Saimaan ammattikorkeakoulu
Tekniikka Lappeenranta
Tietotekniikan koulutusohjelma
Tietojärjestelmien kehitys

Esa Laine

Ristinolla-robotti

Opinnäytetyö 2016

Tiivistelmä

Esa Laine
Ristinolla-robotti, 23 sivua
Saimaan ammattikorkeakoulu
Tekniikka Lappeenranta
Tietotekniikan koulutusohjelma
Tietojärjestelmien kehitys
Opinnäytetyö 2016
Ohjaajat: Jouni Könönen, Saimaan ammattikorkeakoulu

Opinnäytetyön tavoite oli toteuttaa Ciros Studio -simulointiohjelmassa toimiva ristinollaa pelaava robotti sekä mallintaa ja tulostaa automaatiolaboratorioon pelialusta Mitsubishi Melfa RV-2AJ-robotille. Lisäksi tavoitteena oli suunnitella pelilaudan anturointi ja testata peli robotilla. Opinnäytetyön asiakkaana oli Saimaan ammattikorkeakoulun konetekniikan osasto.

Virtuaaliympäristö toteutettiin Ciros Studio -ohjelmistolla, käyttäen pohjana valmista harjoitustehtävän virtuaaliympäristöä. Pelilogiikka ohjelmoitiin Melfa Basic IV -ohjelmointikielellä.

Lopputuloksena saatiin toimiva virtuaaliympäristö sekä pelialusta. Pelilaudan valmistukseen, anturointiin ja robottitestaukseen ei aika riittänyt.

Asiasanat: Robotiikka, Mitsubishi Melfa, Ciros Studio

Abstract

Esa Laine
Tic-Tac-Toe Robot, 23 pages
Saimaa University of Applied Sciences
Technology Lappeenranta
Degree Programme in Information Technology
Information Systems
Bachelor´s Thesis 2016
Instructors: Jouni Könönen

The purpose of this thesis was to create a virtual environment of a Tic-Tac-Toe robot for Ciros Studio simulation software and design a board for Mitsubishi Melfa RV-2AJ robot in the automation lab, add sensors to board and test the robot.

The virtual environment was created from existing course material with Ciros Studio. The game logic was programmed with Melfa Basic IV programming language.

As a result of this thesis a functional virtual environment with game logic was created. There was not enough time to finish the game board and test the game with the robot.

Keywords: Robotics, Mitsubishi Melfa, Ciros Studio

Sisällys

Termit ja käsitteet

Ciros Studio	Automatisoitujen järjestelmien 3D suunnittelu ja simulointiohjelmisto.
Mitsubishi	Japanilainen yritysryhmä, joka valmistaa mm. autoja ja elektroniikkatuotteita.
Melfa Basic IV	Robottien ohjelmointiin käytetty ohjelmointikieli.
Melfa RV-2AJ	Mitsubishin valmistama robotti

1 Johdanto

Tämän opinnäytetyön tavoitteena on suunnitella ja toteuttaa robotille Ristinolla-pelilogiikka, joka toimii virtuaaliympäristössä ja oikealla robotilla. Työssä käytetään pohjana olemassa olevaa virtuaaliympäristöä, joka muokataan pelille sopivaksi. Automaatiotekniikan laboratorioon mallinnetaan pelilauta, joka valmistetaan 3D-tulostimella ja anturoidaan.

Virtuaaliympäristö on tarkoitus tallentaa osaksi konetekniikan osaston robotiikka-kurssin Moodlen materiaalia. Automaatiotekniikan laboratorion pelaavaan robotin on tarkoitus olla osa osastolla vierailevien ryhmien esittelykierrosta.

Asiakkaana työssä on Saimaan ammattikorkeakoulun konetekniikan osasto.

2 Käytetyt työkalut

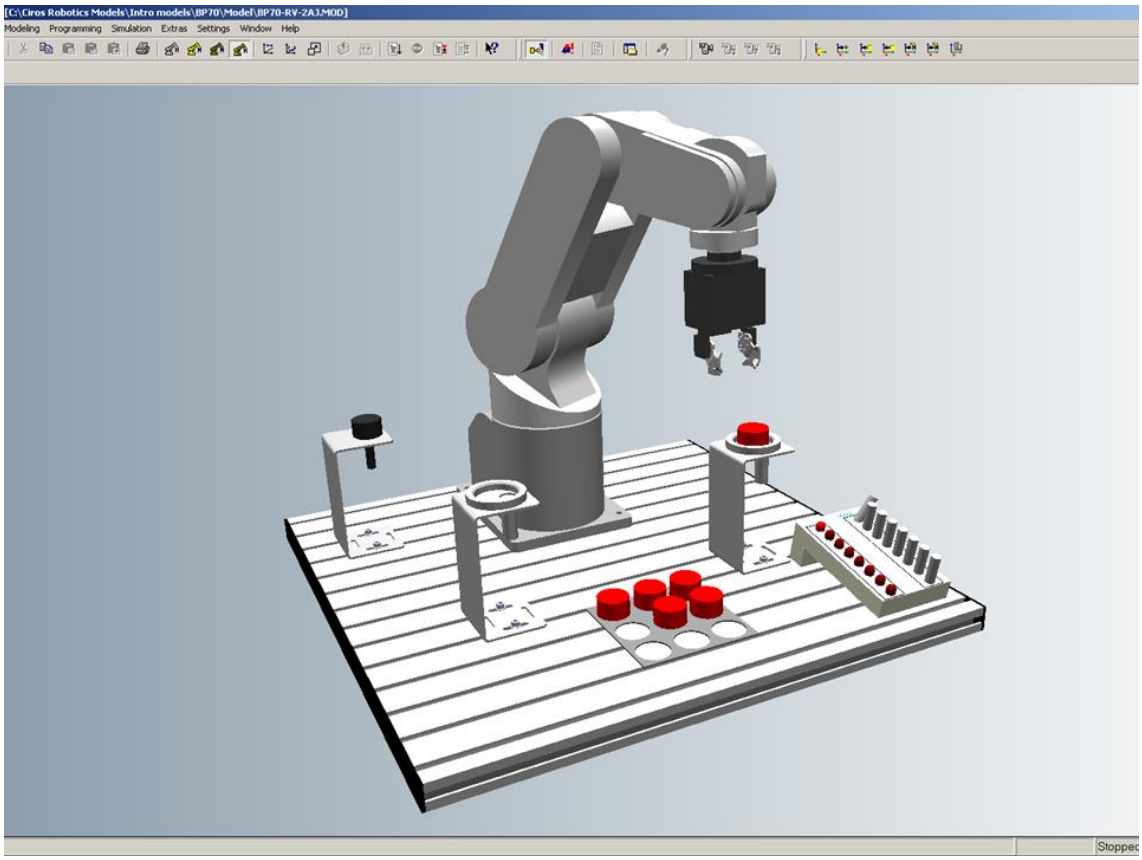
Tässä luvussa käydään läpi työssä käytetyt ohjelmat ja laitteet. Ciros Studio on käytössä konetekniikan osaston robotiikan kurssilla ja sillä ohjataan automaatiolaboratoriossa olevaa Mitsubishi Melfa RV-2AJ-robottia.

2.1 Ciros Studio

Ciros Studion on saksalaisen RIF e.V:n valmistama 3D-simulointiohjelmisto. Ohjelmisto sisältää mallinnuksen, ohjelmoinnin sekä simuloinnin, jotka sijaitsevat saman yhteisen käyttöliittymän alla. Lisäksi käytössä on laaja kirjasto erilaisia valmiita automatisoituja komponentteja ja mekanismeja sekä robotteja. Opetusta varten on myös käytössä valmiita työympäristöjä, joista yhtä käytettiin tässä työssä pohjana. Kuvassa 2.1 on Ciros Studion käyttöliittymä ja pohjana käytetty malli.

Ohjelmoinnissa voidaan käyttää kolmea eri ohjelmointikieltä: Melfa Basic IV, Mitsubishi MRL ja IRL. Työssä käytettyä RV-2AJ-robottia ohjelmoidaan Melfa Basicilla. Kieli sisältää paljon erilaisia robotin ohjaukseen liittyviä komentoja, mutta muuten sen käskykanta on suppea.

Simulaatiossa voidaan mekaanisten toimintojen lisäksi testata erilaisten tunnistimien, antureiden sekä kytkimien toimintaa. Myös törmäystunnistus sisältyy simulaatioon.



Kuva 2.1 Ciros Studion käyttöliittymä

2.2 Mitsubishi Melfa RV-2AJ

Mitsubishi Melfa RV-2AJ on pieni 5-akselinen teollisuusrobotti. Robotin maksimi liikkumisnopeus on 2,1 m/s, se pystyy 0,02 mm tarkkuuteen ja pystyy käsittelemään maksimissaan 2 kg painoisia kappaleita (Mitsubishi Industrial Robot RV-1A/RV-2AJ Series Standard Specification Manual). RV-2AJ:n ohjelmoinnissa käytetään Melfa Basic IV -ohjelmointikieltä.

2.3 Anturit

Anturit ovat tärkeitä komponentteja robotiikassa. Niiden avulla voidaan tunnistaa kappaleita ja niiden sijainteja. Lisäksi anturit ovat välttämättömiä laitteiden turvallisuuden kannalta.

Työssä käytettävät pelinappuloiden tunnistukseen käytettävät anturit ovat kapasitiivisia lähestymisantureita. Laitteiston turvallisuuden takaamiseksi siihen asennetaan optiset anturit eli valoverhot.

2.3.1 Kapasitiiviset anturit

Kapasitiivinen anturi on lähestymisanturi, joka muodostaa eteensä sähkökentän ja se aistii kentässä tapahtuvat muutokset, kun sen lähelle tuodaan kappale (Omron Industrial Automation. Proximity Sensors).

Kapasitiivisella anturilla on suuritaajuuksinen vaihtojännite ja muutos kapasitanssissa aiheuttaa uloslähtevän signaalin muutoksen. Kapasitiivinen anturi soveltuu ei-metallisten kappaleiden tunnistukseen.

2.3.2 Valokennot ja valoverhot

Valokennojen ja valoverhojen toiminta perustuu anturien havaitsemaan muutokseen lähetetyssä valonsäteessä (Omron Industrial Automation. Photoelectric Sensors). Antureita on kolmella eri periaatteella toimivia. Yleisimpiä on lähetin-vastaanotin ja heijastinanturi, joissa kappale katkaiseen lähettimeltä tulevan valonsäteen. Lähetin-vastaanotin-periaatteella toimivassa anturissa on erillinen valonlähde ja anturi. Heijastin-periaatteella toimivassa lähetin ja anturi ovat yhdessä ja toisella puolella on heijastava pinta. Kolmas anturityyppi on suora heijastus, jossa kappaleesta heijastuu valoa anturiin.

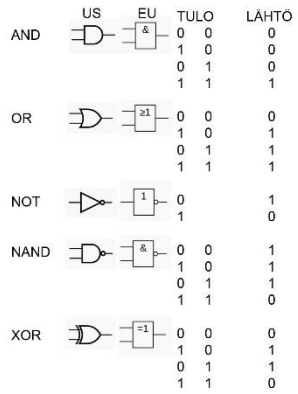
Valoverhossa on useita lähettimiä ja vastaanottimia, jolloin saadaan tiheä verkko valonsäteitä, jos joku valonsäteistä katkeaa antaa anturi signaalin.

Optisten antureiden toiminta voi perustua joko näkyvään valoon tai infrapunasäteilyyn. Hyvin usein käytetään infrapunavaloa, jotta ulkopuolinen valo ei häiritse antureiden toimintaa.

2.4 Loogiset Portit

Loogiset portit ovat elektronisia komponentteja, jotka valmistetaan yleensä transistoreista **(**Applied Robotics & Embedded Programming. How Logic Gates Work)**.** Loogisten porttien avulla voidaan tehdä laskutoimituksia digitaalisissa piireissä. Robotiikassa niitä käytetään muokkaamaan antureiden ja ohjelman laitteistolle antamaa dataa.

Loogisten porttien tulo ja lähtö ovat digitaalisia, 0 tai 1. Kuvassa 2.2 on esitetty erilaisia loogisia portteja tulo- ja lähtöarvoineen. Yhdistämällä erilaisia portteja voidaan muodostaa loogisia piirejä, joilla voi olla useita ulostuloja, ja niillä voidaan suorittaa monimutkaisempia laskutoimituksia.



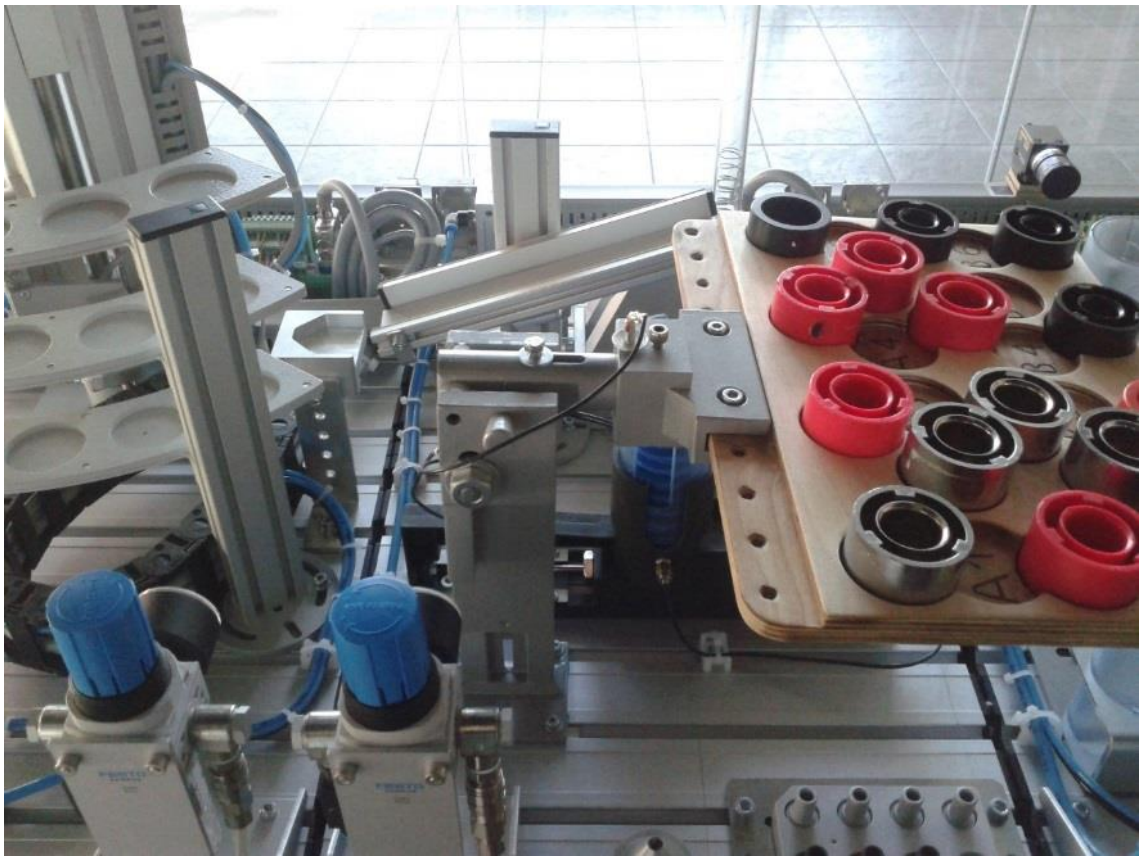
Kuva 2.2 Loogisia portteja

3 Toteutus

Peli on ristinolla 3 x 3 laudalla. Pelaajat asettavat nappulan vuorotellen laudalle ja voittaja on se, joka ensin saa 3 omaa nappulaansa joko vaaka-, pysty-, tai vinosuoralle. Peli päättyy tasan, mikäli laudalla ei enää ole vapaita paikkoja, eikä kumpikaan ole pystynyt muodostamaan kolmen suoraa.

Suunnittelu aloitettiin tekemällä tarvittavat mittaukset automaatiotekniikan laboratoriossa. Mitoituksessa käytettiin laboratoriossa olevaa telinettä sekä pyöreitä palikoita, joita voi käyttää pelinappuloina (kuva 3.1). Näiden lisäksi pelilaudan suunnittelussa huomioonotettavia asioita olivat robotin ulottuvuudet, sekä kouran dimensiot. 3D-mallinnusohjelmalla tehtiin kuva visiosta (kuva 3.2)

Virtuaaliympäristön toteutukseen pohjaksi otettiin Ciros Studiossa oleva valmis robotin työympäristömalli, joka on esitetty kuvassa 2.1. Pelilogiikan suunnittelussa päädyttiin siirtojen ennakoinnin sijasta yksinkertaiseen nappuloiden sijaintia läpikäyvään algoritmiin, ohjelman suorituksen hitauden takia.

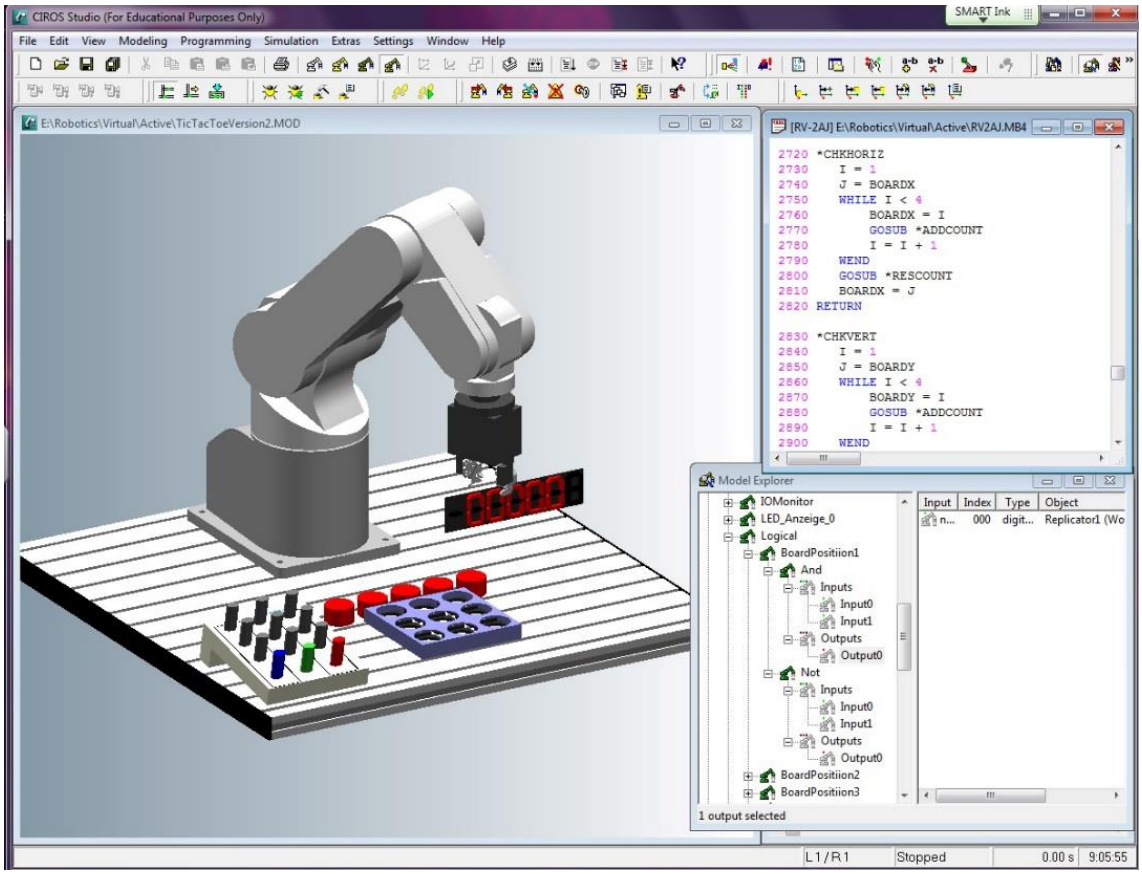


Kuva 3.1 Kiinnitysteline alustoille ja nappulat

Kuva 3.2 Alkuperäisen suunnitelman mukainen mallinnettu kuva

3.1 Virtuaaliympäristö

Alkuperäisestä virtuaaliympäristöstä poistettiin ensin tarpeettomat komponentit. Kytkinpaneeliin lisättiin 4 uutta kytkintä ja asetettiin kytkimet neljään riviin helpottamaan pelaamista. Kolmessa ylimmäisessä rivissä olevat 9 harmaata kytkintä ovat vastustajan siirtoja varten (kuva 3.3). Sininen kytkin on aloittajan valintaa, vihreä vaikeustasoa ja punainen pelin aloitusta varten. Pelialustaa muokattiin paksummaksi ja robotin nappuloiden sekä pelilaudan reikien paikat tallennettiin paikkalistaan.



Kuva 3.3 Virtuaaliympäristö

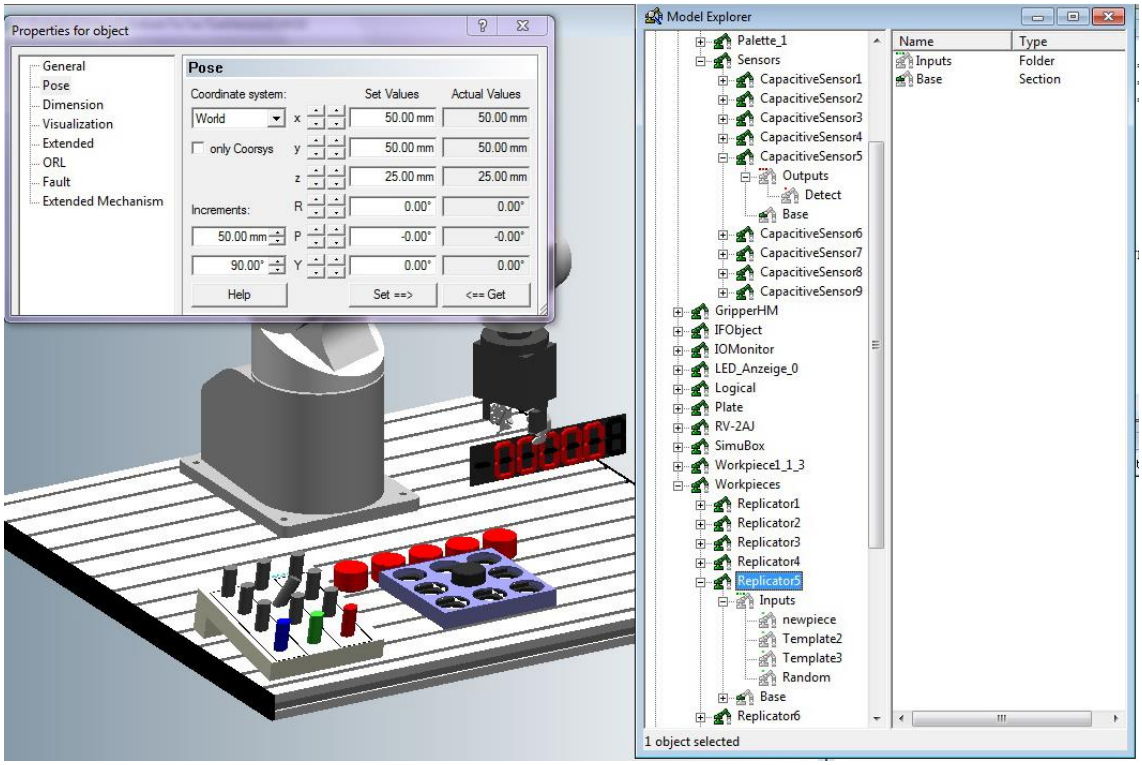
3.2 Anturit

Pelialustaan aseteettiin nappuloiden paikkojen kohdille kapasitiiviset anturit. Kapasitiivinen anturi soveltuu ei-metallisten kappaleiden tunnistukseen. Niiden toiminta perustuu tunnistusalueella olevaan magneettikenttään. Anturi reagoi väliaineen aiheuttamaan muutokseen sähkökentässä.

3.3 Replikaattori

Toinen nappuloiden kohdalle asetettu komponentti on replikaattori eli kopioijakomponentti. Saatuaan kytkimeltä signaalin, replikaattori tulostaa ennalta määritellyn komponentin, tässä tapauksessa pelinappulan, siihen pisteeseen, mihin replikaattori on sijoitettu. Kuvassa 3.4 on käännetty kytkintä ja replikaattori on tulostanut pelinappulan laudalle. Kuvassa näkyy myös komponentin ominaisuudet välilehti, jossa asetetaan replikaattorin paikka xyz-koordinaatistossa. Model

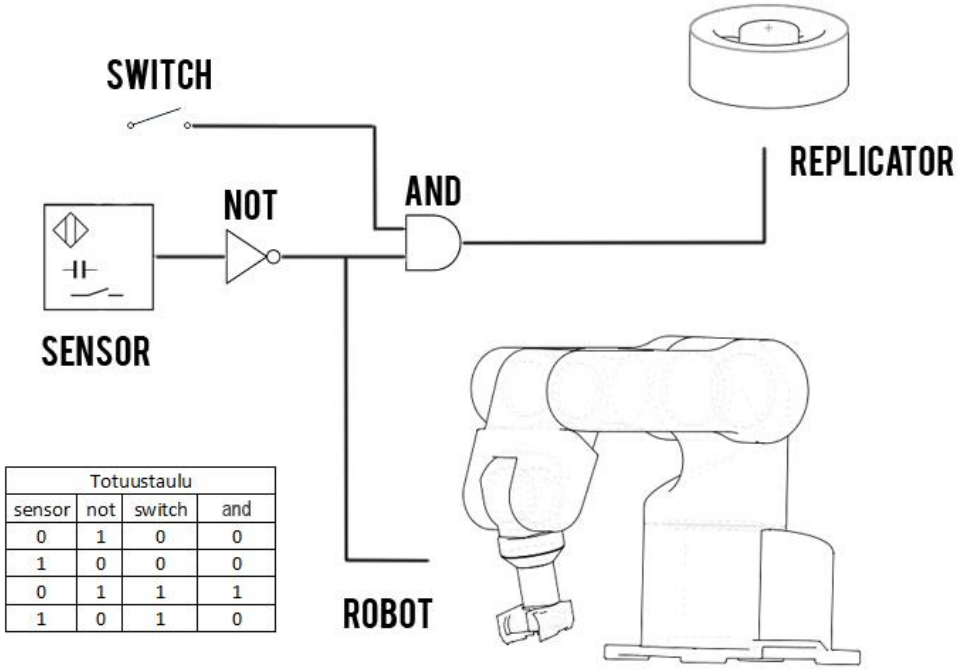
Explorer-ikkunassa näkyvät kaikki virtuaaliympäristön komponentit. Input- ja output-signaaleille eri komponenttien välillä voidaan tehdä yhteys helposti ”drag and drop”-menetelmällä.



Kuva 3.4 Vastustajan pelinappulan tulostus laudalle

3.4 Loogiset portit

Anturin ja kytkimen signaalit kulkevat loogisten porttien läpi. Anturi on kytketty loogiseen EI-porttiin. Tämän jälkeen anturin ja kytkimen signaalit kulkevat loogisen JA-portin läpi. Näin saatu signaali menee replikaattorille (kuva 3.5). Kuvassa olevasta totuustaulukosta nähdään, että replikaattori tulostaa nappulan, kun kytkintä on käännetty ja anturi ei tunnista paikalla nappulaa. Kuvan 3.3 Object Explorer-ikkunassa on sama kytkentä asetettu robotille.



Totuustaulu			
sensor	not	switch	and
0	1	0	0
1	0	0	0
0	1	1	1
1	0	1	0

Kuva 3.5 Signaalin kulku

3.5 Turvavaloverho

Oikealla robotilla työskenneltäessä tulee ottaa huomioon turvallisuus. Vaikka robotti on pieni ja varsin pienitehoinen, niin sormivammojen mahdollisuus on olemassa. Siksi virtuaaliympäristöön asetettiin valmiiksi turvavaloverhon input-signaalin tunnistus oikeaa robottia varten. Valonsäteen katkaiseminen aiheuttaa välittömästi robotin liikkeen pysähtymisen.

3.6 Led-näyttö

Virtuaaliympäristöön lisättiin led-näyttö, jota voi tarvittaessa käyttää testauksessa, esimerkiksi ohjelman suoritusvaiheiden tai siirtojen seuraamiseen.

3.7 Ohjelmointi

Ohjelmointi tehtiin käyttäen Melfa Basic IV-robottiohjemointikieltä.

3.7.1 Alkuasetukset

Virtuaaliympäristön mallinnuksen yhteydessä paikkalistaan asetetut koordinaattipisteet tallennetaan taulukoihin. Pelin vaikeustaso saadaan robotin input-signaalin bitistä 9 (0 = helppo, 1 = vaikea) ja pelin aloittaja bitistä 10 (0 = vastustaja, 1 = robotti). Bitti 11 = 0 kun peli on seis ja 1 kun peli käynnistyy. Kuvassa 3.6 on alustuskoodia.

Lisäksi mallissa olevan led-näytön numeroiden (1 – 9) seitsensegmenttikoodit tallennetaan taulukkoon.

```
360 DIM PDATA(9)        'Taulukko laudan pisteille
370 DIM PRDATA(5)       'Taulukko robotin nappuloiden pisteille
370 DIM LEDBIT(9)       'Taulukko led-näytölle

380    'Asetetaan taulukot
390 LEDBIT(1) = 6
400 LEDBIT(2) = 91
410 LEDBIT(3) = 79
420 LEDBIT(4) = 102
430 LEDBIT(5) = 109
440 LEDBIT(6) = 125
450 LEDBIT(7) = 7
460 LEDBIT(8) = 127
470 LEDBIT(9) = 111


490 PDATA(1) = P11
500 PDATA(2) = P12
510 PDATA(3) = P13
520 PDATA(4) = P14
530 PDATA(5) = P15
540 PDATA(6) = P16
550 PDATA(7) = P17
560 PDATA(8) = P18
570 PDATA(9) = P19
580 PRDATA(1) = P1
590 PRDATA(2) = P2
600 PRDATA(3) = P3
610 PRDATA(4) = P4
620 PRDATA(5) = P5

920 TURN = M_IN(10)         'Input signaalin bitti 10 (aloittaja)
930 LEVEL = M_IN(9)         'Input signaalin bitti 9 (vaikeustaso)
940 WAIT M_IN(11) = 1       'Odotetaan bittiä 11 (Peli alkaa)
```

Kuva 3.6 Alkuasetukset

3.7.2 Nappulan asettaminen

Pelilaudalle asetettua nappulaa vastaava input-signaalin bitti muuttuu 1:ksi. Vastustajan siirtoa odotetaan seuraamalla signaalin bittejä 0 - 8. Kun muutos tapahtuu, kutsutaan pelilogiikkaa. Logiikka palauttaa paikan, johon robotti siirtää

omasta viiden rivistään seuraavana vuorossa olevan nappulan. Kuvassa 3.7 on aliohjelmat vastustajan ja robotin siirroille. Kun robotti liikkuu, valoverhon katkaiseminen aiheuttaa keskeytyksen ohjelmaan ja robotti pysähtyy. Se jatkaa, kun este on poistunut.

```
320 DEF ACT 1, M_IN(12) = 1 GOSUB *WAITCLR  'Keskeytys. Valoverhon tarkkailija

940 'Odota pelaajan siirtoa
950 *WAITPLR
960     WAIT (M_INW(0) AND 511) <> BITCNT   'BITCNT muuttujassa on tallennettuna signaalin arvo
970     PLRMOV = M_INW(0) AND 511 - BITCNT  'Noussut bitti
980     BITCNT = M_INW(0) AND 511           'Tallennetaan signaalin arvo
990     GOSUB *PLRCOORD                     'Tallennetaan 3 x 3 matriisiin
1000    GOSUB *CHKWIN                       'Tarkistetaan voittiko
1010    TURN = 1                            'Vuoron vaihto
1020 RETURN

1030 'Robotti siirtää
1040 *PUTBTN
1050    ACT 1 = 1                           'Herätetään valoverhon tarkkailija
1060    MOV PRDATA(BTNMB), -90              'Siirrä seuraavan oman nappulan luo
1070    SPD 60                              'Nopeutta hiljaisemmaksi
1080    MVS PRDATA(BTNMB)                   'Siirrä alas
1090    HCLOSE 1                            'Nappaa nappula
1110    SPD 100
1120    MOV PRDATA(BTNMB), -90              'Nosta ylös
1130    MOV PDATA(ROBOMOV), -90             'Siirrä laudalle
1140    SPD 60
1150    MVS PDATA(ROBOMOV)
1160    SPD 100
1170    HOPEN 1                             'Pudota nappula
1180    MOV PDATA(ROBOMOV), -90
1190    MOV P100                            'Palaa odotuspisteeseen
1200    ACT 1 = 0                           'Valoverhon tarkkailija nukkumaan
1210    BTNMB = BTNMB + 1                   'Seuraava oman nappulan paikka
1220 RETURN

3580 *WAITCLR                               'Robotin liike pysähtyy kunnes este poistuu valoverhon edestä
3590    WAIT M_IN(12) = 0
3600 RETURN 1
```

Kuva 3.7 Aliohjelmat siirtoja varten

3.7.3 Pelilogiikka

Käytetyn ohjelmointikielen, Melfa Basic IV:n, käskykanta on melko suppea ja ohjelman suoritus hidasta, joten logiikan pitää olla mahdollisimman yksinkertainen. Helpoin tapa on käydä läpi vaaka-, pysty- ja vinorivien tyhjät paikat pelilaudalla, ja sijoittaa nappula parhaaseen paikkaan. Tämä haku tapahtuu vasta, kun pelilaudalla on riittävästi nappuloita

Jos robotti aloittaa:

1. Helppo taso:

1.1 Arvo siirto

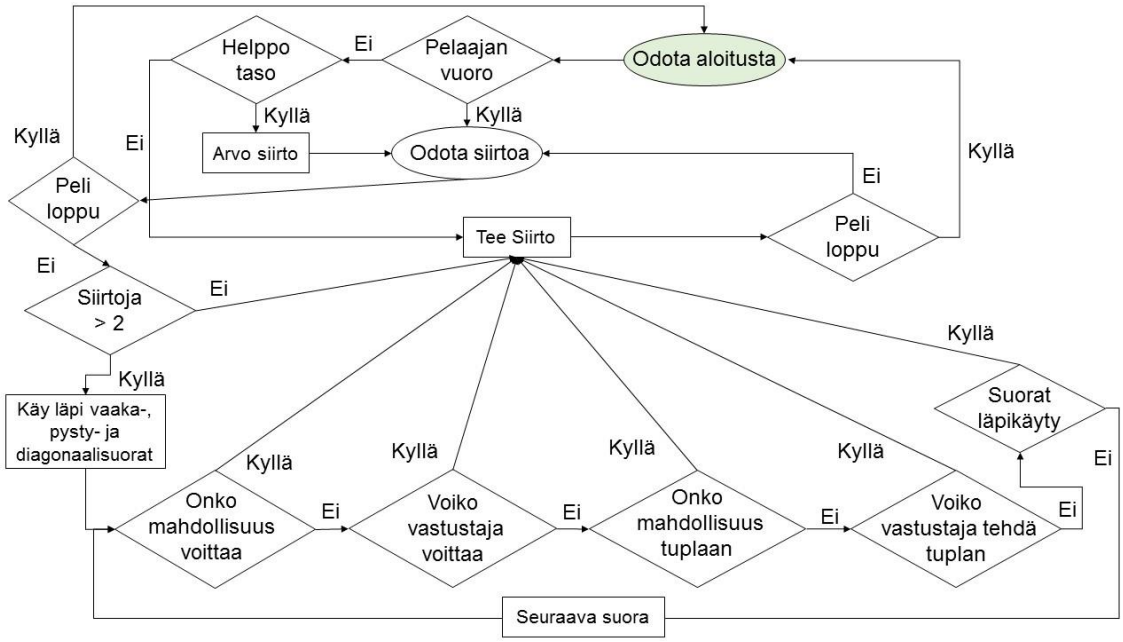
1.2 Vastustajan siirto

1.3 Laita samalle suoralle nappula, jolla arvottu nappula on
1.4 Vastustajan siirto
1.5 Tarkasta ja siirry kohtaan 1.4
2. Vaikea taso:
2.1 Laita keskelle
2.2 Vastustajan siirto
2.3 Laita aina vastakkaiseen kulmaan, laittoipa vastustaja kulmaan tai sivulle
2.4 Vastustajan siirto
2.5 Tarkasta ja siirry kohtaan 2.4
3. Tarkastus (tässä järjestyksessä tarkastetaan suorat ja tehdään siirto):
3.1 Onko mahdollisuus voittaa
3.2 Onko vastustajalla mahdollisuus voittaa
3.3 Onko mahdollista tehdä tuplapaikka (kahdelle riville lopetuspaikka)
3.4 Onko vastustajalla mahdollisuus tehdä tuplapaikka
3.5 Onko oman nappulan vieressä 2 tyhjää -> saadaan lopetuspaikka
3.6 Onko vastustajalla mahdollisuus tehdä lopetuspaikka
3.7 Jos mikään edellisistä ei toteudu laitetaan ensimmäiseen vapaaseen

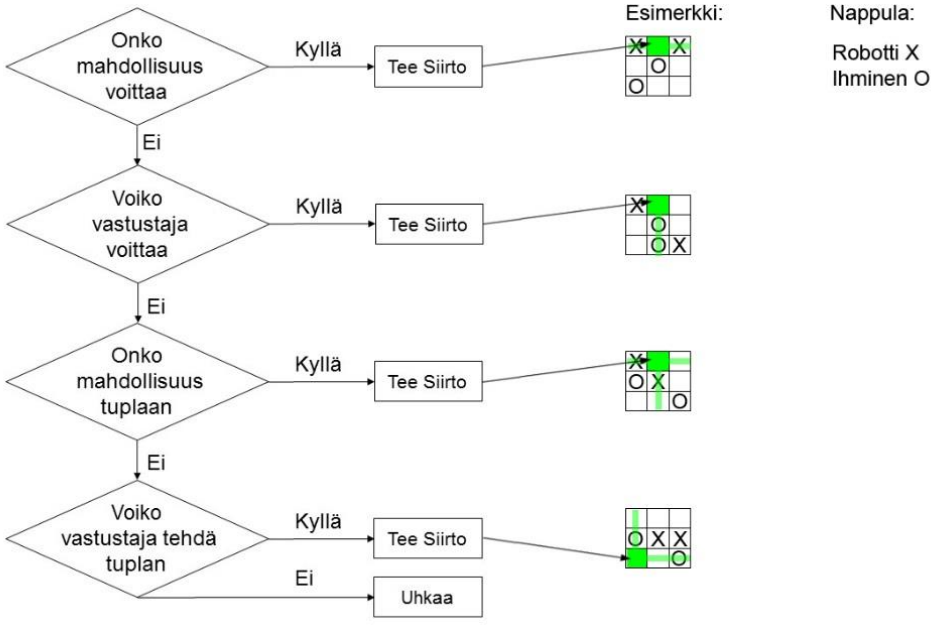
Jos pelaaja aloittaa:

1. Helppo taso:
1.1 Pelaajan siirto
1.2 Arvo siirto
1.3 Vastustajan siirto
1.4 Tarkastus kuten edellä
2. Vaikea taso:
2.1 Vastustajan siirto
2.2 Jos vastustaja laittaa keskelle, laita kulmaan, muutoin keskelle
2.3 Vastustajan siirto
2.4 Tarkastus kuten edellä

Kuvassa 3.8 on esitetty vuokaaviona ohjelman logiikkaosa ja kuvassa 3.9 on esitetty esimerkein logiikan osa, joka etsii lopetuspaikkoja.



Kuva 3.8 Pelilogiikka vuokaaviona



Kuva 3.9 Lopetuspaikkojen tarkastus

4 Testaus

4.1 Pelilogiikan testaus

Erilaisia mahdollisia pelikombinaatioita on tuhansia, joten niiden kaikkien testaaminen pelaamalla on käytännössä mahdotonta. Pelaamalla testattiin ainoastaan kaikki logiikan vaihtoehtoiset toiminnot, joilla se etsii vaaka-, pysty- ja vinoriveiltä parasta mahdollista paikkaa nappulalle. Ohjelman toiminnassa havaittiin yksi puute. Ohjelma ei tarkasta, onko kummallakaan pelaajalla mahdollisuutta voittoon vai onko väistämättä edessä tasapeli. Logiikan toimintaan sillä ei ole vaikutusta ja se päätettiin jättää jatkokehitysideaksi.

Logiikkaa testattiin myös tekemällä yksinkertain ohjelmasilmukka, joka kävi läpi erilaisia aloituksia ja lisäksi arpoi siirtoja. Tällä pyrittiin simuloimaan eritasoisten vastustajien pelaamista. Silmukka pelasi logiikkaa vastaan yhteensä tuhat peliä eri aloittaja- ja tasoasetuksilla. Taulukossa 4.1 on esitetty testipelien tulokset.

	Robotti aloittaa		Vastustaja aloittaa	
	Helppo taso	Vaikea taso	Helppo taso	Vaikea taso
Voitto	22%	78%	16%	62%
Tappio	58%	0%	64%	0%
Tasapeli	20%	22%	20%	38%

Taulukko 4.1 Robotin pelitulosten jakauma

Tasojen välinen ero on huomattava. Sama havaittiin myös pelaamalla robottia vastaan. Testauksessa kokeiltiin muuttaa pelilogiikkaa, jotta helppoa tasoa olisi saatu hieman vaikeammaksi tai vaikeaa tasoa hieman helpommaksi. Molemmille tasoille testattiin ensimmäisen siirron painotettua arvontaa. Vaikeammalla tasolla testattiin myös tuplalopetuspaikan tarkastuksen ja hyökkäyksen (tee itselle lopetuspaikka) poistamista ensimmäisellä tarkastuskerralla. Helppo taso muuttui liian vaikeaksi ollakseen helppo ja vaikea liian helpoksi.

Koska testauksessa ei havaittu varsinaisia vikoja logiikassa, päätettiin se jättää alkuperäiseen muotoonsa.

4.2 Virtuaaliympäristön testaus

Virtuaaliympäristön testaus saatiin suoritettua samalla logiikan testauksen kanssa. Kytkinten, antureiden sekä loogisten porttien toimintaa seurattiin led-näytöllä. Näytölle saatiin signaalin arvo(1 / 0) ja paikka, mistä signaali tulee.

5 Yhteenveto ja pohdinta

Tässä opinnäytetyössä tavoitteena oli tehdä robotille ristinollapeliin soveltuva virtuaaliympäristö ja ohjelmoida pelilogiikka. Alkuperäisiin tavoitteisiin kuului myös pelialustan suunnittelu, 3D-tulostus ja anturointi sekä robotin testaus automaatiotekniikan laboratoriossa. Tavoite osoittautui aivan liian kunnianhimoiseksi ja tehtävät oikean robotin kanssa jouduttiin jättämään kokonaan pois. Aikaa vievin ja haastavin osa työstä oli riittävän nopean ja hyvän logiikan kehittäminen virtuaalimallille.

Virtuaaliympäristön ja pelilogiikan osalta tavoitteet saavutettiin, vaikka testeissä robotti osoittautuikin mahdottomaksi voittaa vaikealla tasolla. Helpommalla tasolla taas robotin voittaminen oli usein erittäin helppoa. 3 x 3 laudalla pelattaessa kuitenkin ensimmäiset siirrot ratkaisevat pelin kulun ja pienikin muutos logiikkaan muuttaa liikaa pelin vaikeutta.

Oppimistavoitteina oli perehtyminen robotin ohjaukseen ja ohjelmointiin, ja työn aikana käytiinkin läpi myös muita robottiohjelmointikieliä. Lisäksi koneautomaation antureihin ehdittiin perehtyä, ennen kuin käytännön osuus jätettiin työstä pois.

Jatkokehityksessä oikean robotin kanssa riittäisi työtä toisen opinnäytetyön verran tai muutamalle opiskelijalle projektityötä. Virtuaaliympäristössä voisi kehittää muiden 3 x 3 laudalla pelattavien pelien logiikoita. Lisäksi logiikkaan voisi lisätä tarkastuksen, onko kummallakaan mahdollisuutta voittaa vai päättyykö peli tasan, jolloin peli loppuu.

Kuvat

Taulukot

Lähteet

Applied Robotics & Embedded Programming. How Logic Gates Work
http://www.rkessler.com/AdvRobotics-Embedded/Lessons/Lesson2/IC/2-IC-How%20Logic%20Gates%20Work.pdf

Mitsubishi Industrial Robot RV-1A/RV-2AJ Series Standard Specification Manual
http://www.rixan.com/Portals/0/RV-1A-2AJ/1n2specs.pdf

Omron Industrial Automation. Proximity Sensors
http://www.omron-ap.com.ph/service_support/technical_guide/proximity_sensor/further_information.asp

Omron Industrial Automation. Photoelectric Sensors
http://www.omron-ap.com.ph/service_support/technical_guide/photoelectric_sensor/index.asp