.

Parametroinnin jälkeen tarkastetaan yhteysasetukset älykameran ja robotin ohjelmistoista. Osoitteet näkyvät kuvassa 13. Viimeiseksi tarkastuksen kohteeksi jää robotin ohjelmakoodi. Kommunikoinnissa tulee käyttää oikeaa porttia ja robotille annettujen rajoitusten tulee olla sellaisella alueella, että se pystyy työskentelemään normaalisti.



Kuva 13. Laitteiden IP -osoitteet ja yhteydet

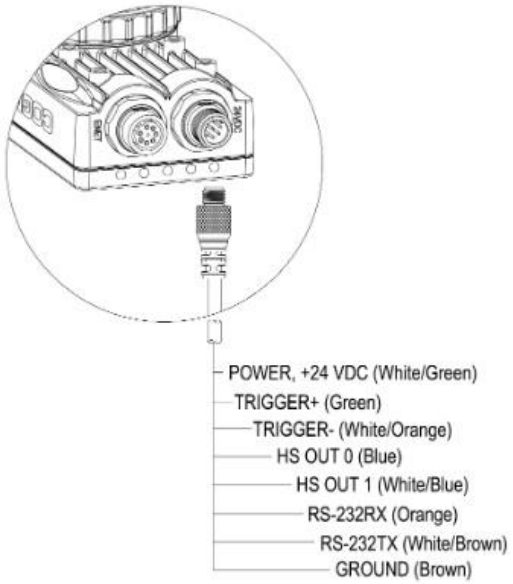
4 KONENÄKÖKAMERAN VAIHTO

Paletteja kuvaava Siemens-merkkinen kamera päätettiin vaihtaa uudempaan Cognex IS5100-C11 -älykameraan. Kamera ottaa 24-bittisiä värikuvia 640x480 resoluutiolla (Cognex corporation 2009, 1). Kamera käyttää omaa In-Sight-ohjelmistoaan.

Kuva 14. Älykamera paikalleenasennettuna

4.1 Älykameran ohjelmiston asennus ja kameran testaus

Aloitin älykameran vaihdon selvittämällä, että laite käyttää aina uusinta saatavilla olevaa versiota In-Sight–ohjelmasta. Latasin ohjelmiston (In-Sight 4.9.0) asennuspaketin laitevalmistajan kotisivuilta ja siirryin suorittamaan asennusta. Ohjelmiston asennus sujui ongelmitta. Jatkoin selvittämällä kameran virtajohdon kytkennän, jotta saisin asentaa testivirtalähteen kameraan.

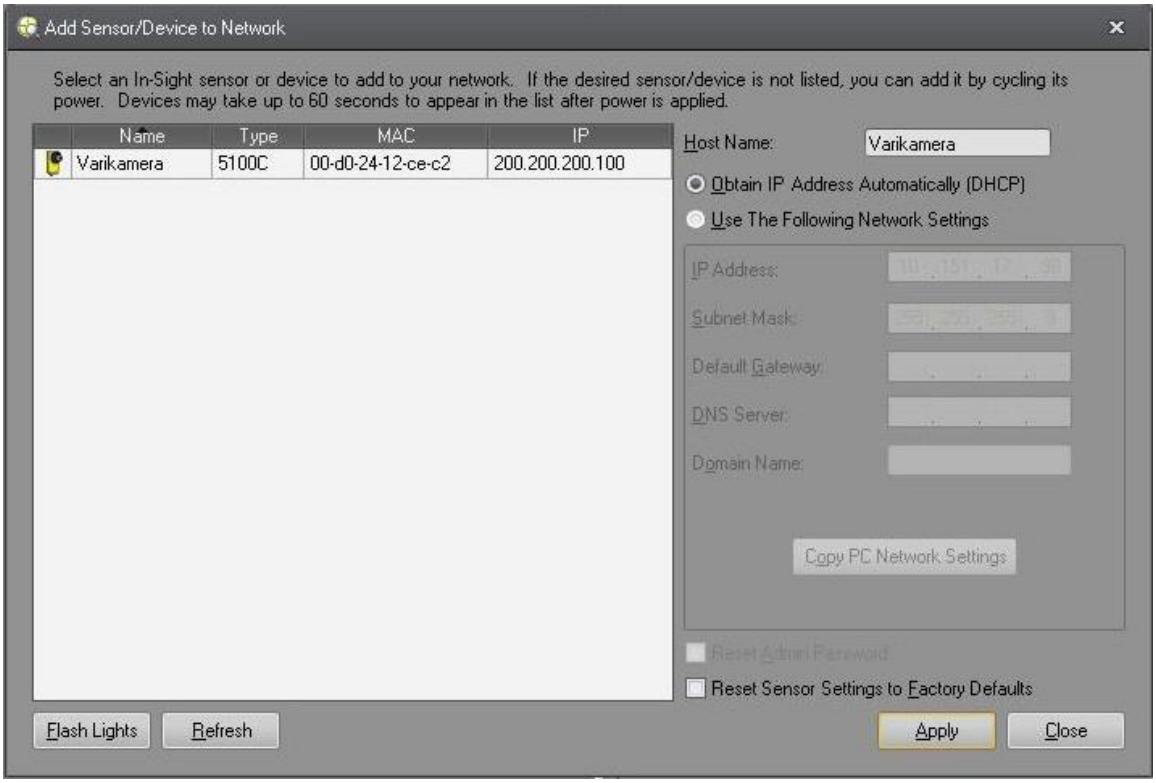


Kuva 15. Älykameran virtajohdon kytkentäkaavio (Cognex corporation 2011)

Kytkin kameran Ethernetkaapelin sekä virtajohdon kuvan 15 ohjeen mukaan ja totesin laitteen olevan päällä virtavalon syttyessä. Lopullisessa asennuksessa kytkentä oli lähes sama kuin edellisellä kameralla. Kytkin 24VDC-, GROUND- ja TRIGGER+ -johtimet samoihin liittimiin, mihin vanha kamera oli kytketty. Lisäksi otin käyttöön TRIGGER- -johtimen, jonka kytkin vapaaseen miinusliittimeen. Trigger on laukaisin, jonka perusteella kamera ottaa kuvan. Tässä tapauksessa tieto tulee kameran alle asetetulta valokennolta ja kuva otetaan kappaleen saapuessa valokennon tunnistusalueelle.

4.2 Kameran ja ohjelmiston välinen konfiguraatio

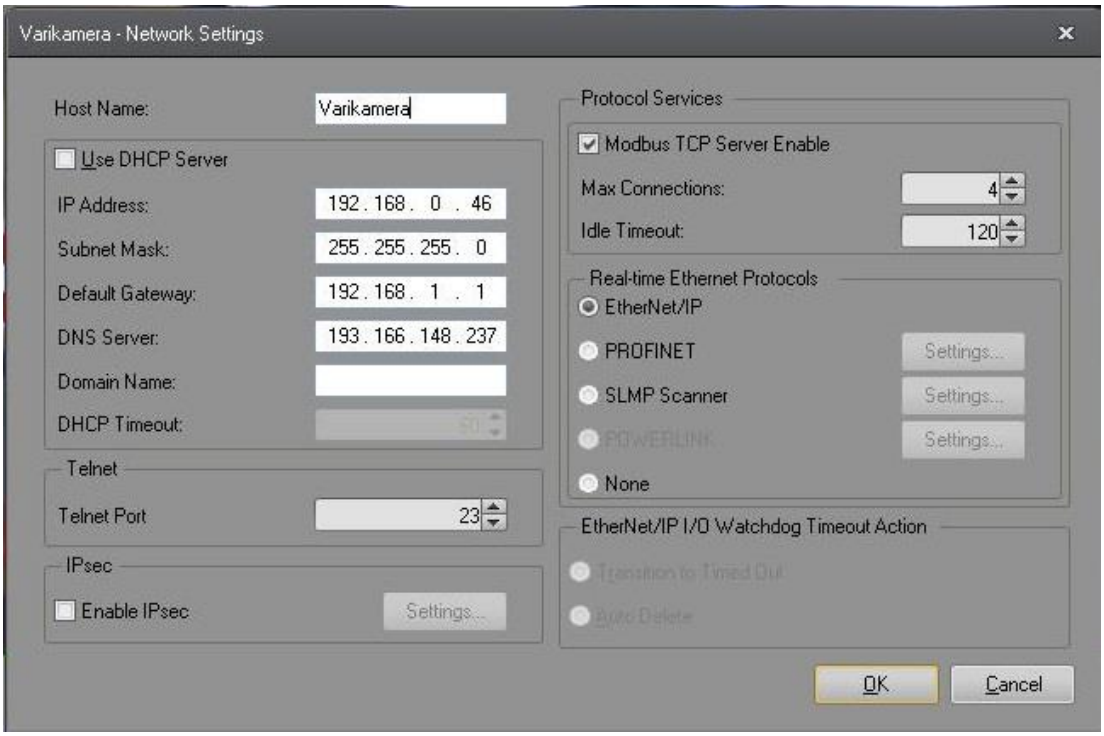
Jotta kameran kuva saadaan näkymään In-Sight-ohjelmistossa, tulee ohjelman verkkoasetukset määrittää oikein. Uusi laite lisätään valitsemalla päävalikon System-kohdasta Add Sensor/Device To Network. Ohjelma alkaa etsiä verkosta laitteita ja näyttää ne kuvan 16 mukaisesti.



Kuva 16. Kameran lisääminen In-Sight -ohjelmassa

Ohjelma löysi lisäämäni kameran ja laittoi oletusarvona kuvassa 17 näkyvät verkkoasetukset.

Näillä asetuksilla yhteys ei kuitenkaan toiminut, vaan jouduin määrittämään asetukset itse. Valitsin Use The Following Network Settings ja kirjoitin tämän kohdan alla näkyvät viisi kohtaa käsin. IP-osoitteen valitsin sellaiseksi, että se oli lähellä muita verkon laitteita. Muut tiedot etsin tietokoneen verkkoasetuksista. Yhteys ei kuitenkaan alkanut toimimaan, sillä IP-osoite ei kelvannut. Koitin useaa eri osoitetta, mutta mikään näistä ei kelvannut. Lopulta päätin valita Copy PC Network Settings. Toiminto haki automaattisesti samat tiedot, jotka olin manuaalisesti kirjoittanut hetkeä aikaisemmin. Valitsin vielä IP-osoitteeksi 196.168.0.46 ja yhteys alkoi toimimaan.



Kuva 17. Älykameran lopulliset verkkoasetukset

4.3 Kappaleentunnistusohjelma

Käytin koko In-Sight -ohjelmassa neljää eri työkalua sekä loogisia lauseita. Ohjelman ensimmäisessä osassa tunnistetaan kappaleen väri ja sijainti.

Kuva 18. Kappaleiden tunnistusosa In-Sight -ohjelmassa

TrainExtractColor-työkalulla määrittelin kolme tunnistettavaa väriä (green, blue, red). Mallikuvasta rajataan alue, joka sisältää tunnistettavaa väriä, jonka jälkeen se lisätään omaan värikirjastoon.

Seuraavaksi lisäsin ExtractColor-työkalun, joka etsii rajatulta alueelta pikseleitä, joita värikirjastoon on tallennettu yllämainitulla työkalulla. Tein jokaiselle paletin paikalle oman tunnistusalueen (slot1-4), josta kolmea väriä etsitään. Pixel count -sarake näyttää, kuinka monta pikseliä kutakin väriä on löydetty. Tämän tiedon perusteella todetaan, minkä värisiä kappaleita paletilla on. Kuvan 18 vaaleansiniseksi rajatulle alueelle kerätään kappaleen väri, joka muutetaan numeeriseksi tiedoksi (0=tyhjä, 1=punainen, 2=sininen, 3=vihreä).

Kuva 19. In-Sight-robottiohjain -rajapinta

Kuvaan 19 olen kerännyt kappaleen sijainnin ja värin numeerisena tietona. Keltaisella alueella muutan nämä tiedot robotille lähteviksi koordinaateiksi. Jokainen numeroarvo keltaisella alueella edustaa offset-arvoa kappaleen poiminta- tai pudotuspisteen nollakoordinaateista millimetreinä. MX, MY ja MZ ovat etäisyyksiä palletin keskipisteestä poimintatilanteessa. M1 ja M2 ovat etäisyydet ensimmäiseltä jättökuljettimelta. Robottiohjaimen koodissa nämä arvot lisätään nouto- ja jättöpisteen koordinaatteihin.

”Datan lähetys robotille:” -kohdassa käytän kahta työkalua, jotta pystyn lähettämään datan robottiohjaimelle. TCPDevice-työkalulla määritetään yhteys, työkaluun on valittu robottiohjaimen IP-osoite sekä käytettävä portti (10003). WriteDevice-työkalulla määritetään lähetettävä data. Ensin määritetään käytettävä kuva (A0) ja yhteys (A39). F35-F39 ovat keltaisen alueen numerotiedot. Ne lähetetään viestissä pilkulla erotettuna (”,”), jolloin robottiohjain vastaanottaa jokaisen luvun eri muistipaikassa. Jos rivillä lukisi ”F35,F36,F37,F38,F39”, robottiohjain vastaanottaisi viestin ”30242833428” yhteen muistipaikkaan.

5 VARASTOINTIOHJELMAN UUDELLEENLUOMINEN

5.1 Johdanto

Luvussa 3 mainitun parametriongelman takia päädyin kirjoittamaan käsivarsirobotin ohjelman alusta alkaen. Tämän johdosta sain rakentaa ohjelman haluamallani tavalla ja muutoksia vanhaan versioon tulikin huomattavasti. Olin tutustunut uusiin käskyihin ja esimerkkiohjelmiin, joista löytyi paljon uusia ideoita käytettäväksi. Toki myös vanhasta ohjelmasta löytyi paljon käytettäviä ominaisuuksia. Käytin ohjelman luomiseen RT Toolbox 2 -ohjelmistoa, jonka ohjelmointikielenä on Melfa basic 4.

5.2 Kalibrointi

Jotta robotti pystyy seuraamaan kuljetinta, sen koordinaatisto pitää kalibroida pulssianturin kanssa. Tässä tapauksessa kuljetin liikkuu lähes x-akselin suuntaisesti. Käytin kalibrointiin valmista Mitsubishin ohjelmaa (Liite 1), jota noudattamalla päästään vaivattomasti tarkkaan tulokseen. Robotti ajetaan aluksi merkin päälle, joka sijaitsee kuljettimen alkupäässä. Ohjelma ottaa pulssianturitiedon sekä robotin koordinaatit ylös. Seuraavaksi kuljetinta ajetaan eteenpäin, minkä jälkeen robotti ajetaan jälleen merkin päälle. Tästäkin paikasta otetaan pulssianturitieto ja robotin koordinaatit ylös, joiden perusteella ohjelma suorittaa kalibroinnin. Pisteiden välin tulee olla kuljettimen suoralla osuudella ja mahdollisimman etäällä toisistaan luotettavimman kalibroinnin varmistamiseksi.

5.3 Ohjelman toiminta

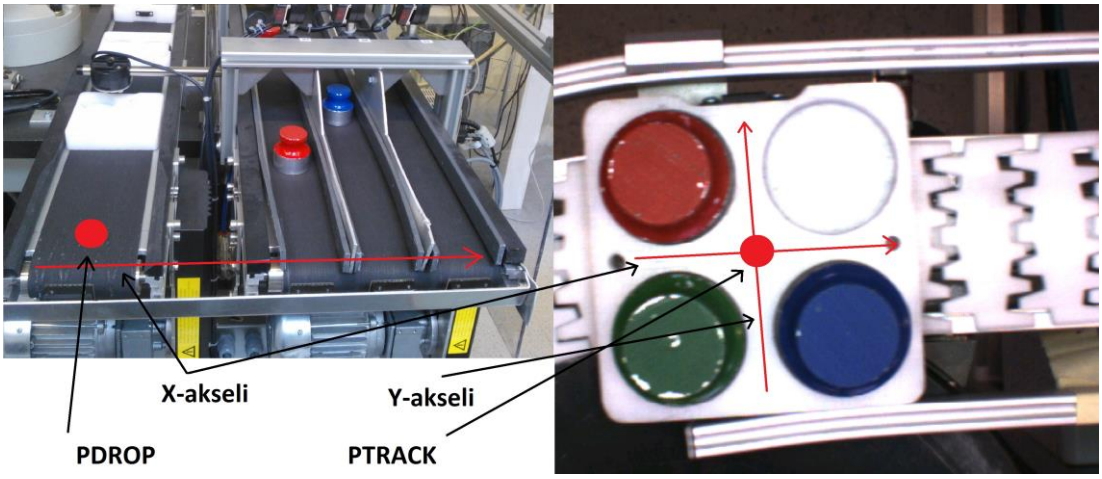
Tässä kappaleessa selostetaan päivitetyn robottiohjelman toiminta vaiheittain. Ohjelmakoodi löytyy liitteestä 2.

Riveillä 10–190 robotille alustetaan sovelluksessa käytettävät parametrit, koska samalla robottiohjaimella ajetaan usein myös toisia ohjelmia. Tämän vuoksi osa asetuksista saattaa olla eri tavalla kuin viimeksi ohjelmaa ajettaessa. Robotille asetetaan

turvalliset momenttirajoitukset sekä ensimmäisen kerran kotipisteeseen ajettaessa käytetään todella hidasta nopeutta, jotta mahdollinen törmäys ei aiheuttaisi vahinkoa. Hihnakuljettimet asetetaan päälle ja suurimmalle nopeudelle sekä määritetään tarvittaville IO-tiedoille muuttujat (imu, vkenno). Rivillä 180 määritetään toiminta, jonka perusteella siirrytään paletintunnistusohjelmaan. Toiminnan laukaisijana toimii kameran alle asetettu valokenno. ACT-käskyn ollessa aktiivisena ja laukaisutoiminnon mennessä päälle, siirrytään aliohjelmaan siitä huolimatta, mitä ohjelmariviä kyseisellä hetkellä suoritetaan.

Riveillä 200–260 ollaan työkierron alussa. Robotti siirretään kotipisteeseen kameran läheisyyteen odottamaan paletin saapumista valokennolle. Ensin varmistetaan jättökuljettimien olevan tyhjänä kappaleiden törmäämisten estämiseksi ja lopuksi uuden kappaleen poiminta sallitaan asettamalla ACT -toiminto päälle.

Valokennon aktivoituessa hypätään riville 650, josta jatketaan riville 810 asti. Vaihe aloitetaan hakemalla paletin tiedot kameraohjelmalta. Rivillä 670 avataan portti ”COM 3”, johon Ethernet-määrittelyt on luotu. Seuraavalla rivillä tiedot talletetaan viiteen robottiohjelman muuttujaan. Nämä muuttujat antavat offset-tietona kappaleen koordinaatit nostamista ja jättämistä varten. Noutopisteen nollapaikka (PTRACK) on paletin keskellä ja jättöpaikan (PDROP) paletille tarkoitetulla jättökuljettimella (kuva 20). Rivillä 740 PTRACK asetetaan pisteeksi, jota aletaan seurata tracking-toiminnolla. Lopuksi tiedot kirjoitetaan seurantapuskuriin ja palataan riville 290.



Kuva 20. Offset-koordinaatiston suunta sekä nouto- ja jättöpisteet

Työkiertojen välillä ohjelma jää odottamaan valokennon aktivoitumista riveille 280–290. Tässä tilanteessa seurantabufferi on tyhjä (M_TRBFCT=0), joten ohjelma jää silmukkaan kunnes valokenno aktivoi ACT-käskyn ja ohjelma siirtyy tunnistusvaiheeseen (yllä oleva kappale). Tunnistusvaiheen jälkeen palataan riville 290, mutta nyt seurantapuskurissa on tietoa, joten työkierto jatkuu.

Riveillä 340–630 suoritetaan varsinainen paikoitus. Ensin asetetaan seuranta päälle, jolloin robotti alkaa liikkua synkronoidusti hihnakuljettimen kanssa. Rivillä 370 asetetaan robotin liikenopeus pieneksi, sillä muutoin ohjain menee vikatilaan. Tartuntatyökalu ajetaan poimittavan kappaleen ylle, jonka jälkeen asetetaan imu päälle. Seurantaa jatketaan vielä jonkin matkaa ja kappale nostetaan suoraan ylös. Riville 460 saavuttaessa tarkastetaan, onko kappale poimittu onnistuneesti. Jos poiminta onnistui, kappale viedään jättöpisteelle, jos ei, niin palataan kotipisteeseen.

5.4 Muutokset ja eroavaisuudet vanhaan ohjelmaan nähden

Näkyvin ero vanhaan ohjelmaan on aliohjelmien käyttö. Vanhassa ohjelmassa kappaleen tunnistamisen jälkeen tämän tiedot talletetaan muistipaikkaan. Muistipaikan tiedon perusteella siirrytään erillisiin tiedostoihin luotuihin aliohjelmiin. Aliohjelmia on viisi kappaletta, jokaista väriä varten ja kaksi ohjelmaa palettia varten. Jokaisessa ohjelmassa on luotu erikseen paikkamuuttuja kappaleen pudottamista varten. Uudessa versiossa koko koodi on samassa tiedostossa. Nouto- ja jättöpisteen koordinaatit määritetään offset-tietoina paletin keskeltä ja ensimmäiseltä jättökuljettimelta. Tällöin riittää, että koordinaattitiedot summataan pääohjelmassa määriteltyihin pisteisiin. Tämä tekee koodista ymmärrettävämmän, ja säästytään erillisten aliohjelmien tekemiseltä.

Lisäksi uudessa versiossa on useita käskyjä, joita vanhasta versiosta ei löydy sekä muita pieniä eroavaisuuksia. Käytin pelkän valokennolta saadun tiedon sijasta ACT-käskyä tunnistusosioon siirtymisessä, sillä voidaan varmistaa, että toiminto tapahtuu vain haluttuna aikana. Ennen seurannan aloittamista uudessa versiossa tarkistetaan M_TRBFCT-toiminnolla, että seurantabufferissa on dataa. Lisäksi vanhassa versiossa käytettiin hyväksi pulssianturin lukemaa paletin sijainnin määrittämisessä, kun

taas nykyversiossa tarkastellaan suoraan x-koordinaattia. Tällä tavoin käyttäjän ei tarvitse laskea pulssianturin arvon ja x-koordinaatin muutossuhdetta käsin ja koodi on helpommin ymmärrettävissä.

6 YHTEENVETO

Opinnäytetyö eteni melko hitaasti ja eräiden ongelmien kanssa käytettiin paljon aikaa. Ongelmatilanteiden systemaattinen analysointi olisi varmasti ollut tehokkaampi vaihtoehto, mutta jokaiseen tilanteeseen löytyi lopulta ratkaisu. Kaikki vaikeudet liittyivät jollain tavalla robottiin, joten suurin osa työajasta meni sen parissa. Älykameran kanssa työskentely ei ollut yhtä tuttua kuin robotin, mutta se ei silti aiheuttanut vaikeuksia. Fyysinen asennus, ohjelmiston asennus ja ohjelman teko sujuivat kaikki ongelmitta.

Lopullinen sovellus onnistui todella hyvin. Turvallisuus, helppokäyttöisyys ja luotettavuus on huomioitu sovellusta rakennettaessa. Halusin tehdä robotin koodista melko lyhyen ja helposti ymmärrettävän, mikä mielestäni saavutettiin. Toisaalta tämä tarkoittaa sitä, että älykameran ohjelmassa on tehty toimenpiteitä, joita olisi voitu tehdä myös robotin koodissa. Älykameran robotille lähettämät tiedot ovat suoraan millimetreinä, jolloin ne on helppo laskea yhteen nouto- tai jättöpisteen koordinaattien kanssa. Toinen vaihtoehto olisi ollut lähettää robotille pelkkä numerotieto, joka kertoisi kappaleen värin sekä sijainnin paletilla. Tällöin koordinaatit pitäisi laskea robotin ohjelmassa, mikä monimutkaistaisi koodia. Toisessa vaihtoehdossa muutosten tekeminen saattaisi olla helpompaa, mutta tarkoituksena ei ollut tehdä muuteltavaa sovellusta vaan valmis paketti. Ohjelman tarkoituksena oli myös esitellä seurantaominaisuus mahdollisimman selkeästi tulevaisuuden projekteja varten, minkä takia robotin koodi oli hyvää pitää lyhyenä ja selkeänä.

LÄHTEET

Cognex corporation. 2009. In-Sight 5100,5100C,5401,5400C,5403,5400 Specification Sheet. Viitattu 4.9.2014. http://www.cognex.com/support/downloads/File.aspx?d=1839&langtype=2057

Cognex corporation. 2011. In-Sight® 5000 Series Vision System Installation Manual. Viitattu 16.9.2014. http://clpa.eu/portals/57/VEx/Docs/Cognex_In-Sight%205000_Installation%20Manual.pdf

Hakola, A. 2010. Käsivarsirobotin lisäominaisuuksien käyttöönotto ja laboratoriotyöohjeen laatiminen. AMK-opinnäytetyö. Satakunnan ammattikorkeakoulu.

Halme, R. Lisäohjeita/Tracking. Vastaanottaja: Jorma Tuomela. Lähetetty 14.9.2007 klo 19:51:18. Viitattu 5.9.2014.

Mitsubishi electric. 2002. Ethernet interface CRn-500 series. Viitattu 4.9.2014. http://int76.ru/upload/iblock/73a/73acf48da6755f7fdee5a04301b1fc55.pdf

Mitsubishi electric. 2004. CRn-500 conveyor tracking function. Viitattu 16.9.2014.

Mitsubishi electric. 2006. Robot advanced training. Viitattu 5.9.2014. http://www.google.fi/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCAQFjAA&url=http%3A%2F%2Fsuport.siriustrading.ro%2F03.Training%2F06.RI%2F2.Cursuri%2F060810-Advanced.ppt&ei=HK0JVIroG8HmavOcgeAI&usg=AFQjCNH0EjgsnyqgieW2gAXdte2aeV_xDA&sig2=AQ6305SUl4bk1Dhujqu7Dg&bvm=bv.74649129,d.d2s&cad=rja

PULSSIANTURIN KALIBROINTIOHJELMA

```
10 '#############################################################
20 '# Conveyor Tracking Calibration processing between robot-conveyors
30 '# Classification of program : RB-CV Calibration Program
40 '# Version : A0a
50 '# COPYRIGHT : MITSUBISHI ELECTRIC CORPORATION.
60 '#############################################################
70 '(1)Stick the marking seal to the upper course of the conveyor
80 '(2)Move the robot to center of the seal
90 MX10EC1#=M_ENC 'Get the encoder data 1
100 PX10PS1=P_FBC 'Get the position 1
110 '
120 '(3)Move the robot up
130 '(4)Move the conveyor forward
140 '(5)Move the robot to center of the seal again
150 MX10EC2#=M_ENC 'Get the encoder data 2
160 PX10PS2=P_FBC 'Get the position 2
170 P_101(1)=PX10PS2
180 '(6)Move the robot up
190 '(7)Execute by step execution till END
200 GOSUB *S10ENC 'Calculation processing of P_ENCDLT
210 P_ENCDLT=PY10ENC
220 END
230 '
240 '##### Calculation processing of P_ENCDLT #####
250 'MX10EC1:Encoder data 1
260 'MX10EC2:Encoder data 2
270 'PX10PS1:Position 1
280 'PX10PS2:Position 2
290 'PY10ENC:Value of P_ENCDLT
300 *S10ENC
310 M10ED#=MX10EC2#-MX10EC1#
320 IF M10ED#>800000000.0 THEN M10ED#=M10ED#-1000000000.0
330 IF M10ED#<-800000000.0 THEN M10ED#=M10ED#+1000000000.0
340 PY10ENC.X=(PX10PS2.X-PX10PS1.X)/M10ED#
350 PY10ENC.Y=(PX10PS2.Y-PX10PS1.Y)/M10ED#
360 PY10ENC.Z=(PX10PS2.Z-PX10PS1.Z)/M10ED#
370 PY10ENC.A=(PX10PS2.A-PX10PS1.A)/M10ED#
380 PY10ENC.B=(PX10PS2.B-PX10PS1.B)/M10ED#
390 PY10ENC.C=(PX10PS2.C-PX10PS1.C)/M10ED#
400 RETURN
PX10PS1=(+255.273,-209.923,+24.897,+0.000,+0.000,+83.560)(0,0)
PX10PS2=(+256.937,+162.904,+24.999,+0.000,+0.000,+83.560)(0,0)
PY10ENC=(+0.000,+0.050,+0.000,+0.000,+0.000,+0.000,+0.000,+0.000)
```

(Mitsubishi electric 2004)

PÄIVITETTY PALETOINTIOHJELMA SEURANTA 2014

```
10 SERVO ON
20 M_OUT(9)=1 'Kuljettimet päälle
30 M_OUT(10)=1
40 M_OUT(8)=1
50 DEF IO IMU = BIT,6 'Tarttujan imun ohjaus
60 DEF IO VKENNO = BIT,7 'Kuljettimen valokenno, kameran trigger
70 DEF INTE MAXX,MINX
80 MAXX%=250 ' X-koordinaatin maksimiarvo (ei käytössä)
90 MINX%=-370 ' X-koordinaatin minimiarvo seurantaa aloitettaessa
100 TORQ 1,50 'Turvamomenttirajat robotin akseleille (%:a maksimista)
110 TORQ 2,50
120 TORQ 3,50
130 TORQ 4,50
140 TORQ 5,50
150 TORQ 6,50
160 ACCEL 60,60 'Robotin kiihdytyksien säätö (%:a maksimista)
170 TRCLR 1 'Seurantabufferin tyhjennys
180 DEF ACT 1,VKENNO=1 GOSUB *TUNNISTA 'Palletin tullessa valokennolle siirrytään tunnistusohjelmaan
190 OVRD 15 'Ohjelmaa käynnistäessä siirrytään aloituspisteeseen hitaalla nopeudella
200 *START
210 MOV PSTART 'Kotipiste
220 OVRD M_NOVRD 'Asetetaan normaalit ajonopeudet
230 JOVRD M_NJOVRD
240 WAIT M_IN(10)=1 'Odota, että jättökuljettimet ovat vapaana
250 WAIT M_IN(11)=1
260 ACT 1=1 'Sallitaan tunnistusohjelmaan siirtyminen
270 '-----------------------------------------
280 *NOUTO
290 IF M_TRBFCT<1 THEN *NOUTO 'Odotetaan, että seurantabufferiin tulee dataa
300 TRRD PTRACK,M_ENC 'Luetaan seurantabufferin data
310 *SAFE_X
320 PC=TRWCUR(1,PTRACK,M_ENC) 'Otetaan palletin sijaintitieto talteen turvarajan tarkastamista varten
330 IF PC.X<MINX% THEN *SAFE_X 'X-suuntaisen turvarajan tarkastus
340 '-------------------KAPPALEEN VARASTOINTI-------------------------
350 TRK ON,PTRACK,M_ENC 'Aloitetaan seuranta
360 CNT 1,15,15 'Asetetaan jatkuvaliikkeinen liiketila päälle robotin liikkeiden pehmentämiseksi
370 OVRD 4
380 MOV PTRACK
390 IMU=1 'Asetetaan tarttujan imu päälle
```

```
400 WAIT P_CURR.X>150 'Seurataan kappaletta kuljettimen suoran loppupäähän asti
410 TRK OFF 'Lopetetaan seuranta
420 OVRD 30
430 MVS ,-150
440 CNT 0 'Asetetaan normaali liiketila päälle
450 OVRD 70
460 IF IMU=1 THEN 'Jos kappaleeseen tarttuminen onnistui, siirrytään varastointiin
470 MOV PVALI2
480 MOV PVALI1
490 MOV PDROP, -100
500 OVRD 10
510 MVS PDROP
520 IMU = 0 'Pudotetaan kappale jättöpisteeseen
530 DLY 0.5
540 MVS PDROP,-150
550 OVRD 70
560 MOV PVALI1
570 MOV PVALI2
580 TRCLR 1
590 ELSE 'Palataan alkuun jos kappaletta ei saada poimittua
600 IMU=0
610 TRCLR 1
620 ENDIF
630 GOTO *START 'Aloitetaan työkierto alusta
640 '----------------------------------------
650 *TUNNISTA
660 ACT 1=0 'Estetään ohjelman keskeytys työkierron aikana
670 OPEN "COM3:" AS #1 'Avataan yhteys kameralle
680 INPUT #1,MX,MY,MZ,M1,M2 'Talletetaan palletin tiedot kameralta
690 CLOSE #1 'Suljetaan yhteys
700 PTRACK=PSENS
710 PTRACK.X=PTRACK.X+MX 'Noutopisteen määrittely
720 PTRACK.Y=PTRACK.Y+MY
730 PTRACK.Z=PTRACK.Z+MZ
740 TRBASE PTRACK 'Asetetaan seurannan aloituspiste
750 PDROP=PDDROP
760 PDROP.X=PDROP.X+M1 'Jättöpisteen määrittely
770 PDROP.Z=PDROP.Z+M2
780 WAIT VKENNO=0 'Odotetaan, että palletti ohittaa valokennon
790 M_ENC=0 'Nollataan pulssianturi
800 TRWRT PTRACK,M_ENC 'Kirjoitetaan data seurantabufferiin
810 RETURN 0 'Palaa keskeytysriville (NOUTO)
820 END
```