jamk.fi

Teollisuusrobotin simulointi Delmia-ohjelmistolla

Toni Ahonen

Maaliskuu 2015

Automaatiotekniikan koulutusohjelma
Tekniikan ja liikenteen ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
AHONEN, Toni	Opinnäytetyö	04.03.2015
	Sivumäärä 66	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi
Teollisuusrobotin simulointi Delmia-ohjelmistolla

Koulutusohjelma
Automaatiotekniikan koulutusuohjelma

Työn ohjaaja(t)
RANTAPUSKA, Seppo
STRÖM, Markku

Toimeksiantaja(t)
Vision Systems Oy
RAHKOLA, Kari

Tiivistelmä

Opinnäytetyön tavoite oli selvittää simulointimallin avulla asiakasyritykselle oliko heidän tuotantonsa eräs työvaihe robotisoitavissa. Jotta saataisiin tietää onko kyseisen työvaiheen robotisointi kannattavaa, oli ensin varmistettava simuloimalla saataisiinko asennettavat osat asennettua kappaleeseen halutussa kulmassa ja riittävän nopeasti. Tehtävään valitun robotin täytyi olla varustettu myös liikkeenseurannalla, koska kappale, johon osat tultaisiin asentamaan, oli liikkeessä osien asennuksen aikana.

Valmistettavuus selvitettiin luomalla Delmia-ohjelmistoon simulointimalli kyseisestä työvaiheesta. Simulointimallissa virtuaalisella teollisuusrobotilla simuloitiin työvaiheessa tarvittavat liikesarjat. Simulointimallin luotettavuuden varmistaminen nopeuden osalta tehtiin luomalla erillinen robottiohjelma, joka ajettiin sekä Delmialla, Roboguidella ja Jyväskylän ammattikorkeakoululla olleella Fanuc-nivelvarsirobotilla.

Opinnäytetyö toteutettiin pääasiassa Jyväskylän ammattikorkeakoulun tiloissa toimeksiantajan tilojen sijasta, koska opinnäytetyössä tarvittujen ohjelmien käyttö oli helpointa JAMK:n tiloissa verkkolisenssien vuoksi. Opinnäytetyö koostuu johdannon lisäksi robottien ja simulaation teoriasta, sekä projektin työn osuudesta. Opinnäytetyön tuloksena saatiin varmistus sille, että työvaiheen robotisointi on mahdollista. JAMK:n robotilla saavutettu suoritusnopeus ei täysin vastannut simulointiohjelmien simuloimaa nopeutta, mutta silläkin päästiin nopeuteen joka oli työvaiheeseen riittävä.

Avainsanat (asiasanat)

Delmia, Fanuc, Roboguide, simulointi, teollisuusrobotti

Muut tiedot

Tämä opinnäytetyö on osittain salattu

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Description

Author(s)	Type of publication	Date
AHONEN, TONI	Bachelor's thesis	04.03.2015
		Language of publication: Finnish
	Number of pages: 66	Permission for web publication: x

Title of publication
Simulation of industrial robot

Degree programme
Automation Engineering

Tutor(s)
RANTAPUSKA, Seppo
STRÖM, Markku

Assigned by
Vision Systems Oy
RAHKOLA, Kari

Abstract

The aim of this thesis was to determine for the client company, if it was feasible to robotize one stage of their production. Therefore, to confirm whether or not it was worthwhile to use an industrial robot in this task, a simulation model was built to ensure that the relevant parts could be installed at the desired angle and fast enough. Additionally, during installation the block where the parts were to be installed was in motion during the installation. Because of that the robot had to be also equipped with motion tracking.

The manufacturability was studied by creating a simulation model of the task at hand using Delmia software. In the simulation model, the virtual industrial robot was programmed to perform motion sequences needed in the stage in question. The simulation model's reliability was ensured by creating a separate robot program. That robot program was then run in Delmia, Fanuc's own robot simulation software Roboguide and a real industrial robot owned by JAMK University of Applied Sciences.

The thesis was mainly carried out in the JAMK premises, since it was easiest to use the important programs in the school premises due to the network licenses. The thesis consists of theories about robotics in general and the simulation and the part written on the project component. As a result, this thesis confirms that usage of an industrial robot in this work stage is possible. The speed reached by JAMK robot did not quite match the speed simulated by the simulation software, however, it reached the speed sufficient enough for this work stage.

Keywords/tags (subjects)

Delmia, Fanuc, industrial robot, Roboguide, simulation

Miscellaneous

This thesis is partially hidden

SISÄLTÖ

KUVIOT

KÄSITTEET

CAD	Tietokoneavusteinen suunnittelu, suunnittelija hyödyntää tietokonetta mallintamisessa jne.
Fps	Ruutua sekunnissa, päivitysnopeuden yksikkö.
Fysikaalinen avaruus	Tila, jossa kaikki kappaleet sijaitsevat. Tässä tapauksessa tarkoitetaan tilaa robotin ympärillä.
Servomoottori	Toimilaite, jossa on takaisinkytkentä akselin kulman mittaamiseksi ja pyörimiskertojen laskemiseksi.
Spline	Pehmeä kurvi kontrollipisteiden välillä.
Tagi	Merkki, tässä tapauksessa piste, jolla on suunta fysikaalisessa avaruudessa.

1 Johdanto

1.1 Opinnäytetyön lähtökohdat

Opinnäytetyöni aiheena oli selvittää asiakasyritykselle, voitaisiinko eräs vaihe heidän tuotannossaan suorittaa teollisuusrobotilla. Työn toimeksiantajana oli Vision Systems Oy, joka suunnitteli ko. asiakasyritykselleen 3D-konenäkö kamerajärjestelmää kyseiseen työvaiheeseen. 3D-konenäkö kamerajärjestelmän avulla olisi mahdollista myös luoda 3D-kartta paikoista, joihin osia halutaan tuotteeseen tässä työvaiheessa asentaa. Tätä 3D-karttaa hyödyntämällä haluttaisiin prosessia automatisoida siten, että työvaiheen suorittavalle laitteelle voitaisiin kartan perusteella luoda suoritettava ohjelma. Tällä hetkellä työvaiheen suorittaa manipulaattori-laitteisto, mutta asiakasyrityksellä haluttiin selvittää voitaisiinko työvaihe suorittaa teollisuusrobotilla. Asian selvittämiseksi tehtiin simulointi, eli valmistettavuuden arviointi kyseisestä työvaiheesta teollisuusrobotilla. Projektista tehtiin salassapito sopimus, joten suuri osa tämän opinnäytetyön työn sisällöstä on salattu.

Tämä opinnäytetyö on tehty syksyn 2014 ja alkutalven 2015 aikana. Projektin pääsin aloittamaan jo kesällä 2014 työharjoittelun merkeissä, jossa esim. tutustuin työssä käytettäviin ohjelmiin, tein alustavia kokeiluja teollisuusrobotin simuloimisesta sekä valmistelin varsinaista työtä. Työskentelin pääsääntöisesti Jyväskylän ammattikorkeakoulun tiloissa, koska työ suoritettiin ammattikorkeakoululla olleilla Delmia-ohjelmistolla sekä Fanuc M16iB/20-nivelvarsirobotilla. Simuloinnin tuloksista palautin raportin asiakasyritykselle alkusyksystä 2014, mutta tässä vaiheessa projektiin jäi vielä avoimia kysymyksiä, joita selvitettiin tarkemmin myöhemmin.

Aluksi työssä esitellään Vision Systems Oy:n sekä asiakasyrityksen toimintaa. Sen jälkeen teoriaosuudessa käydään läpi mielestäni oleellisimmat asiat tämän työn osalta teollisuusroboteista. Teollisuusrobottien lisäksi teoriaosuudessa käsitellään myös simuloinnin teoriaa ja simuloinnissa käytettyjä ohjelmia. Teoriaosuuden jälkeen kerrotaan projektin työn osuudesta.

1.2 Nykyinen laitteisto

KUVIO 1 SALATTU

KUVIO 2 SALATTU

1.3 Vision Systems Oy

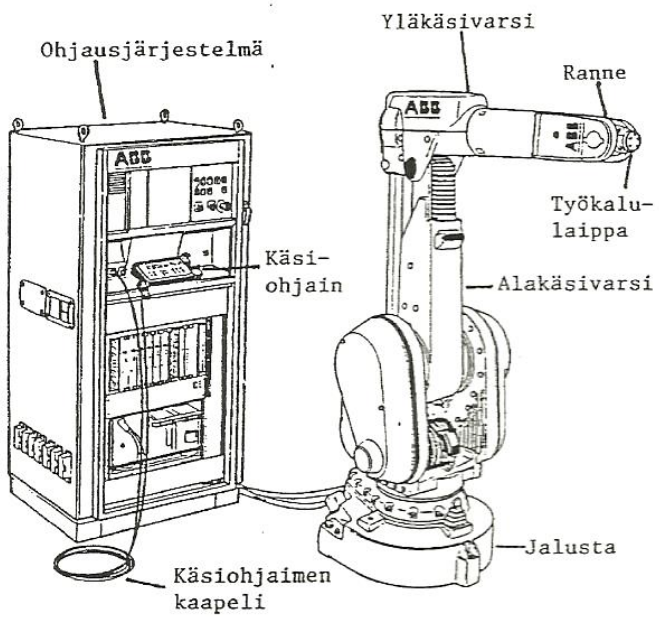
Vision Systems on vuonna 1985 perustettu korkean teknologian yhtiö, joka on erikoistunut konenäköön ja optiseen mittaustekniikkaan. Yhtiö on tunnettu optisista antureista, VISI-älykameroista ja monista teollisuuteen vaativiin olosuhteisiin kehitetyistä mittausjärjestelmistä. Merkittävä osa toiminnasta suuntautuu kansainvälisille markkinoille joko suoraan tai alansa johtavien laite- ja konevalmistajien kautta. Keväällä 2013 Vision Systems käynnisti uuden liiketoiminta-alueen, sulautettujen ohjelmistosuunnittelupalvelujen tuottamisen teollisille laite- ja konevalmistajille. Tämä suunnittelupalvelu-liiketoiminta yhtiöitettiin kesällä 2014 ja toiminta on kasvanut nopeasti työllistäen nyt jo 14 henkilöä. Koko konsernin henkilöstö on nyt 25 henkilöä. (Rahkola 2015.)

Yhtiö näkee robotiikan yhdessä konenäön kanssa olevan mielenkiintoinen kasvupotentiaali tuleville vuosille (Rahkola 2015).

1.4 SALATTU

2 Robotit

Robotti-sana on peräisin tšekkoslovakialaisen kirjailija Karel Čapekin näytelmästä ’Rossum’s Universal Robots’, jonka ensiesitys oli vuonna 1921. Sanan keksi hänen veljensä taitelija Josef Čapek, ja sillä kuvattiin keinotekoisia työntekijöitä. Nykyisin monenlaisia, ihmisiä auttamaan kehitettyjä koneita, voidaan kutsua roboteiksi. Teollisuusrobotin määritelmä (ISO 8373) on ”uudelleen ohjelmoitavissa oleva monipuolinen, vähintään kolmenivelinen mekaaninen laite, joka on suunniteltu liikuttamaan kappaleita, osia, työkaluja tai erikoislaitteita ohjelmoitavien tehtävien suorittamiseksi teollisuuden sovelluksissa”. (Lehtinen n.d., 1-2.)



KUVIO 3 Tavanomaisen nivelvarsirobotin komponentit (Kuivanen 1999, 13.)

2.1 Robotin määritelmä

Robotit luokitellaan kuuteen luokkaan JIRA:n (Japanilainen teollisuusrobotti järjestö) mukaan seuraavasti:

- Luokka 1: Operaattorin käyttämä manipulaattori, jossa on useita ohjattavia vapausasteita.
- Luokka 2: Kiinteän sekvenssin robotti, laite joka toistaa ennakkoon määriteltyjä tehtäviä, ja on vaikea muokata.

- Luokka 3: Muunneltavan sekvenssin robotti, samanlainen kuin luokka 2, mutta helppo muokata.
- Luokka 4: Johdattamalla ohjelmoitava robotti, joka toistaa operaattorin sille johdattamalla opetetut liikkeet.
- Luokka 5: Numeerisesti ohjattu robotti, operaattorin ohjelmoitavissa.
- Luokka 6: Älykäs robotti, tarkkailee ympäristöään ja voi suorittaa tehtävänsä vaikka ulkopuoliset olosuhteet muuttuisivatkin.

(Niku 2001, 2.)

Yhdysvalloissa ja Suomessa yllä olevista luokista kahta ensimmäistä ei luokitella roboteiksi, toisin kuin siis vaikkapa Japanissa, mikä osaltaan selittää Japanin suurta robottimäärää tilastoissa. Voidaan siis yleisesti sanoa, että helppo uudelleenohjelmoitavuus on tärkeä määrittely robotille. (Robotiikka 2008, 9.)

Robotiikan esihistoria

Varhaisimpia robottien edeltäjiä voisivat olla vaikkapa ranskalaisen Jacques de Vaucanson'n rakentamat paineilmalla toimiva huilunsoittaja, sekä ankka joka koostui yli neljästäsadasta liikkuvasta osasta. Vaucanson'n ankka osasi leyhytellä siipiään ja simuloida jopa ruoansulatusta. Näiden mekaanisten laitteiden lisäksi Vaucanson suunnitteli myös kutomisprosessin automatisoinnin reikäkortein operoitavin kutomokonein. Kutomisen automatisointia ei kuitenkaan koskaan pantu täytäntöön kutomotyöläisten vastarinnan takia, ja valitettavasti myöskään luotettavia dokumentteja laitteiden tekniikasta ei ole säilynyt. Häntä ennen ja hänen jälkeensä on myös valmistettu useita laitteita, joita voidaan yhtä hyvin kutsua robotiikan esihistoriaksi. Robotiikan esihistorian voidaan katsoa päättyneen vuoteen 1898, jolloin Nikolai Tesla esitteli pienikokoisen radio-ohjattavan veneen. Tällöin oli käytössä jo monia robotiikkaa enteileviä sovelluksia. (Huttunen 2004, 3-4.)

Ensimmäiset teollisuusrobotit

Toisen maailmansodan jälkeen kehitettiin numeerisesti ohjattuja työstökoneita, joilla kappaleita voitiin valmistaa entistä tarkemmin. Tärkeä keksintö robotiikalle tuli matematiikasta, kun Jacques Denavit ja Richard Hartenberg määrittelivät tavan esittää kinemaattisia ketjuja homogeenisten siirtomatriisien avulla. Kinemaattisen ketjun avulla voitiin esimerkiksi mallintaa robottikäsivarsi. Tämän lisäksi transistorin keksiminen mahdollisti tietokoneiden hyödyntämisen robotiikan apuna. Niinpä vuonna 1961 George Devol ja Joe Engelberger esittelivät ensimmäisen teollisuuskäyttöön tarkoitetun robotin, Unimaten. Ensimmäinen Unimate robotti asennettiin General Motorsin autotehtaalle, jossa sen tehtävä oli siirtää valumuoteista tulevat kuumat autonosat jäähdytysnesteeseen, ja jäähdytyksen jälkeen edelleen kuljettimelle. Käyttövoimansa neljälle ohjelmoitavalle akselilleen Unimate sai sähkömoottoreista ja sitä ohjasi alkeellinen tietokone, jonka rumpumuistiin mahtui 180 toiminta-askelta. (Huttunen 2004, 5.)

2.2 Robottivalmistajat

Erilaisia teollisuusrobotteja on suunniteltu useita tuhansia ja niitä on valmistanut useita satoja eri yrityksiä. Ajan myötä markkinat ovat keskittyneet ja valmistajien määrä on supistunut. Tunnettuja robottivalmistajia ovat esimerkiksi ABB, Fanuc, KUKA ja Motoman. (Kuivanen 1999, 12.)

Suomessakin on valmistettu ja valmistetaan teollisuusrobotteja. Nokia Oyj valmisti lisenssillä Unimation Puma 560-robottia Neuvostoliiton markkinoille 1980-luvulla ja olipa Nokialla omakin NS-16 käsivarsirobotti. Nykyisin Suomessa toimivia robottivalmistajia ovat Cimcorp Oy, joka valmistaa portaalirobotteja sekä Blastman, jonka robotit ovat erikoistuneet suurten teräskappaleiden hiekkapuhallukseen. Jotkut Suomessa tehdyt automaattinosturit täyttävät myös robotin määritelmän. (Teollisuusrobotti 2014.)

2.3 Robottien käyttö

Teollisuusrobotteja on käytetty ja kokeiltu useissa eri teollisuuden työtehtävissä. Työtehtäviä, joissa robotteja yleisesti käytetään, ovat esimerkiksi tuotteiden kokoonpano, hitsaus, työstökoneiden palveleminen ja maalaus. (Niku 2001, 20.)

Suurin syy teollisuusrobottien määrän kasvulle on niiden hintojen alentuminen samaan aikaan kun ihmistyövoiman hinta on lisääntynyt. Sen lisäksi että teollisuusrobottien hinnat ovat laskeneet, ne ovat tulleet tehokkaammiksi: nopeammiksi, tarkemmiksi ja joustavimmaksi. Tämän myötä yhä useampiin teollisuuden työtehtäviin on tullut kannattavaksi hankkia robotti korvaamaan ihminen. (Craig 2005, 1)

Suomessa uusia teollisuusrobotteja otetaan käyttöön vuosittain keskimäärin 300–400 kappaletta. Vaihtelu vuosien välillä on ollut kuitenkin suurta. Huippuvuonna 2005 teollisuusrobotteja otettiin käyttöön jopa 556, kun taas vastaavasti vuoden 2008 talouskriisin jälkeen vuosina 2009–2011 määrä romahti alle kolmensadan. (Suomen teollisuusrobottitilasto 2012.)

Robottien hyödyt ja haitat

Robotit pystyvät korvaamaan ihmisen esimerkiksi useissa ihmiselle vaarallisissa tai tylsissä työtehtävissä. Ne pystyvät työskentelemään ihmiselle vaarallisissa työympäristöissä, eivätkä tarvitse mukavuuksia. Robotit eivät väsy tai kyllästy tehtäviinsä, eivätkä ota sairaslomapäiviä. Myös jos työtehtävä vaatii suurta nopeutta tai tarkkuutta, robotti usein pystyy suoriutumaan siitä ihmistä paremmin. (Niku 2001, 5.)

Vastaavasti robotin toimintakuntoon saattaminen vaatii paljon työtä. Jotkut meille ihmisille hyvin yksinkertaiset asiat voivat ollakin teollisuusrobotille hyvinkin hankalia suorittaa. Lisäksi robotti ei osaa reagoida yllättäviin tilanteisiin, joita siihen ei ole ohjelmoitu. (Niku 2001, 6.)

2.4 Robotin mekaniikka

Yksinkertaistettuna teollisuusrobottien mekaniikka koostuu tukivarsista ja tukivarsia liikuttavista toimilaitteista (esimerkiksi sähköservomoottori). Kahden tukivarren välille muodostuu nivel, jota toimilaite ohjaa. Robottien nivelet ovat useimmiten joko suorakulmaisia, jolloin tukivarret liikkuvat toistensa suhteen tietyn suoran suunnassa tai kiertyviä, jolloin tukivarret kiertyvät suoran ympäri. (Kuivanen 1999, 15.)

Yleisin robottien kinemaattinen rakenne on avoin, jossa tukivarret on kytketty peräkkäin. Tällöin nivelten lukumäärä on sama kuin robotin vapausasteiden (dof) lukumäärä. Avoimen kinemaattisen rakenteen robottien ulottuvuus on suuri, mutta haittana kuormankantokyky on pieni. Suljetun kinemaattisen rakenteen roboteissa tukivarsia kytketään rinnakkain. Tällöin robotti on rakenteeltaan tukevampi ja sillä päästään suurempiin nopeuksiin, mutta työalue rajoittuu tällaisella robotilla pieneksi. (Kuivanen 1999, 16-17.)

2.4.1 Toimilaitteet

Yleisimmin roboteissa käytetään toimilaitteina sähköservomoottoreita. Suuria kuormia varten on olemassa myös hydraulisilla servoilla varustettuja robotteja - tosin nykyään uusissa roboteissa käytetään hydrauliikan sijaan tehokkaita sähkömoottoreita. Myös pneumaattisia toimilaitteita voidaan käyttää roboteissa, esim. räjähdysvaarallisissa tiloissa. Pneumaattisten servotoimilaitteiden ohjauksen kehittyessä myös niihin on kohdistunut hieman kaupallista mielenkiintoa. (Kuivanen 1999, 19.)

Roboteissa toimilaitteena voidaan käyttää monenlaisia sähkömoottoreita, kuten tasavirta-, vaihtovirta- ja askelmoottoreita. Askelmoottoreita lukuun ottamatta roboteissa käytettävät sähkömoottorit on varustettu takaisinkytkennällä eli ne ovat sähköservomoottoreita. (Niku 2001, 186.)

2.4.2 Voimansiirto

Toimilaitteen akselin liikkeen ja voiman siirtämiseen robotin nivelille tarvitaan voimansiirto. Robottien voimansiirrossa voidaan käyttää esimerkiksi työntötankoja, kuularuuveja ja hammaspyörästöjä. Nykyään lähes kaikkien robottien kaikki akselit on

varustettu jarruilla, jottei robotti pääse romahtamaan esim. sähkökatkon aikana. Robottien jarrut toimivat turvallisuussyistä lepovirta periaatteella eli jarrut ovat päällä kun niille ei syötetä virtaa. (Kuivanen 1999, 19-20.)

2.4.3 Anturointi

Koska robotin tulee kyetä liikuttamaan työkalua halutulla tavalla, tarvitsee liikkeiden ohjauksessa tietää robotin asema peruskoordinaatistossa. Teollisuusroboteilla työkalun asema määritetään epäsuorasti eli kun antureiden avulla tiedetään robottien nivelten kiertymiskulmat, niiden ja tiedossa olevista tukivarsien kinemaattisista pituuksista pystytään laskemaan työkalun paikka ja asento työalueella. (Kuivanen 1999, 21.)

Jotta pystytään tietämään toimilaitteiden asento, niiden kiertymiskulmat tarvitsee mitata asema-anturilla. Koska robotin servomoottorit ovat yhdistetty alennusvaihteella niveleen, tulee nivelen kiertymäkulman mittauksesta tarkka. Anturi on yleensä sijoitettu tilansäästösyistä suoraan servomoottorin akselille ja anturin kääntökulman lisäksi on tiedettävä kuinka monta kierrosta moottorin akseli on pyörinyt. (Kuivanen 1999, 30.)

Jokaisella robotin vapausasteella on oma anturinsa, josta asema luetaan tuhansia kertoja sekunnissa ohjausjärjestelmään. Paikka-tiedoista saadaan kiertymiskulman lisäksi laskettua myös tieto akselin liikesuunnasta, nopeudesta ja kiihtyvyydestä, joita kaikkia tarvitaan nivelen paikkasäädössä. (Kuivanen 1999, 30.)

2.4.4 Inkrementtianturit

Inkrementti- eli pulssianturissa valoa lähettävän LEDin ja valoa vastaanottavan diodin välissä on optinen pulssikiekko. Pulssikiekko on kiinnitetty akseliin, jonka paikkaa halutaan mitata. Pulssikiekossa on vuorotellen läpinäkyviä ja läpinäkymättömiä viivoja ja näiden viivojen lukumäärästä tulee anturin resoluutio. Akselin kiertymäkulmaa mittaavassa anturissa akselin (ja näin myös kiekon) pyöriessä LEDin lähettämä valo vuorotellen näkyy ja on näkymättä vastaanottavalle diodille. Tästä saatu analoginen

signaali muutetaan suorakulmaiseksi kanttiaalloksi anturin ulostuloon. Jotta myös pyörimissuunta saadaan selville, on kiekolla kaksi 90 asteen vaihe-erossa toisiinsa nähden olevaan viivakehää. Vaihe-eron takia anturin resoluutio on itse asiassa nelinkertainen yhden kanavaan pulssimäärän verrattuna, koska voidaan ottaa huomioon molempien kanavien nousevat ja laskevat reunat. (Pulssianturien teoriaa n.d.)

Inkrementtianturilta saadaan ohjausjärjestelmään siis tietona akselin kulkema matka pulsseina sekä liikesuunta. Yksi pulssi vastaa tiettyä akselin kulkemaa matkaa tai kiertyvissä nivelissä akselin kiertynyttä kulmaa. Inkrementtianturia käytettävät robotit on kalibroitava käynnistysvaiheessa (ajettava kotiasemaan), koska kaikki pulssit ovat samanlaisia, ja paikkatieto lasketaan siis pulssien määrästä. (Kuivanen 1999, 31.)

2.4.5 Absoluuttianturit

Absoluutti- eli koodianturissa pulssien sijaan kiekolla on usealle sisäkkäiselle kehällä värjätty binäärikoodi, joka ilmaisee akselin asennon. Absoluuttianturi tietää siis jatkuvasti todellisen sijaintinsa, eikä robotin tarvitse tehdä synkronointia käynnistyessään. Esimerkiksi 12-bittisellä koodianturilla saadaan 4096 eri asentoa (2 ^ 12 = 4096). (Kuivanen 1999, 32.)

2.5 Ohjausjärjestelmä

Ohjausjärjestelmä on robotin aivot, joka ohjaa robotin servotoimilaitteita ja kommunikoi ympäröivän maailman kanssa. Ohjausjärjestelmä koostuu tavanomaisesti keskusyksiköstä (prosessitietokone joka sisältää useita suorittimia), robottiohjelmien tallennukseen käytetystä massamuistista, käsiohjaimesta (robotin operointi ja ohjelmointi), liitännöistä ulkoisille tietokoneille, robotin liikkeet suorittavista nivelkohtaisista servotoimilaitteista ja teholähteistä, jotka vastaavat laitteiston sähkönsyötöstä. (Kuivanen 1999, 34.)

Esimerkiksi JAMK:n Fanuc M16iB/20-robotin ohjausjärjestelmän nimi on R-J3. R-J3 koostuu sen ohjeen mukaan virtayksiköstä, käyttöliittymäpiiristä, liikkeenohjauspiiristä, muistipiiristä ja syöttö-/ulostulopiiristä. Ohjausyksikön lisäksi ohjausjärjestelmään kuuluvat käsiohjainlaite ja käyttöpaneeli. (FANUC robotti R-J3 Ohjaus n.d., 44.)

2.5.1 Kinematiikka

Robotti laskee kinematiikkaa käyttäen työkalunsa sijainnin fysikaalisessa avaruudessa. Kinematiikassa robotti laskee kinemaattisten yhtälöiden ja matriisilaskennan avulla joko työkalun paikan fysikaalisessa avaruudessa tai robotin nivelten kulmat. Suoran kinematiikan avulla lasketaan missä robotin työkalun sijaitsee, kun tiedetään robotin nivelkulmat. Käänteinen kinematiikka laskee robotille tarvittavat nivelkulmat sen perusteella missä paikassa ja asennossa työkalun halutaan olevan. (Niku 2001, 29.)

2.5.2 Suora kinematiikka

Kun kaikkien tukivarsien pituudet ja niiden välissä olevien nivelten kulmat sekä robotin työkalulaippaan kiinnitetyn työkalun työkalupisteen etäisyys laippaan tiedetään, voidaan laskea paikka ja orientaatio missä robotin työkalu sijaitsee fysikaalisessa avaruudessa. (Niku 2001, 53.)

2.5.3 Käänteinen kinematiikka

Kun robotin työkalu halutaan siirtää tiettyyn paikkaan ja orientaatioon fysikaalisessa avaruudessa, tarvitsee robotin laskea nivelkulmat, joilla tuo koordinaatti saavutetaan. Käänteisessä kinematiikassa robotin tukivarsien pituudet ja robotin haluttu asema on tiedossa. Käänteinen kinematiikka laskee näiden tietojen perusteella missä asennossa nivelten kuuluu olla, jotta tuo asema saavutetaan. Käänteisen kinematiikan avulla robottiohjain siis laskee robotin nivelten arvot yhtälöiden avulla. (Niku 2001, 53.)

2.5.4 Denavit-Hartenbergin yhtälöt

Denavit ja Hartenberg julkaisivat vuonna 1955 kaavat, joilla pystytään mallintamaan robotti ja selvittämään niiden liikeyhtälöt. Heidän tekniikastaan on tullut standardi esittämään ja mallintamaan robottien liikkeitä. Denavit-Hartenbergin kaavoja voidaan käyttää mille tahansa robotti tyypille, kuten esimerkiksi Scara- tai nivelvarsiroboteille. (Niku 2001, 67.)

2.5.5 Robotin dynamiikka

Massa tarvitsee kiihtyäkseen voiman. Koska robotti itse sekä sen käyttämät työkalut ja siirtämät työkappaleet painavat, täytyy robotin toimilaitteiden olla riittävän voimakkaita ja vääntäviä, jotta ne jaksavat käsitellä tämän kuorman. Jos voimaa tai vääntöä ei ole riittävästi, robotin liikenopeus ja tarkkuus kärsivät. Yhtälöt, joiden avulla voidaan laskea kuinka voimakkaita robotin toimilaitteiden tulisi olla, ovat F = m/A, jossa F = voima (N), m = massa (kg) ja A = kiihtyvyys (m/s^2) sekä M = J * α, jossa M = vääntömomentti (Nm), J = hitausmomentti (kgm^2) ja α = kulmakiihtyvyys (rad/s^2). Newtonin mekaniikan lisäksi robotin liikeohjaimen dynaamisissa liikeyhtälöissä voidaan käyttää energiaan perustavaa Langrangianin mekaniikkaa. Langrangianin mekaniikka on teollisuusrobotin tapaisissa monimutkaisissa tapauksissa huomattavasti yksinkertaisempaa laskea kuin Newtonin mekaniikka. (Niku 2001, 119-120.)

Kun robotin kinematiikan lisäksi on tiedossa robotin dynamiikka voi robottiohjain suunnitella liikeradan, jotta robotti saa työkalun siirtymään haluttuun paikkaan, halutulla nopeudella ja haluttua reittiä pitkin. Robottiohjain siis ratkaisee ensin käänteisen kinemaattisen yhtälön (eli laskee nivelten kulmat), jonka jälkeen se laskee kuinka paljon moottoreille tarvitsee syöttää virtaa, jotta vääntöä saadaan sen verran, että työkalu saadaan liikkumaan haluttuun asemaan. (Niku 2001, 147.)

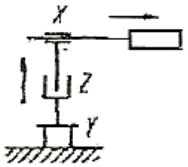
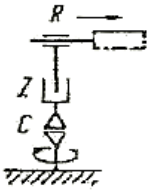
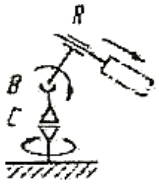
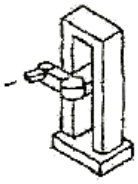
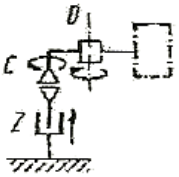
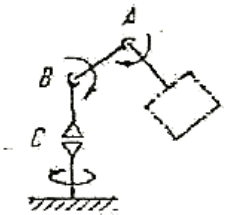
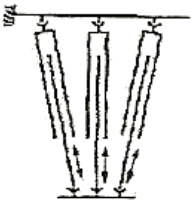
2.5.6 Tiedonsiirto

Robotille voidaan siirtää ohjelmia esimerkiksi levykkeellä tai muistikortilla. Nykyaikainen, joustava robottijärjestelmä vaatii kuitenkin tiedonsiirtoliitynnän, jotta sille voidaan ohjelmien lisäksi myös antaa käskyjä ja tuotantoa ohjaavia parametreja ylemmän tason tuotannonohjausjärjestelmästä. Perinteinen tapa robotin tiedonsiirtoon on ollut sarjaliikenneportit RS-232C ja RS-422. 90-luvun loppupuolella robotteihin tuli ainakin tilattavaksi lisäoptioksi ethernet-liityntä, jolla robotti voitiin yhdistää tehtaan lähiverkkoon. Ennen ethernet-liitynnän yleistymistä tiedonsiirrossa käytettiin paljon kenttäväyliä (mm. profibus) sarjaliikenneporttien lisäksi. (Kuivanen 1999, 48-49.)

Nykyään teollisuusrobotit ovat liitettävissä myös internetiin, jolloin niiden asetuksia ja tietoja päästään katsomaan ja muuttamaan vaikka toiselta puolelta maapalloa. Nykyaikainen robottiohjain voidaan liittää jo vakiona useaan eri tehdasväylään, ja lisäoptioina robottiohjaimiin on usein saatavilla eri liitäntäprotokollia. (Malm 2008, 36.)

2.6 Robottityypit

Teollisuusrobotit jaotellaan luokkiin niiden mekaanisen perusrakenteen sekä liikekoordinaatiston mukaan. Teollisuusrobottien yleiset rakenteet ovat: suorakulmainen, sylinteri, napakoordinaatisto, scara, kiertyvänivelinen sekä rinnakkaisrakenteinen. (Asp, Heinonkoski & Hyppönen 2008, 111.)

Nimitys pääakseleiden mukaan	Rakenne	Kinemaattinen kaavio	Työalue
Suorakulmainen robotti			
Sylinterirobotti			
Napa-koordinaatisto-robotti			
Scara-robotti			
Kiertyvänivelinen robotti			
Rinnakkais-rakenteinen robotti			

KUVIO 4 Yleisimpien robottityyppien rakenteet ja työalueet (Kuivanen 1999, 12.)

Suorakulmaisissa roboteissa kolme ensimmäistä vapausastetta ovat lineaarisia. Tyypillisin suorakulmainen robotti on portaalirobotti, jonka rakenne on tuettu työalueen nurkista palkeilla. Portaalirobotteja käytetään esim. elintarviketeollisuuden varastoinneissa. (Kuivanen 1999, 16.)

Sylinterirobotissa on yksi koko sen rakennetta kääntävä akseli, mutta sen muut akselit ovat lineaarisia. Sen nimitys tulee sylinterikoordinaatistosta. (Laitinen, Ylä-Outinen & Tahvanainen 2014.)

Napakoordinaatistorobotissa yksi akseli kääntää koko robottia ja toinen akseli robotin käsivartta pystysuunnassa. Sen muut akselit ovat lineaarisia. (Laitinen, Ylä-Outinen & Tahvanainen 2014.)

Scara-robotti on Japanissa kehitetty robottityyppi erityisesti kokoonpanotehtäviin. Niissä on kaksi tai kolme kiertyvää niveltä työkalun asemointiin vaakatasossa ja yksi lineaarinen nivel pystyliikkeisiin. Sen erityisominaisuuksia ovat suuret liikenopeudet sekä tarkkuus. (Keinänen, Kärkkäinen, Lähetkangas & Sumujärvi 2007, 259.)

Yleisin robottityyppi on kiertyvänivelinen robotti, jossa kaikki vapausasteet ovat kiertyviä. Useimmissa kiertyvänivelisissä roboteissa on kuusi vapausastetta, joka vaaditaan, jotta työkalun saa mihin tahansa paikkaan ja asentoon työalueella. Esimerkiksi hitsaukseen riittää kuitenkin vain viisi vapausastetta, koska hitsauslangan kiertymäkulmalla itsensä ympäri ei ole merkitystä. (Kuivanen 1999, 16-18.)

Toisin kuin muiden yleisten robottityyppien, rinnakkaisrakenteisen robotin rakenne on suljettu. Sen lineaariset nivelet on kiinnitetty yhteiseen tukivarteen, jonka asentoa ja paikkaa muutetaan nivelten liikkeellä. Suljetun kinemaattisen rakenteen etuna on keveys ja kestävyys, mutta työalue on rajattu. Niitä käytetään erityisesti elintarviketeollisuudessa kun tarvitaan suuria nopeuksia. (Kuivanen 1999, 16-18.)

2.7 Standardisointi

Robottien ilmoitettuja ominaisuuksia on vaikea vertailla eri valmistajien välillä. Tavallisimpia valmistajien ilmoittamia tietoja ovat työalueen profiili sivulta ja päältä, nivelten suurimmat nopeudet (°/s) ja liikealue (°), kuormankantokyky ja toistotarkkuus. (Kuivanen 1999, 14.)

2.7.1 Toistotarkkuus ja absoluuttinen tarkkuus

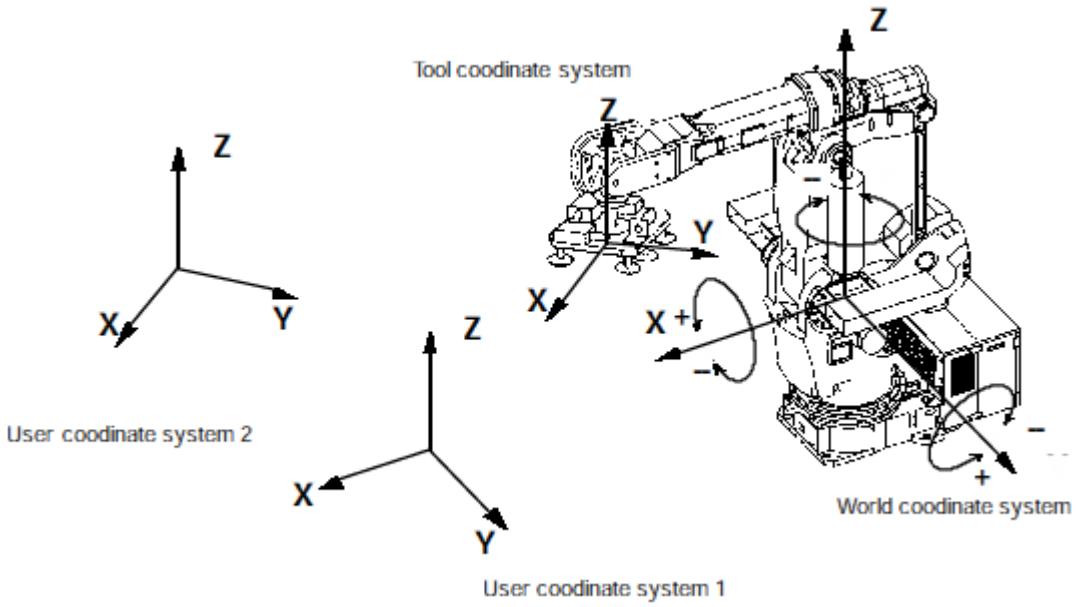
Toistotarkkuus tarkoittaa sitä tilastollista tarkkuutta, jolla robotti palaa takaisin aikaisemmin opetettuun pisteeseen, ja sitä ei pidä sekoittaa absoluuttiseen tarkkuuteen. Absoluuttiseen tarkkuuteen vaikuttaa esimerkiksi kuinka tarkka malli robotin liikeohjaimelle on robotista mallinnettu, ja se voi olla jopa kymmeniä tai satoja kertoja toistotarkkuutta epätarkempi. Robotin oma paino, työkalun ja työkappaleen painon lisäksi huonontavat absoluuttista tarkkuutta, koska robotin mekaniikka joustaa painon vaikutuksesta. Toistotarkkuuteen taas vaikuttaa toimilaitteiden ja toimilaitteiden anturoinnin resoluutio ja se on usein tarkempi kuin 0,1 mm. (Kuivanen 1999, 14.)

2.7.2 Kuormankantokyky

Robottien kantokyky voidaan ilmoittaa joko normaalikantokykynä (*nominal payload*) tai maksimikantokykynä (*max wrist load* jne.). Normaali kantokyky kertoo kuinka raskaan hyötykuorman (työkalu + liikutettava kappale) robotti pystyy liikuttamaan tarkkuuden ja nopeuden kärsimättä. Maksimikantokyky ilmoittaa kuinka suuren hyötykuorman vastaavasti robotti pystyy liikuttamaan tarkkuuden kärsimättä. Teollisuusrobotti siis useimmiten pystyy liikuttamaan huomattavasti raskaamman painon kuin sen ilmoitettu kantokyky on, mutta hitaammin ja huonommalla tarkkuudella. (Niku 2001, 15.)

2.8 Koordinaatistot

Koordinaatistojärjestelmä määrittelee robotin aseman ja asennon. Nivelkoordinaatiojärjestelmä määrittelee aseman ja asennon robotin nivelten kiertymillä. Karteesiakoordinaatistojärjestelmiä robotilla on useita ja niissä robotin asema ja asento määritellään X, Y ja Z – koordinaatein koordinaatiojärjestelmän origosta sekä kulmien W, P ja R siirrolla X, Y ja Z – akselien suhteen. (FANUC robotti R-J3 Ohjaus n.d., 85.)



KUVIO 5 Robotin koordinaatistoja (FANUC Robot series R-J3i Model B Operators Manual n.d., 176.)

Mekaanisen liitynnän koordinaatistojärjestelmä on vakio, joka on määritelty robotin rannelaipan pintaan. Sitä käytetään määrittelemään robottiin kiinnitetyn työkalun koordinaatisto. Käyttäjän ei tarvitse huolehtia mekaanisen liitynnän koordinaatistojärjestelmästä. (FANUC robotti R-J3 Ohjaus n.d., 86.)

Työkalun koordinaatistojärjestelmä määrittelee työkalun keskipisteen (*TCP, tool center point*). Työkalun koordinaatiston asettaa käyttäjä, esim. kolmen pisteen, kuuden pisteen tai suoraa osoitusmenetelmää käyttäen kohtaan, jossa työkalun keskipiste on. Jos sitä ei ole määritelty, robotti käyttää työkalukoordinaatistonaan mekaanisen liitynnän koordinaatistoa. (FANUC robotti R-J3 Ohjaus n.d., 86.)

Vapaan kappaleen koordinaatistojärjestelmä on kiinnitetty työtilaan. Tavallisilla roboteilla vapaan kappaleen koordinaatistojärjestelmä on sidottu robotin jalustaan. Määritelmän mukaan Z-akseli kulkee robotin ensimmäisen vapausasteen läpi, X-akseli osoittaa ensimmäisen nivelen työalueen keskikohtaan ja XY-taso yhtyy lattiaan. Vapaan kappaleen koordinaatiojärjestelmän mukaan määritellään käyttäjä koordinaatistojärjestelmät ja hitaan liikkeen koordinaatistojärjestelmä. Suomenkielisessä kirjallisuudessa vapaan kappaleen koordinaatistojärjestelmää kutsutaan nimellä peruskoordinaatisto tai yleinen koordinaatistojärjestelmä. Fanuc kutsuu englanninkielisissä oppaissaan (ks. Kuvio 5) vapaan kappaleen koordinaatistojärjestelmää *World coordinate systemiksi*, eli maailmakoordinaatistoksi, kun taas yleisesti maailmakoordinaatistoksi kutsutaan käyttäjän koordinaatistojärjestelmiä. (FANUC robotti R-J3 Ohjaus n.d., 86.)

Käyttäjäkoordinaatisto on robotin ulkopuolinen koordinaatisto, jonka käyttäjä voi määrittää robotin työskentely-ympäristöön, esimerkiksi kuljettimeen tai johonkin solun osaan. Käyttäjäkoordinaatistoja voi olla useita tallennettuna robotille. Yleisesti kirjallisuudessa käyttäjäkoordinaatistoa kutsutaan maailmakoordinaatistoksi, mutta Fanucin englanninkielinen materiaali viittaa maailmakoordinaatistolla robottiin sidottuun peruskoordinaatistoon. (Kuivanen 1999, 21.)

Hitaan liikkeen koordinaatiston määrittelee käyttäjä. Sitä käytetään vain siirtämään robottia tehokkaasti hitaan liikkeen tilassa (jog) eli käsiajossa. Hitaan liikkeen koordinaatisto ei siis ole käytettävissä robotin toimiessa itsenäisesti. (FANUC robotti R-J3 Ohjaus n.d., 86.)

2.9 Robottien ohjelmointi

Robottien ohjelmoinnissa robottisovellukseen laaditaan logiikka ja toimintajärjestys jolla tarvittavat liikkeet toteutetaan. Liikkeiden lisäksi robotille ohjelmoidaan muille laitteille lähetettävät signaalit sekä muilta laitteilta robotille luettavat signaalit (joiden avulla robotti voidaan tahdistaa muihin laitteisiin tai saada valmistuksen kannalta oleellista tietoa, jonka perusteella robottiohjelman parametreja muutetaan

suorituksen aikana). Robottien ohjelmoinnin evoluutio on kulkenut alun sähkömekaanisista kytkennöistä, johdattamalla ohjelmoinnin kautta opettamalla ohjelmointiin, josta edelleen etäohjelmointiin ja mallipohjaiseen etäohjelmointiin. (Kuivanen 1999, 78.)

Nykyään robottien ohjelmointia myös automatisoidaan esim. konenäköjärjestelmien avulla sekä automatisoidulla mallipohjaisella etäohjelmoinnilla, jossa tietokone päättelee CAD-kuvan perusteella robotille tehtävän ohjelman (Malm 2008, 96-98).

2.9.1 Johdattamalla ohjelmointi

Robottien ohjelmointi on aikoinaan alkanut sähkömekaanisista kytkennöistä, joiden avulla robotti on vaihe kerrallaan ajanut päin haluttuja rajakatkaisijoita. Jouhevuutta liikeratoihin saatiin kun ihmiset alkoivat lihasvoimin liikuttaa työkalua halutun radan mukaisesti ja nauhoittaa paikka-antureiden arvot instrumenttinauhuriin, josta ne voitiin myöhemmin toistaa robotin nivelten toimilaitteiden ohjearvoksi. Tätä tapaa, eli johdattamalla ohjelmointia, käytetään vielä ainakin maalausrobottien ohjelmoinnissa. Nykyisin robotit kuitenkin pääsääntöisesti ohjelmoidaan joko robotin käsiohjaimen avulla opettamalla tai tietokoneella etäohjelmointina. Usein näitä kahta menetelmää käytetään myös rinnakkain. (Kuivanen 1999, 78.)

2.9.2 Opettamalla ohjelmointi

Opettamalla ohjelmointi tehdään robotin käsiohjaimen avulla. Käsiohjaimella robotin niveliä voidaan ajaa joko nivel kerrallaan haluttuun asemaan tai halutun koordinaatiston sisällä työkalupäätä liikuttamalla. Kun työkalupää saadaan ajettua haluttuun pisteeseen, tallennetaan piste robotin muistiin. Näin tapahtuvan paikkojen opettamisen lisäksi robottiohjelman teksti (loogiset rakenteet jne.) voidaan kirjoittaa käsiohjaimen funktionäppäimistöllä. (Kuivanen 1999, 79.)

2.9.3 Tekstipohjainen etäohjelmointi

Yleisin tapa tehdä robottiohjelmia on opettaa robotille muutama asema käsivartta liikuttamalla, mutta tehdä lisää asemia sekä robotin toiminnan logiikka tietokoneella. Tietokoneella tehtävällä tekstipohjaisella etäohjelmoinnilla vapautetaan robotti työskentelemään sinä aikana kun ohjelmointia tehdään. (Kuivanen 1999, 78-79.)

2.9.4 Mallipohjainen etäohjelmointi

Mallipohjaisessa etäohjelmoinnissa tietokoneelle tehdään simuloitavasta tuotantosolusta 3D-malli CAD-ohjelmia hyödyntäen. 3D-malli ympäristössä toimiessa ohjelmoijan ei tarvitse osata robotin omaa ohjelmointikieltä, vaan ohjelmat voidaan tehdä näyttäen robotille tehtäviä. Mallipohjaista etäohjelmointia tehtäessä ohjelmoija voi myös varmistaa, että robotille suunniteltu liikerata ei sisällä törmäyksiä. Lisäksi sellville saadaan suoritusaika, joka robotilta kuluu toimintojen suorittamiseen. (Kuivanen 1999, 81-82.)

Mallipohjaisissa etäohjelmointiohjelmistoissa, kuten Delmiassa, on omat moduulinsa ainakin 3D-mallintamiselle, kinemaattisten mekanismien suunnitteluun, layoutin suunnitteluun sekä lisäksi laitekirjastot. 3D-mallit valmistettavista tuotteista, tuotantosolun irtaimistosta ja oheislaitteista voidaan tehdä omalla CAD-moduulilla tai erillisessä CAD-ohjelmassa. (Kuivanen 1999, 82-83.)

Mallipohjaisella ohjelmointijärjestelmällä voidaan tehdä etäohjelmoinnin lisäksi myös tuotannon simulointia, jolla voidaan varmistaa uuden tuotteen valmistettavuus. Simuloinnissa käyttäjä seuraa näytöltä ohjelman suorittamista. Ohjelmiston törmäystarkastelun ollessa päällä ohjelman suoritus pysähtyy, jos ohjelmisto huomaa robotin törmäävän johonkin. Simulointi pysähtyy myös, jos robotti ajaa itsensä singulariteettiin tai joku sen nivelistä ylittää akselin suurimman mahdollisen nivelkulman. Sen jälkeen käyttäjän on korjattava ohjelmaa, jotta se toimisi virheettömästi. Simulointia voidaan ajaa myös tausta-ajona, jolloin simuloinnin päätyttyä ohjelma ilmoittaa ongelmakohdista. (Kuivanen 1999, 82-86.)

2.9.5 Robottien ohjelmointikielet

Ohjelmointikieli on rajapinta teollisuusrobotin ja ihmisen välillä. Sen avulla ihminen voi ohjelmoida teollisuusrobotille halutun toiminnallisuuden, jota robotti noudattaa. Jokaisella robottivalmistajalla on oma robottiohjelmointikielensä. Useat niistä perustuvat yleisille ohjelmointikielille kuten Basiciin, Pascaliin ja C-kieleen. Robottien ohjelmointikielet voidaan jakaa usealle eri tasolle, lähtien konekielestä ja päättyen korkean tason ohjelmointikieliin. Korkean tason ohjelmointikielillä kirjoitetut ohjelmat käännetään kääntäjällä robotin mikroprosessorin ymmärtämäksi konekieleksi. (Niku 2001, 17-18.)

2.10 Fanuc-robotin liikekäsky

Fanucin liikekäsky koostuu viidestä eri elementistä. Liikeformaatti kertoo minkä tyyppisellä liikkeellä asemaan siirrytään ja asematieto kertoo asematiedon numeron, jonka perusteella haetaan asematieto. Syöttöaste kertoo nopeuden, jolla asemaan siirrytään ja asemointireitti paikoituksen asemoidutaanko pisteeseen vai ohitetaanko se. Viimeisenä liikekäskyssä on lisäkäskyt. (RJ3 Programming Manual n.d., 140.)

2.10.1 Liikeformaatit

Nivelliikkeen (*Joint*) aikana robotti liikuttaa kaikkia akseleitaan samanaikaisesti päästäkseen kohdeasemaan. Nivelliike on yleisin liiketyyppi ja käytännöllinen silloin kun työkalun orientaatiolla ei ole väliä liikkeen suorituksen aikana. Nopeus annetaan prosentteina maksiminopeudesta tai suoritusaikana. (RJ3 Programming Manual n.d., 41.)

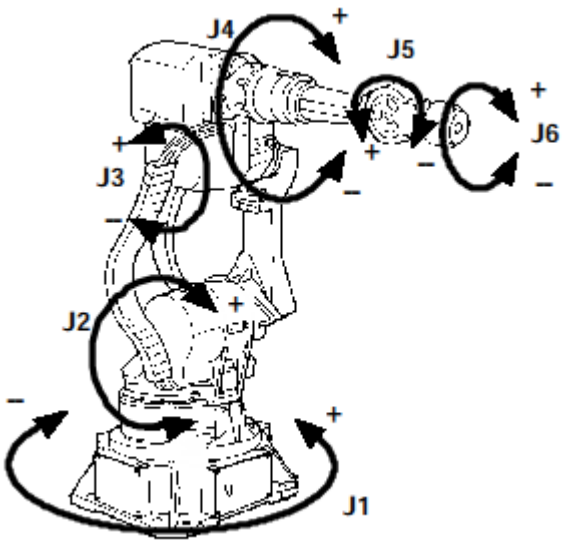
Lineaariliikkeellä (*Linear*) robotin työkalupää liikkuu lähtö- ja kohdeaseman välillä suoraa linjaa pitkin. Joissakin työtehtävissä työkalun on välttämätöntä säilyttää orientaationsa, jolloin lineaariliike on välttämätön. Nopeus lineaariliikkeelle voidaan antaa esim. mm/s, cm/min tai suoritusaikana. (RJ3 Programming Manual n.d., 42.)

Ympyräliikkeessä (*Circular*) robotti liikuttaa työkalupäätä ympyrän kaarta pitkin lähtöasemasta väliaseman kautta kohdeasemaan. Ympyräliikkeeseen ohjelmoidaan sekä

väliasema että tavoiteasema muiden liiketyyppien sijaan, jotka tarvitsevat vain kohdeaseman. Nopeus ympyräliikkeelle annetaan esim. mm/s, astetta/s tai suoritusaikana. (RJ3 Programming Manual n.d., 43.)

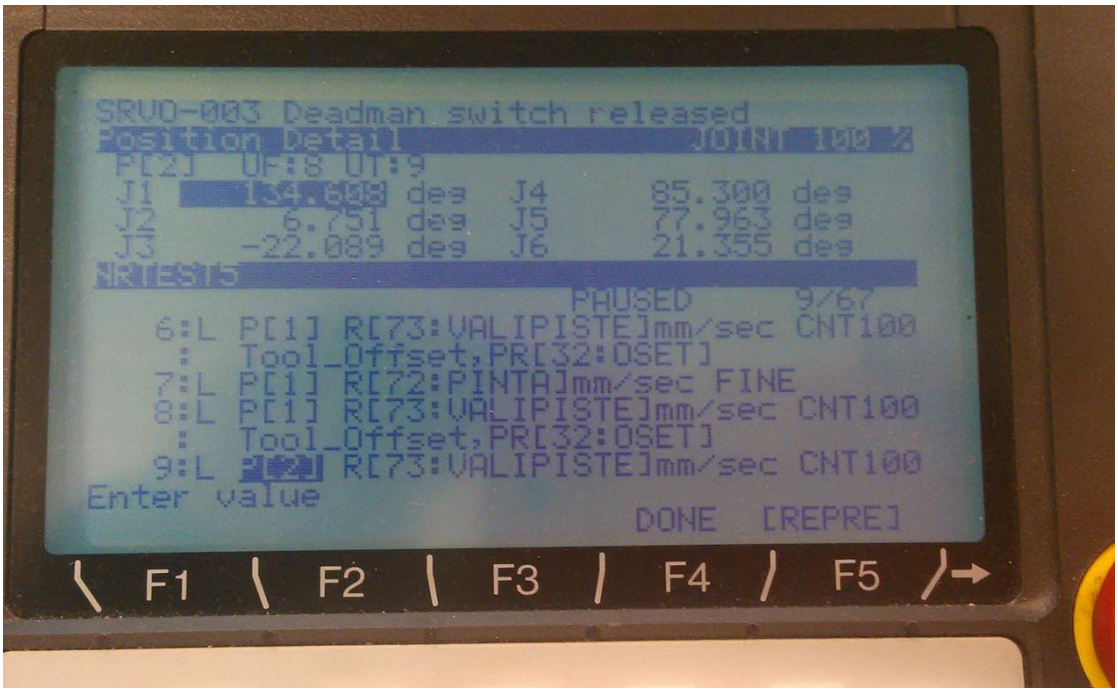
2.10.2 Asematiedon esitystavat

Karteesia-koordinaateilla pisteet ovat tallennettu robotin muistiin X, Y ja Z akselin arvoina sekä kiertymäkulmina W, P ja R näiden akselien ympäri. Paikkatietojen lisäksi pisteen tietoihin on tallennettu missä käyttäjä- ja työkalukoordinaatistossa se sijaitsee sekä robotin konfiguraatio kyseisessä pisteessä. (RJ3 Programming Manual n.d., 44.)



KUVIO 6 Kuusiakselisen Fanuc-nivelvarsirobotin akselit (FANUC Robot series R-J3i Model B Operators Manual n.d., 412.)

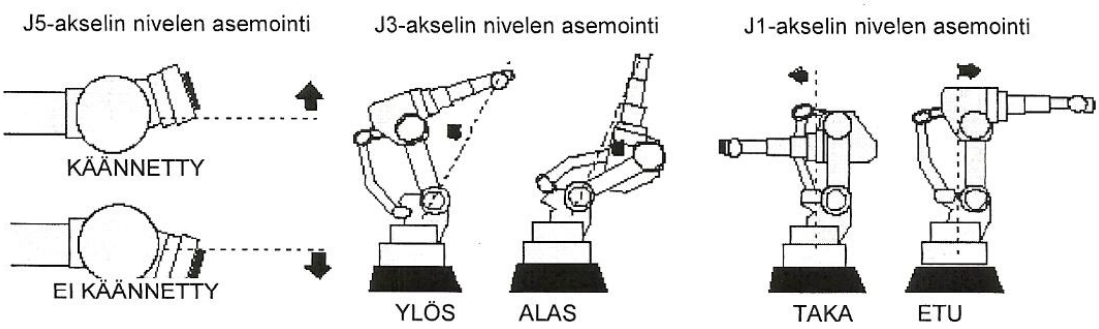
Karteesia-koordinaattien lisäksi robotille tallennetut pisteet voidaan esittää myös robotin akselien nivelkulmina. Tallennettuna on siis kaikkien kuuden akselin (J1 – J6) kääntymiskulmat, joilla työkalupää on tallennetussa pisteessä. (RJ3 Programming Manual n.d., 44.)



KUVIO 7 Piste tallennettuna akselien nivelkulmina

2.10.3 Konfiguraatio

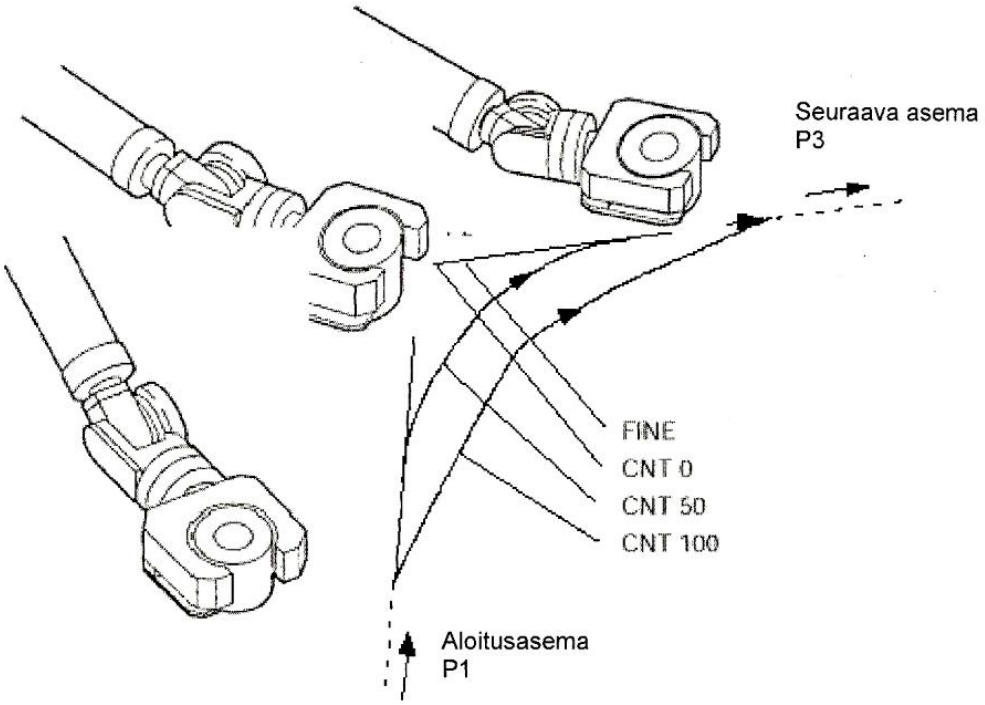
Robotin konfiguraatio edustaa sen asentoa. Konfiguraatio tarkoittaa tässä yhteydessä sitä mihin suuntaan robotin nivelet ovat kääntyneet. Koska kuusiakselisella robotilla on mahdollista päästä samoihin pisteisiin useilla eri konfiguraatioilla, tarvitsee pisteisiin määritellä myös konfiguraatio mitä halutaan käyttää. Esimerkiksi Fanucin nivelvarsirobotilla määritellään ranteen nivelen (J5) kääntö/ei kääntöä (*Flip/Noflip*), osoittaako nivel J3 ylös vai alas (*Up/Down*) ja osoittako nivel J1 eteen vai taakse (*Back/Front*). Perinteisesti kuvissa robotti on asennossa FUT tai NUT (ks. Kuvio 6). (FANUC Robot series R-J3i Model B Operators Manual n.d., 177.)



KUVIO 8 Fanuc nivelvarsirobotin konfiguraatiot (Fanuc robotti R-J3 Ohjaus n.d., 144.)

2.10.4 Paikoitus

Paikoituksella (*Termination*) määritetään robotin liikkeen lopetuksen tarkkuus. FINE eli hienopaikoituksella robotti pysähtyy tavoitetasemaan ennen kuin se jatkaa siirtymistä seuraavaan pisteeseen. CNT eli jatkuvalla paikoitukselle annetaan arvo (0-100), joka kertoo kuinka lähelle tavoitetasemaa sen on lähestyttävä. CNT-paikoitusta käyttäessä robotti ei mahdollisesti käy tavoiteasemassa ollenkaan, vaan ohittaa sen kauempaa. CNT100 arvolla robotti liikkuu nopeimmin ja kauimpana tavoiteasemasta. (FANUC robotti R-J3 Ohjaus n.d., 150.)



KUVIO 9 Fine ja CNT paikoituksen ero (Fanuc-robotti R-J3 Ohjaus n.d., 150.)

2.10.5 Muut liikekäskyt

Liikekäskyn perään voidaan lisätä muita lisäkäskyjä. Lisäkäskyjä ovat esimerkiksi kiihtyvyyskäsky ACC, jolla robotin liikkeiden kiihtyvyyteen voidaan vaikuttaa tai työkalun offset käsky, jota on hyödynnetty kuviossa 7. (FANUC robotti R-J3 Ohjaus n.d., 151.)

3 Simulointi

Simuloinnissa pyritään jäljittelemään todellista maailmaa, samoin kuin esimerkiksi ihminen voi simuloida mielikuvituksessaan todellista maailmaa. Tietokoneella tehtävässä simuloinnissa tietokoneelle luodaan virtuaalinen malli todellisesta maailmasta. Kun teollisuusrobotteja halutaan simuloida tietokoneella, luodaan CAD-ohjelmalla 3D-malli robotin toimintaympäristöstä, esimerkiksi tuotantosolusta, johon robotti tultaisiin asentamaan. Lisäksi tarvitaan simuloitavasta robotista 3D-malli, joka voidaan saada robottivalmistajalta, simulointiohjelmiston (tai etäohjelmointiohjelmiston) valmiista kirjastosta tai tarvittaessa myös tehdä itse esim. CAD-ohjelmaa hyödyntäen. (Calin, D 2013.)

Robotiikan simuloimisen hyödyt:

- Tuotteen valmistettavuutta voidaan testata etukäteen
- Mahdollisuus tehdä kokeiluita ilman laitehankinnoista koituvia kustannuksia
- Sovellukset voidaan siirtää oikealle robotille ilman muutoksia tai pienillä muutoksilla
- Monimutkaisissa tapauksissa simulointi voidaan tehdä vaiheittain

Robotiikan simuloimisen haitat:

- Simuloidaan vain sitä mitä on ohjelmoitu simuloitavaksi
- Oikeassa maailmassa robotille saattaa tulla vastaan tapauksia joita ei simuloidessa tiedetä tai huomata

(Most Advanced Robotics Simulation Software Overview 2014.)

3.1 Delmia-ohjelmisto

Delmia-ohjelmistolla voidaan suunnitella, simuloida ja mallintaa tuotantoprosesseja. Ohjelmiston avulla tuotteen valmistettavuus voidaan selvittää ja tuotantoprosessi suunnitella jo varhaisessa vaiheessa tuotekehityksen aikana. Delmiassa voidaan tuotteen kokoonpanon kuluva aika määrittää ja tuotantokustannukset arvioida jo suunnitteluvaiheessa. Delmian avulla voidaan myös suorittaa teollisuusrobottien mallipohjaista etäohjelmointia, jota varten ohjelmistosta löytyy laaja kirjasto teollisuusrobotteja useilta eri valmistajilta. (Digitaalinen valmistus n.d.)

3.1.1 Laitteiden tehtävien määrittely

Delmiassa on useita eri työtiloja, joista pitää valita sopiva kuhunkin tapaukseen ja simuloinnin vaiheeseen. Näiden seuraavaksi esiteltävien työtilojen lisäksi Delmiassa on kymmeniä työtiloja erilaisiin tilanteisiin, kuten esim. useita työtiloja joiden avulla ihminen saadaan mallinnettua simulaatioon sekä omat työtilansa erilaisten työstökoneiden mallinnukselle ja etäohjelmoinnille. Laitteiden tehtävien määrittelyssä *(Device task definition*) prosessiin tuodaan 3D-mallit käytettävistä laitteista ja tuotteista, ja ne asetellaan haluttuun järjestykseen ja niille ohjelmoidaan halutut toiminnot. Tässä työtilassa on myös katalogi, josta eri valmistajien valmiit robottimallit voidaan tuoda prosessiin. Laitteiden tehtävien määrittelyssä siis erillisistä malleista luodaan kokonaisen tuotantosolun malli. (Delmia documentation, Device task definition 2010.)

3.1.2 Työsolun ajastus

Työsolun ajastus (*Workcell sequencing*) on 3D-simulointi työtila, jolla voidaan simuloida tuotantoprosessia. Työsolun malliin tulevia robottien ja muiden laitteiden työtehtäviä voidaan ajastaa työsolun ajastus työtilassa. Työvaiheiden synkronoinnissa käytetään virtuaalisia IO-signaaleja. Työsolun ajastus tilassa tehdään myös varsinainen työsolun simulointi, josta saadaan tiedoksi työvaiheiden tahtiajat. Lisäksi simuloitaessa kaikki solun resurssit yhtä aikaa, varmistetaan tuotantoprosessin toimivuus,

sekä valmistukseen kuluva kokonaisaika. (Delmia documentation, Workcell sequencing 2010.)

3.1.3 Robotin offline-ohjelmointi

Robottien offline-ohjelmointi (*Robot offline programming*) on työtila Delmiassa, jonka avulla voidaan kääntää Delmian V5-robotti ohjelmat oikeiden teollisuusrobottien ymmärtämille kielille. Myös Delmiaan voidaan tuoda ohjelmia, jotka on kirjoitettu robottien omilla ohjelmointikielillä, ja kääntää niistä Delmia V5-robottiohjelmia, joita voidaan simulointimallissa suorittaa. (Delmia documentation, Robot offline programming 2010.)

3.2 Roboguide

Yleisten simulointi- ja etäohjelmointiohjelmistojen lisäksi jokaisella robottivalmistajalla on oma ohjelmansa omien robottiensa simulointiin ja etäohjelmointiin. Fanucin nykyinen ohjelmisto on nimeltään Roboguide. Roboguidessa robottiohjelmat voidaan tehdä näyttämällä tai Roboguiden virtuaalisella käsiohjaimella, joka vastaa Fanucin oikealla käsiohjaimella ohjelmointia. Roboguiden tahtiaikojen simuloinnin pitäisi olla hyvin tarkka, koska siihen on mallinnettu Fanucin ohjausjärjestelmä. Roboguiden ominaisuuksia ovat esimerkiksi törmäystarkastelu, AVI-videoiden luominen prosessista sekä profiler-funktio, jolla analysoidaan jokaisen ohjelma rivin suorittamiseen kuuluvaa aikaa. (Roboguide n.d.)

4 Simulointimallin luominen Delmia-ympäristöön

Työvaiheen simuloimiseksi Delmia-ohjelmistoon tehtiin työvaiheesta simulointimalli. Mallia varten asiakasyritykseltä saatiin 3D-mallit muutamasta heidän tuotteestaan, jotka kattoivat heidän mukaansa heidän tuoteportfolionsa riittävästi. Tuotteiden 3D-malleista oli salassapito syistä karsittu pois kaikki muu tieto, paitsi kohdat joihin osia tultaisiin tuotteessa asentamaan. Osien asennuspaikat piti käsin määrittää Delmia-

ohjelmistolle, sillä Delmiassa ei ollut valmista funktiota tässä muodossa olevien koordinaattien käyttämiseksi robottiohjelmassa. Osien asennuspaikat määritettiin hiirellä näyttämällä tageiksi, joita niitä voitiin käyttää robotille ohjelmaa muodostettaessa. Asiakasyritys halusi myös tuotteen liikkuvan osien asennuksen aikana, joten simulointimalliin tehtiin mekanismi, jolla tuotetta voitiin liikuttaa. Tagit piti sitoa mekanismiin, jotta ne liikkuisivat myös mukana mekanismia liikuttaessa.

KUVIO 10 SALATTU

4.1 SALATTU

4.2 Robotin tuominen simulaatioon

Asiakasyrityksen tiloissa käytiin tekemässä yksinkertainen 3D-malli asiakkaan nykyisestä tuotantosolusta, jotta nähtiin kuinka teollisuusrobotti siihen sopii. Simulointimalliin tuotiin tuotteen ja tuotantoympäristön jälkeen teollisuusrobotti. Teollisuusrobotti tuotiin Delmian valmiista kirjastosta. Robotiksi simulointiin valittiin Fanucin M16iB/20, koska asiakkaalla oli kokemusta Fanucin roboteista. Lisäksi Jyväskylän ammattikorkeakoululla oli kyseinen robotti. M16iB/20 oli myös kooltaan ja kuormankantokyvyltään sopivin Fanucin robotti työvaiheeseen. M16iB/20-robottia ei ole enää myynnissä, vaan sen on korvannut Fanucin mallistossa M20iA-sarjan robotit. JAMK:n Delmia versiossa ei kuitenkaan ollut aivan uusimpia robotteja eikä Fanucin maahantuoja Fastemsillakaan ollut Delmia yhteensopivia malleja uudemmista roboteista. Näin päädyttiin simuloinnissa käyttämään hieman vanhemman sukupolven robottia. M16iB/20-robotin lisäksi päätettiin kokeilla simuloida myös muita Fanucin robotteja, jotka Delmiassa olikin helppo jälkikäteen vaihtaa M16iB/20:n tilalle simulointimalliin.

Robotti asetettiin simulointimalliin maan tasolle ja suurin piirtein keskelle tuotetta liikuttelevaa laitteistoa. Etäisyys laitteeseen asetettiin silmämääräisesti. Robotin paikkaa oli mahdollista myös myöhemmin helposti muuttaa, jos se osoittautuisi olemaan esimerkiksi liian kaukana työkappaleesta

KUVIO 11 SALATTU

4.3 Työkalu

Asiakasyritykseltä saatiin tuotteiden 3D-mallien lisäksi myös 3D-malli heillä nykyisin työvaiheessa käytössä olevasta työkalusta. Työkalun malliin mallinnettiin yksinkertainen suppilonmallinen osa, josta se tultaisiin kiinnittämään robotin työkalulaippaan. Sitä millaisella osalla oikeasti työkalu tultaisiin kiinnittämään robottiin, ei vielä mietitty. Kiinnitysosaan piti määritellä peruskoordinaatisto, jolla se simulointimallissa liittyy robotin työkaluakselin koordinaatistoon. Peruskoordinaatiston lisäksi työkalulle piti määritellä myös sen työkalukoordinaatisto, eli kohta joka on esimerkiksi sormitarttujan keskipiste. Kun työkalun koordinaatistot oli määritelty, se voitiin liittää prosessiin. Delmiassa se kiinnitettiin robotin työkalulaippaan ’Set tool’-toiminnolla. Tuotteen liikuttamismekanismille tehtiin ohjelma, jossa se liikuttaa tuotteen haluttuun kohtaan, jonka saavutettuaan se laittaa virtuaalisen lähdön päälle. Virtuaalinen lähtö ilmaisee robotille, että tuote on halutussa kohdassa, ja työvaiheen suoritus voidaan aloittaa.

KUVIO 12 SALATTU

KUVIO 13 SALATTU

4.4 SALATTU

KUVIO 14 SALATTU

KUVIO 15 SALATTU

KUVIO 16 SALATTU

4.5 SALATTU

4.5.1 SALATTU

4.5.2 SALATTU

4.6 Simulointi

Delmiassa ajettiin robotille useita testejä vaihdellen robotin liikenopeuksia, liikkeen tarkkuuksia ja kiihtyvyysarvoa. Robotille voitiin opettaa useita nopeuksia ja tarkkuuksia jne. profiileiksi, joiden arvoja voitiin muuttaa. Profiileita voitiin sitten kutsua ohjelmasta, jolloin esimerkiksi robotti suoritti liikkeen haluttuun profiilin syötetyllä nopeudella. Testiajoista otettiin kokonaisajat ylös, jotta nähtiin miten eri arvot vaikuttavat suorituksen nopeuteen. Nopeus, tarkkuus jne. arvojen lisäksi testattiin myös liiketyyppejä. Tuotteen pinnalle laskeutuminen pitää tehdä lineaariliikkeellä, jotta osa saadaan tuotteeseen asennettua varmasti halutussa kulmassa, mutta väliliikkeissä työkalun asennolla ei ole merkitystä. Niinpä väliliikkeissä olisi käytännöllistä käyttää robotin akseleille hellävaraisempaa ja yleisesti nopeampaa nivelliikettä. Delmian simulointimallissa nivelliikkeen käyttö ei kuitenkaan onnistunut, vaan työkalu suoritti omituisia liikesarjoja. Olisikin mielenkiintoista päästä kokeilemaan, onko myös oikealla robotilla vastaavia ongelmia liikkuvan kohteen seurannassa nivelliikkeellä, vai onko tässä kohtaa joku bugi Delmia-ohjelmistossa.

KUVIO 17 SALATTU

KUVIO 18 SALATTU

Robotin nopeuden asetusarvoa muuttaessa suoritusaika alkoi kasvaa vasta kun nopeuden asetti alle 800 mm/s, vakiona nopeuden asetusarvon ollessa robottityypin maksimi 2000 mm/s. Robotti ei siis kerkeä kiihtyä tässä tapauksessa olevilla lyhyillä matkoilla maksiminopeuteensa. Suoritusnopeudeksi työvaiheelle saatiin lopulta jokaisella simuloidulla tuotteella merkittävästi nopeampi aika kuin asiakkaan nykyisellä laitteistolla. Tässä vaiheessa vaikutti siis siltä, että asiakasyrityksen toivomaan suori-

tusnopeuteen päästäisiin teollisuusrobottipohjaisella ratkaisulla helposti. Lisäksi robottien tietoja tutkimalla voitiin olettaa, että Fanucin uudemmat M10 ja M20-sarjan robotit pystyisivät n. 10 – 20 % nopeampaan suoritusaikaan kuin M16iB/20-robotti, jolla tämä simulointi tehtiin. Asiakkaalle nopeutta tärkeämpää oli kuitenkin työvaiheen laatu, joka voitaisiin nähdä vasta tekemällä kokeilu oikealla teollisuusrobotilla.

4.7 SALATTU

4.8 SALATTU

4.9 SALATTU

4.10 Vaihtoehtoisten robottien testaaminen

Delmiassa oli helppo kokeilla myös simulointimallia muilla roboteilla. Fanucin Delmiassa olleista malleista M16iB/20-robottia lähimpänä olivat pienempi M6iB/6S sekä suurempi M710iC/50, joista jälkimmäinen löytyi myös asiakasyritykseltä. Soluun tuotiin alkuperäisen robotin lisäksi siis vaihtoehtoinen robotti, asetettiin sille sama työkalu kuin alkuperäiselle ja linkitettiin sen tuotteen liikutusmekanismi sille kuljettimeksi. Lisäksi uudelle robotille tehtiin samannimiset nopeus-, tarkkuus-, ja kiihtyvyysprofiilit kuin alkuperäiselle, jotta ohjelma toimi halutusti. Robottiohjelma pystyttiin kopioimaan suoraan alkuperäiseltä robotilta uudelle robotille. Kun tarvittavat siirrot oli tehty, voitiin alkuperäinen robotti poistaa ja siirtää uusi robotti sen paikalle soluun.

Testeissä käytettyjen kolmen robotin (M16iB/20, M6iB/6S ja M710iC/50) ja niiden eri variaatioiden välillä ei löytynyt useissa testeissä mitään suurempia nopeuseroja, mikä sai epäilemään simuloinnin luotettavuutta. Delmian ohjekirjassa kerrottiinkin, että Delmian tahtiaikojen simulointi ei välttämättä ole kovinkaan tarkka. Lisää tarkkuutta simulointiin saisi RRS (*Realistic Robot Simulation*) Delmian lisäpalikalla, joka JAMK:lta

Delmia-ohjelmistoon löytyikin. RRS:n käyttäminen kuitenkin edellyttäisi oikeaa robottiohjainta, joka toimisi palvelimena Delmialle ja kertoisi todelliset suoritusajat työvaiheissa. Mistään ei kuitenkaan löytynyt tietoa miten Fanucin robottiohjain saataisiin valjastettua palvelimeksi, joten todettiin että simuloinnin luotettavuus pitäisi varmistaa jollain muulla tavalla. Parhaaksi ratkaisuksi katsottiin tehdä vertailusimulointi Fanucin omalla simulointiohjelmistolla (Roboguide) sekä kokeilla saman ohjelman suorittamista myös JAMK:n M16iB/20-robotilla.

5 Vertailutesti nopeuden varmistamiseksi

Jotta Delmialla tehty simuloinnin paikkansapitävyys voitaisiin varmistaa, päätettiin tehdä vertailuohjelma joka ajettaisiin sekä Delmiassa, Roboguidessa että JAMK:n M16iB/20-robotilla. Ohjelmaksi valittiin yhden tuotteen ohjelmasta kaksikymmentä ajettavaa koordinaattia, sillä vertailutestissä ei voitu käyttää tuotteen liikuttamismekanismia. Liikkuvan kappaleen seuranta olisi ollut suuritöinen, ja vaatinut projektiin huomattavasti lisää rahallista panostusta. Liikkeen seuranta olisi pitänyt toteuttaa joko pulssianturin ja lisäoption ohjelmistoon (epätarkka ratkaisu) avulla tai erillisellä Fanucin ulkoisella akselilla (kallis ratkaisu). Todettiinkin että paikallaan pysyvillä koordinaateilla saadaan aivan yhtä hyvin selvitettyä erot, mitä simulointiohjelmistojen ja reaalimaailman välillä saattaa olla.

5.1 Roboguide

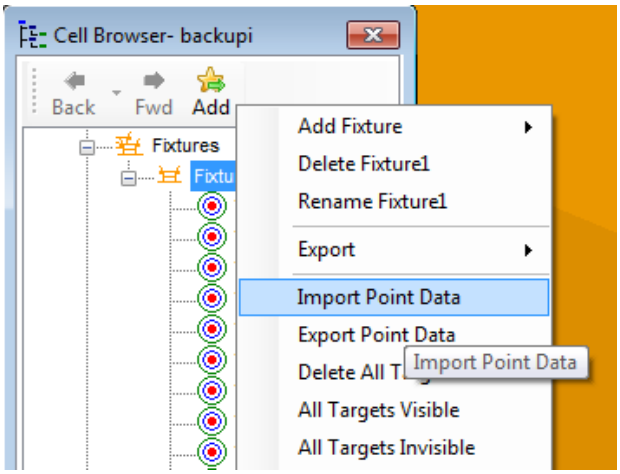
Työmäärältään vähäisimmäksi tavaksi toteuttaa ohjelman siirto Delmiasta Roboguideen todettiin olevan ohjelman siirtämisen sijaan siirtää pelkästään koordinaatit, jonka jälkeen luoda niiden perusteella uusi robottiohjelma. Koordinaatit Delmiasta saatiin ulos luomalla robottiohjelma. Delmialle tarvitsi kertoa missä Java SDK:n exe-tiedosto sijaitsee, jotta Delmian sisäisellä kielellä oleva robottiohjelma saatiin käännettyä Fanucin ymmärtämään muotoon. Delmian sisäisellä robottiohjelmointikielellä tehty ohjelma olisi voitu kääntää myös esimerkiksi Daihen, Huyndai, Kawasaki

ja KUKA-robottien ymmärtämään muotoon. Fanucin ohjelma tulostui ulos tiedostomuodossa, jonka pystyi avaamaan esimerkiksi aivan tavallisella tekstieditorilla. Tiedoston alkuosassa oli ns. otsikkotietoja, ohjelman nimi, päivämäärä jne. Otsikkotietojen jälkeen sijaitsi varsinainen robottiohjelma, jossa oli robotin liikekäskyt. Tiedoston lopussa oli ohjelmassa käytettävien pisteiden sijainnit. Pisteen sijainnin tiedoissa oli minkä numeroisessa työkalu- ja käyttäjäkoordinaatistossa se oli tallennettu, robotin akselien konfiguraatio pisteessä (esim. NUT) sekä pisteen koordinaatit (X, Y ja Z) millimetreinä sekä kiertymät (W, P ja R) asteina.

KUVIO 19 SALATTU

5.1.1 Ohjelman generoiminen

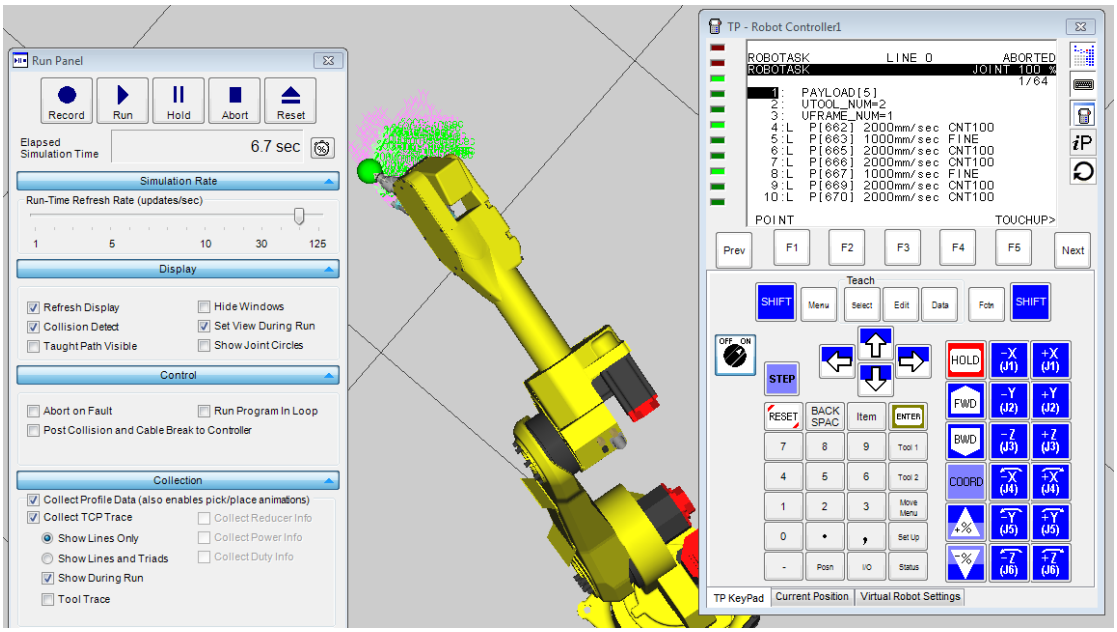
Roboguiden import-toimintoja tutkimalla selvisi, että helpoiten pisteiden koordinaatit saataisiin Roboguideen tallentamalla ne taulukkomuodossa. Tätä varten robottiohjelman sisältäneestä tiedostosta pisteiden koordinaatit siirrettiin käsin taulukkoon. Soluun luotiin fikstuuri, johon voitiin tuoda piste informaatiota tiedostosta, jotta taulukossa olevat koordinaatit saatiin Roboguideen. Koordinaateista tuli näin kohteita, joihin robotti voidaan ajaa ja jotka ovat sidottuina fikstuurin koordinaatteihin. Koordinaattien tuonnin jälkeen luotiin uusi ’Target group’, johon lisättiin äsken tuodut kohteet. Niiden perusteella Roboguide osasi generoida robottiohjelman, johon voitiin valikosta valita liikkeiden parametrit sekä tarvittaessa lisätä offset-pisteet halutulle etäisyydelle. Käyttäjä pystyy myös valitsemaan halutut käyttäjä- ja työkalukoordinaatistot ohjelmalle. Ohjelma siirtyi generoinnin jälkeen virtuaalisen robottiohjaimen ohjelmat valikkoon, josta sitä voitiin heti kokeilla. Virtuaalisella käsiohjaimella siihen voitiin myös jälkikäteen tehdä muutoksia aivan kuten oikeallakin robotin käsiohjaimella.



KUVIO 20 Koordinaattien tuominen Roboguideen

5.1.2 Ohjelman suorittaminen

Ohjelman lisäksi Roboguiden virtuaaliselle robottiohjaimelle asetettiin työkalukoordinaatisto samaksi kuin oikealle JAMK:n robotillekin. Työkalukoordinaatiston koordinaatit katsottiin JAMK:n robotin käsiohjaimesta ja samat koordinaatit asetettiin virtuaaliseen robottiohjaimeen virtuaalisella käsiohjaimella. Tämän jälkeen malli olikin valmis simuloitavaksi. Tätä 20-paikkaista ohjelmaa ajettiin myös useilla eri liikkeen parametreillä ja ajat otettiin ylös excel-taulukkoon, jotta niitä voidaan vertailla Delmialla simuloituihin aikoihin sekä oikealla robotilla ajetuilla ajoilla.



KUVIO 21 Simuloiminen Roboguide-ohjelmassa

5.2 Ohjelman siirto Delmiaan sekä JAMK:n robotille

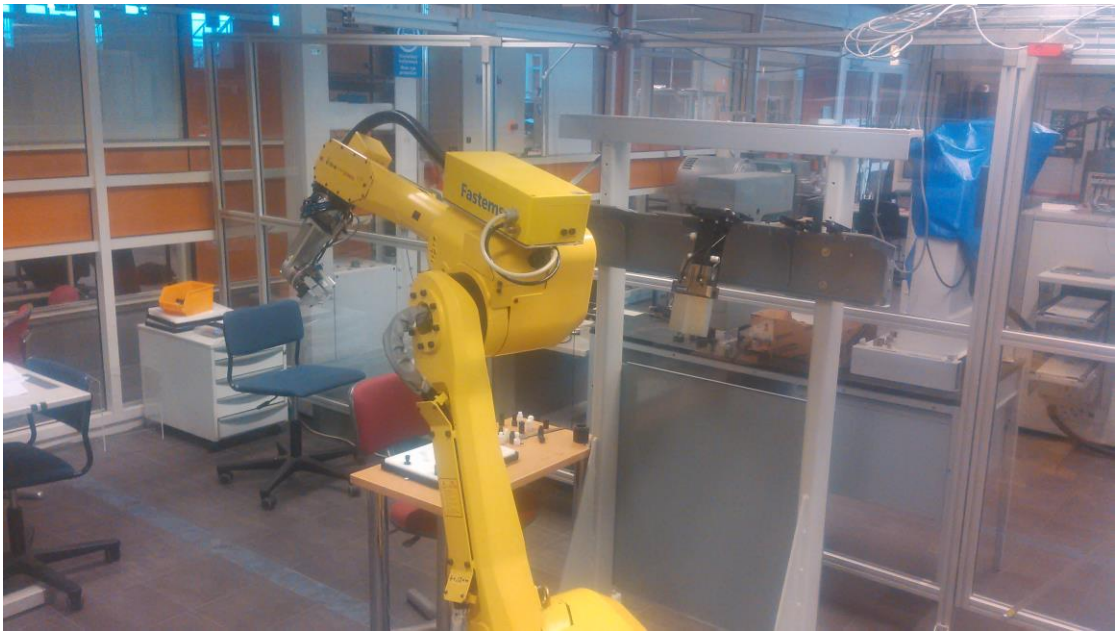
Roboguidella tehtyjen simulointien jälkeen oli aika siirtää ohjelma takaisin Delmiaan sekä JAMK:n M16iB/20-robotille. Karteesia-koordinaattien sijasta päädyttiin käyttämään robotin nivelkoordinaatteja, jotta testiohjelman paikat saataisiin koordinaatistoista riippumatta varmasti molempiin simulointiohjelmiin sekä vielä JAMK:n robotille samaksi. Nivelkoordinaatit saatiin esiin virtuaalisesta käsiohjaimesta vaihtamalla koordinaattien esitystapa karteesia-koordinaateista nivelkoordinaateiksi.

5.2.1 Delmia

Delmiaan nauhoitettiin uudet pisteet ajamalla ’jog’-toiminnolla robotin nivelet samaan asemaan kuin Roboguidessa ja sen jälkeen tallentamalla kyseinen kohta uudeksi tagiksi. Tagien perusteella voitiin taas luoda robottiohjelma, jota ajettiin samoilla parametreilla kuin Roboguide-ohjelmaakin. Delmiassakin ajat otettiin ylös niiden vertailemiseksi.

5.2.2 JAMK:n M16iB/20-robotti

Jyväskylän ammattikorkeakoulun Fanucin M16iB/20-teollisuusrobottiin tehtiin käsiohjaimella uusi robottiohjelma, joka vastasi Roboguideen sekä Delmiaan tehtyjä ohjelmia. Käsiohjaimella lisättiin uusia pisteitä, joiden nivelkoordinaatit vaihdettiin käsin samaksi kuin Roboguide ohjelman virtuaalisessa käsiohjaimessa. Jotta liikkeen parametreja oli helpompi muuttaa, robotilla otettiin käyttöön rekisterit liikkeen nopeuden määrittämiseksi, jotka alustetaan robottiohjelman alussa. Näin kaikkien samanlaisten liikkeiden nopeutta voidaan muuttaa yhdestä kohtaa ja käsiohjaimella näppäilyä tuli huomattavasti vähemmän kun robotin suoritusnopeuksia muutettiin. Lisäksi offset-pisteet pystyttiin kätevästi määrittelemään 'Tool_offset'-lisäkäskyllä, joka siirtää paikkarekisteriin tallennetun arvon mukaisesti tallennettua pistettä työkalun suuntaisesti. Oikealla robotilla suoritettiin robottiohjelma samoilla liikkeen parametreilla kuin simulointiohjelmilla ja suoritukset kuvattiin, jotta suoritukseen kuluva aika saatiin selville. Myöhemmin selvisi, että robottiohjelmaan olisi myös voinut laittaa ajastimen, jolla suoritusajan mittaus olisi ollut käytännöllisempää.



KUVIO 22 JAMK:n M16iB/20-robotti

5.3 Tulosten vastaavuus

Kun kaikki ajat oli kirjattu excel-taulukkoon, ryhdyttiin niitä vertailemaan keskenään. Nopeasti ilmenikin, että testiin käytetyissä ajoissa oli suuria eroja ohjelmien ja oikean robotin välillä. Delmialla tehty simulointi oli nopein, Roboguidella tehty simulointi hieman sitä hitaampi ja oikealla robotilla suoritettu testiajo huomattavasti näitä kahta hitaampi. Jotta erojen syitä pystyttäisiin analysoimaan, päätettiin tutkia eri liikkeisiin kuluvaa aikaa. Simulointiohjelmista eri liikkeisiin kuluvat ajat sai katsottua taulukkomuodossa, mutta jotta robotilta saatiin kuhunkin vaiheeseen kuluva aika, piti suoritus tallentaa videona ja toistaa kuva kerrallaan.

5.3.1 Robotin videointi

Aluksi robotin kuvaamiseksi otettiin käyttöön JAMK:lla ollut suurnopeuskamera. Suurnopeuskameralla videoitiin ensin robotin käsiohjaimen näyttöä, josta ilmeni mitä liikettä robotti on kulloinkin suorittamassa. Käsiohjaimen näytön päivitysnopeus tuli kuitenkin hyvin nopeasti vastaan, kun robotin nopeutta kasvatettiin ja näyttö saattoikin hypätä useita rivejä kerrallaan. Pian huomattiinkin, että tavallisen kännykkäkameran kuvanopeus (30 fps) riitti robotin liikkeen seuraamiseen. Robotin liikkeestä analysoitiin kuva kerrallaan mitä vaihetta se suoritti milläkin hetkellä ja yhden vaiheen suorittamisaika saatiin kuvien lukumäärästä laskemalla. Esimerkiksi kun videosta nähdään, että robotilta kuluu 6 kuvan ajan siirtää työkalua ensimmäisen pisteen päältä toisen pisteen päälle, tällöin tähän vaiheeseen kuluva aika on 6 * 0,0333..s = 0,2 s.

KUVIO 23 SALATTU

Kuhunkin vaiheeseen kuluvasta ajasta otettiin keskiarvot simulointiohjelmilla sekä oikealla robotilla. Aikoja vertailtaessa huomataan, että Roboguide osasi simuloida työkalun tuotteen pinnalle laskemiseen FINE-paikoituksella vaatiman ajan, mutta kummatkin simulointiohjelmat yliarvioivat robotin nopeuden CNT-paikoitusta käyttävissä välipisteissä. Jotta robottiin saatiin lisää nopeutta, välipisteiden nopeudeksi kokeiltiin ottaa lopulta täysi 2000 mm/s (jota nopeutta robotti ei toki ehdi näin lyhyillä saavuttaa), pinnalle laskeutumisen nopeudeksi 1000 mm/s ja liikkeisiin lisättiin ACC-lisäkäsky, jolla robotin kiihtyvyys- ja hidastavuusarvoja korotettiin. Robotin liikenopeus alkoi ihmissilmään näyttämään jo hyvin vauhdikkaalle, mutta simulointiohjelmistojen arvioimasta ajasta jäätiin vielä huomattavasti. Delmian suorittama simulointi nopeutui vain vähän liikekäskyn nopeutta lisättäessä, oikeaan robottiin vauhtia tuli huomattavasti lisää ja Roboguideen vielä tätäkin enemmän. Simulointiohjelmistot arvioivat näillä arvoilla työvaiheeseen käytettävän ajan noin kolmanneksen nopeammaksi kuin mihin oikea teollisuusrobotti näytti pystyvän. Maksiminopeudella ja -kiihtyvyydellä myös JAMK:n robotti pystyi suorittamaan työvaiheen liikesarjat asiakkaan nykyistä laitteistoa nopeammin. ACC-lisäkäskyä ei ole kuitenkaan robotin keston kannalta järkevää käyttää jatkuvasti, sillä sen käyttö aiheuttaa suuremman lämpökuorman toimilaitteille, sekä suuremman robotin mekaanisen kulumisen. Tavoiteltuun nopeuteen päästiin siis myös oikealla robotilla, mutta vielä haluttiin selvittää miksi todellisen maailman ja simuloinnin välille muodostui merkittäviä eroja.

5.3.2 Syy robotin hitauteen

Koska erot simuloidussa nopeudessa olivat huomattavat verrattuna oikealla robotilla kokeiltuun nopeuteen (Roboguiden tarkkuuden pitäisi olla n. 95 %), asiaa päätettiin kysyä Fanucin maahantuoja Fastemsilta. He pyysivät lähettämään robotin backup-tiedoston heille, jotta he voisivat tutkia onko siinä jokin asetus väärin. Backup-tiedosto sisälsi esim. kaikki robottiin tallennetut ohjelmat, asetukset ja koordinaatistot. Backup-tiedosto robotista voidaan siirtää tietokoneelle kahdella tavalla: joko tallentamalla PCMCIA-muistikortille tai siirtämällä se ethernet-liitynnällä. Backup-tiedosto

lähetettiin Fastemsille, mutta sitä ei saanut avattua. He lähettivät tiedoston myös Japaniin Fanucille ja sieltä kerrottiin JAMK:n robotin ohjelmistoversion olevan niin vanha, ettei tiedostoa saa enää auki. Fastems oli kuitenkin tutkinut backupin mukana lähetetyn robottiohjelman, jonka he olivat testanneet sekä Roboguidella että heillä olleella samanlaisella M16iB/20-robotilla ajamalla. Ajaksi he olivat ohjelmalle saaneet sekä robotilla että simuloimalla hieman hitaamman ajan kuin JAMK:lla olleet Delmia ja Roboguide, mutta kuitenkin huomattavasti JAMK:n robottia nopeamman. Oli siis hyvin aiheellista epäillä, että simulointiohjelmien ylioptimistisuuden sijaan ero simuloinnin ja oikean robotin välillä saattaisikin johtua JAMK:n robotin jonkun systeemimuuttujan väärästä arvosta tai vanhasta ohjelmistoversiosta. Palaverissa päätettiin JAMK:n robotille ottaa ohjelmistopäivitys ja samalla robotille saataisiin päivitettyä myös optio, jolla robotille voitaisiin liittää konenäkökamera.

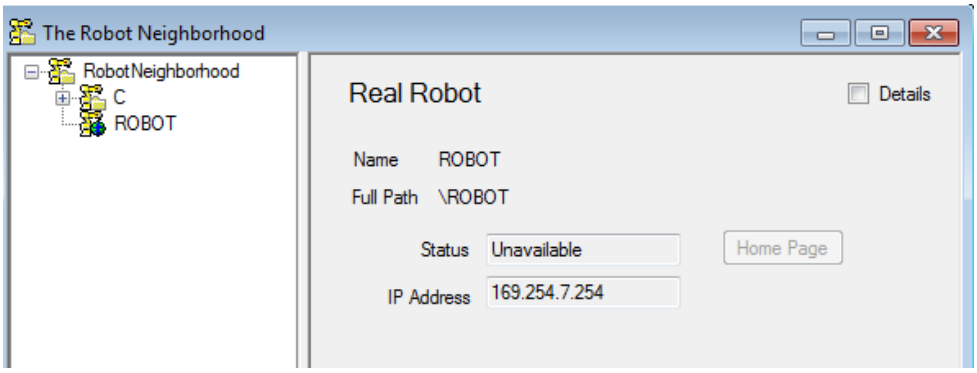
5.4 JAMK:n robotin päivittäminen

Päivitystä odotellessa Fanucin maahantuonti ehti vaihtua Fastemsilta MT-Centerille. Robottiohjelmiston päivityksen yhteydessä oli päätetty uusia myös JAMK:n Roboguide-lisenssit. Roboguide-ohjelmalla oli aikaisemmin onnistuttu siirtämään yksittäisiä robottiohjelmia tietokoneen ja robotin välillä, mutta esimerkiksi robotin backup-tiedostoa ei ollut saatu avattua tietokoneella ilmeisesti juurikin robotin vanhasta ohjelmistosta johtuen. Niinpä päivityksen myötä toivottiin myös, että backup-tiedosto saataisiin avattua Roboguideen, jolloin sen käyttö robotin ohjelmoinnin tukena tulisi helpommaksi. Backup-tiedostosta saataisiin Roboguideen käyttöön esim. robotissa olevat työkalu- ja käyttäjäkoordinaatistot, jolloin ohjelmia robotille voitaisiin tehdä luokassa etäohjelmointina ja ne toimisivat robotilla ilman muokkaamista.

5.4.1 Robot neighborhood

Robottiin tehdyn päivityksen myötä sen malli saatiin nyt avattua Roboguideen. Robotti lisättiin Roboguidelle näkyväksi Roboguiden versiosta riippuen joko Robot Neighborhood- tai Add Robot-ohjelmalla. Sen jälkeen Roboguidesta voitiin lähettää

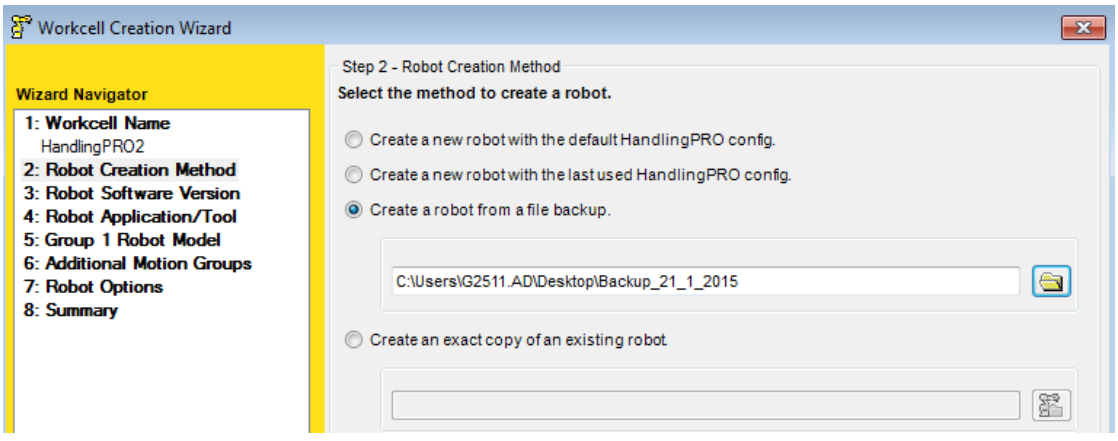
ohjelmia robotille ja hakea robottiohjelmia robotin muistista Roboguideen. Simulaattorin avulla oli myös mahdollista liittää robotti Roboguideen siten, että kun robotilla ajoi ohjelmaa, Roboguidessa ollut virtuaalinen robotti suoritti näytöllä samat liikkeet.



KUVIO 24 Robot neighborhood

5.4.2 Uusi malli

Robotin ohjelmistopäivityksen yhteydessä muistitikulle oli tallennettu backup-tiedosto robotista. Roboguideen luotiin uusi robottisolu, johon robotti tuotiin JAMK:n robotin backup-tiedostosta. Robotin malli olisi myös voitu hakea ethernet-liitynnän kautta suoraan robotilta. Roboguide tiesi mitä ohjelmisto-optioita oli JAMK:n robottiin valittu, joten se osasi valita ne käytettäväksi myös simulointimallissa.



KUVIO 25 Robottisolun luonti backupista

Virtuaaliseen robottiohjaimeen tulivat nyt kaikki JAMK:n robotissa olevat tiedot, esimerkiksi ohjelmat, koordinaatistot, rekisterit ja kuorma-asetukset. Roboguidessa voitiin nyt avata ja suorittaa myös aikaisemmin robotille tehtyjä ohjelmia virtuaalisesti.

5.4.3 SALATTU

5.5 Toinen verifiointitesti

Robotin päivityksen jälkeen vaihdettiin myös testattava ohjelma toisen tuotteen mukaiseksi. Nyt työkalukoordinaatistoksi valittiin JAMK:n robotin työkalun sijaan oikean työvaiheessa käytettävän työkalun koordinaatisto. Ajat päätettiin ottaa ylös muutamalla eri välipisteen liikenopeuden arvolla sekä Delmiasta, JAMK:n M16iB/20-robotista, Roboguideen JAMK:n robotista avatusta backupista sekä Roboguiden tyhjästä robotista. Näiden Roboguiden kahden M16iB/20-robotin mallin erotuksesta saataisiin selville, jos JAMK:n robotissa on esimerkiksi joku eriävä järjestelmämuuttujan arvo hidastamassa suoritusta. Lisäksi nyt ohjelmaa kokeiltaisiin kahdella eri Roboguiden softaversiolla, jotta nähdään onko aiemmassa 7 versiossa eroa uudempaan 8 versioon.

Toisen testiajon tulokset

Toisessakin testiajossa JAMK:n robotti jäi jälleen nopeudeltaan selvästi hitaammaksi simulointiohjelmiin verrattuna. Roboguiden eri ohjelmistoversioiden välille ei syntynyt simuloidessa eroja. Eroa ei myöskään ollut backupista avatun JAMK:n robotin mallin tai Roboguidella luodun tyhjän robottimallin välillä (johon oli asetettu kuorma-asetukset ja koordinaatistot samaksi kuin JAMK:n robotissa). Välipisteiden nopeutta lisätessä huomattiin jälleen Delmian simuloinnin saavuttavan maksiminopeutensa jo varsin alhaisella prosentuaalisella nopeudella (ks. Kuvio 26). Oikea robotti sekä Roboguide-ohjelmisto nopeutuivat vielä 1500 mm/s nopeuteen asti, jonka jälkeen nii-

den suoritusaika alkoi kasvaa. Tästä voidaan päätellä, että maksimaaliseen nopeuteen päästäkseen robotin olisi hyvä ennakkoon tietää kuinka suureen nopeuteen se ehtii kunkin liikkeen aikana kiihtyä. Ehkä ohjelmallisesti välipisteiden välisestä arvioidusta etäisyydestä voitaisiin laskea jokaiselle siirtymälle optiminopeus, jolla lyhyempien siirtymien nopeuskäsky olisi pienempi ja pidempien siirtymien suurempi.

KUVIO 26 SALATTU

Ohjelmassa kokeiltiin vaihtaa myös välipisteiden lineaariliikkeet nivelliikkeiksi. Delmiassa eroa ei ollut liiketyyppien välillä, mutta Roboguidessa ja oikealla robotilla suoritus hidastui merkittävästi siirryttäessä lineaariliikkeistä nivelliikkeisiin.

Lopuksi vielä varmistettiin robotin sisäisen ajastimella otettujen aikojen paikkansapitävyys videoimalla suoritus. Videoidusta suorituksesta katsottiin suoritusaika joka osoittautui hieman nopeampi kuin robotin sisäisen ajastimen antama suoritusaika. Lopulliseksi robotilla saavutetuksi ajaksi paikallaan olevaan osien asennukseen saatiin huomattavasti nykyistä laitteistoa nopeampi aika. Voidaan olettaa, että tuotteen liikkuessa, osat saataisiin siihen asennettua samassa ajassa, ehkä jopa hieman nopeammin, koska myös tuotteen liikkuessa, robotin siirtymät lyhenevät hieman. Tavoiteltuun aikaan päästiin siis kohtuullisen vaivattomasti, mutta suuri ero simulointiohjelmistojen ja oikean robotin ajassa jäi vielä askarruttamaan, sillä erityisesti Roboguiden simuloinnin kuuluisi olla tätä tarkempi. Syytä tähän eroavaisuuteen ei kuitenkaan löydetty.

5.6 Uudempien robottityyppien testaaminen

Uudempien robottityyppien (M20iA ja M10iA/12S) kokeilemista varten näille roboteille luotiin omat työsolunsa Roboguideen ja niiden virtuaaliseen robottiohjaimeen

ladattiin sama ohjelma kuin M16iB/20-robotilla. Uudemmat robotit osoittautuivat simulaatiossa hieman M16iB/20-robottia nopeammaksi, mutta tosin vasta kun niiden liikenopeuden arvot kasvatti suuremmaksi kuin M16iB/20-robotin maksimin 2000 mm/s. Uudempien nivelvarsirobotteihin voitiin asettaa lineaariliikkeen nopeudeksi jopa 4000 mm/s ja rinnakkaisrakenteisiin robotteihin jopa 8000 mm/s. Robottityypiksi työtehtävään sopivin olisi luultavasti M20iA-robotti tai sen versio M20iA/20M, jonka kolme viimeistä akselia ovat hieman perusmallia nopeammat. Sen hyötykuorma sekä ulottuma ovat suuremmat kuin M10-sarjan roboteilla, mutta vastaavasti sen nivelten maksimi nopeudet (° / s) hitaammat. Myös rinnakkaisrakenteisen robotin käyttäminen työvaiheessa voisi olla aiheellista selvittää, sillä niillä päästään todella suuriin nopeuksiin. Työalue rinnakkaisrakenteisella robotilla on kuitenkin varsin pieni.

6 Tulokset

Opinnäytetyöni tavoitteena oli selvittää simulointimallin avulla, voidaanko työvaihe suorittaa teollisuusrobotilla. Työn tuloksena saatiin varmistus asialle, mutta toisaalta ilmaan jäi myös avoimia kysymyksiä asian tiimoilta, jotka voitaisiin varmistaa vasta oikealla robottisolulla. Tärkein asia mitä ei tässä työssä voitu vielä selvittää, oli työn laatu teollisuusrobottia käyttäessä. Osien asennuksen laatu tulisi lopulta ratkaisemaan, onko työvaihe kannattavaa robotisoida.

6.1 Tulosten tarkastelu

Delmia-ohjelmistoon tehdystä simulointimallista saatiin selville valmistettavuuden lisäksi arvioitu aika mikä työvaiheen suoritukseen kuluu. Suoritusnopeuden varmistamiseksi tehtiin erillinen ohjelma, jolla vertailtiin simulointiohjelmistojen ja oikean robotin nopeuden eroa. Ero osoittautui yllättävän suureksi, mutta myös robotilla suoritettuna ohjelma oli riittävän nopea. Lisäksi voidaan olettaa uudempien robottien sel-

viytyvän työstä hieman nopeammin. Projektin loppupäätelmä on että työvaihe voidaan suorittaa teollisuusrobotilla nykyistä järjestelmää nopeammin ja luultavasti paremmalla laadulla.

6.2 Jatkokehitys

Seuraava vaihe projektissa olisi ollut rakentaa robottisolu, jolla työvaihetta olisi kokeiltu oikealla teollisuusrobotilla. Robottisoluksi oli valmiina kaksi suunnitelmaa. Yksinkertaisemmassa suunnitelmassa nykyiseen JAMK:n Fanuc robottiin olisi kiinnitetty asiakasyritykseltä saatava työkalu ja kokeiltu tehdä osa työvaiheesta. Tässä suunnitelmassa ei ollut mukana tuotteen liikettä, vaan asennuskokeilu olisi suoritettu paikallaan olevaan tuotteeseen. Toinen suunnitelma sisälsi myös tuotteen liikuttamisen asennuksen aikana. Tuotteen liikutteluun oli kaksi mahdollista vaihtoehtoa. Ensimmäisessä vaihtoehdossa Vision Systems olisi rakentanut liikuttelulaitteiston tuotteelle, joka olisi varustettu pulssianturilla. Pulssianturilta saataisiin robotille tieto tuotteen paikasta. Tämä vaihtoehto olisi vaatinut robottiin ’Line Tracking’-option, ja Fanucin maahantuojan mukaan se olisi melko epätarkka ratkaisu. Toinen vaihtoehto kappaleen liikutteluun oli Fanucin oma ulkoinen akseli, jota robottiohjain olisi ohjannut. Fanucin oman ulkoisen akselin käyttäminen oli kuitenkin hyvin kallis ratkaisu. Tässä vaiheessa projektin jatkokehitys menetettiin kuitenkin Fanucin maahantuojalle, joka suorittikin demolaitteiston rakentamisen asiakasyritykselle.

7 Pohdinta

Sain tehdä projektia omaan tahtiini, mutta kuitenkin tiettyjä sovittuja palaveripäiviä määriä silmällä pitäen. Tämä vapaus olikin itselleni mieluisaa, mutta voi olla että työ olisi valmistunut nopeammin virastotyöaikoja noudattaen. Projekti saattoi pysähtyä pariksi päiväksi jostakin ongelmasta johtuen, jolloin joko odoteltiin vastausta miten ongelma ratkaistaan tai etsittiin tietoa kirjallisuudesta, internetistä tai käyttöohjekirjasta. Projektia olikin erittäin mieluisa tehdä ja tukea oli hyvin tarjolla sekä Vision Systemsiltä että JAMK:n puolelta. Erityisesti JAMK:n ohjaava opettaja Seppo Rantapuska

auttoi monissa mieltä askarruttavissa kysymyksissä, joihin yhdessä etsimme ratkaisua tai olimme yhteydessä tahoon, joka tiesi kustakin asiasta. Myös asiakasyrityksestä vastattiin kiitettävästi työhön liittyviin kysymyksiin. Laajuudeltaan varsinainen konkreettisen työn osuus jäi kuitenkin ehkä hieman lyhyeksi, johtuen siitä ettei varsinaista testisolua päästy rakentamaan.

Projektin aikana opin valtavasti teollisuusroboteista. Teollisuusrobottien lisäksi opin lisää 3D-mallintamisesta sekä käyttämään Catia-, Delmia- ja Roboguide-ohjelmia. Projektissa robotille tekemäni ohjelmat olivat melko yksinkertaisia ja lineaarisia, joten robottien ohjelmointiin kiinnostaisi perehtyä tarkemmin. Robottisolujen suunnitteluun yleisesti liittyviä turvalaite- ja layout-suunnittelua ei tässä vaiheessa projektia tehty, koska työssä oli tarkoitus tehdä vain valmistettavuuden arviointi. Uskon asiakasyrityksen saaneen työstä tärkeää tietoa työvaiheen mahdollista robotisointia varten.

Lähteet

Asp, R, Heinonkoski, R, Hyppönen, H. 2008. Automaatio – helppoa elämää ?. Helsinki: Opetushallitus.

Calin, D. 2013. Robotics simulation softwares with 3D-modeling and programming support. Introrobotics 27.1.2013. Viitattu 8.1.2015. http://www.intorobotics.com/robotics-simulation-softwares-with-3d-modeling-and-programming-support/

Craig, J. 2005. Introduction to robotics, Mechanics and control. USA: Pearson Education, Inc.

Delmia Documentation. 2010. Delmian sähköinen käyttöohjekirja, Version 5 Release 20. Dassault Systemes.

Digitaalinen valmistus. N.d. Delmia ohjelmiston jälleenmyyjän esittely sivu. Viitattu 8.1.2015. http://www.delfoi.com/web/products/delmia_products/fi_FI/Delmia_ohjelmistot/

FANUC Robot series R-J3i Model B Operators Manual. N.d. Fanuc-robotin käyttöohjekirja. UK: Fanuc Robotics.

FANUC robotti R-J3 Ohjaus. N.d. Fanuc robotin käsittelytyökalun käyttöohje. Fanuc LTD.

Huttunen, J. 2004. Robotiikan historia. Helsingin yliopiston Tietojenkäsittelyn historia – seminaarin esitelmä 26.3.2004. Viitattu 12.1.2015. http://www.cs.helsinki.fi/u/kerola/tkhist/k2004/alustukset/robotiikka/roboalus.pdf

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

Kuivanen, R. 1999. Robotiikka. Tampere: Talentum Oyj/Metallitekniikka.

Laitinen, L, Ylä-Outinen, O & Tahvanainen, A. 2014 Robotiikka ja automaatio. Viitattu 14.1.2015. http://tuotanto2014.sakarikoivunen.fi/2014/11/06/robotiikka-ja-automaatio/

Lehtinen, H. N.d. Robotit. Suomen automaatioseura ry:n verkkodokumentti robotiikasta. Viitattu 12.1.2015. www.automaatioseura.fi/index/tiedostot/Robotit.doc

Most Advanced Robotics Simulation Software Overview. 2014. Artikkeli robotiikan simulointiohjelmista. Smashing Robotics 25.9.2014. Viitattu 8.1.2015. http://www.smashingrobotics.com/most-advanced-and-used-robotics-simulation-software/

Niku, S. 2001. Introduction to Robotics Analytics, Systems, Applications. USA: Prentice Hall.

Pulssinanturien teoriaa. N.d. Oem:in pulssiantureiden esittely sivusto. Viitattu 14.1.2015. http://www.oem.fi/Tuotteet/Anturi/Pulssianturit/Yleista/Pulssiantu-rien_teoriaa/825723-526144.html

Rahkola, K. 2015. Vision Systems. Sähköpostiviesti 25.2.2015. Vastaanottaja T. Ahonen. Vision Systemsin esittely.

RJ3 Programming Manual. N.d. Fanuc-robotin ohjelmointimanuaali. UK: Fanuc Robotics.

Roboguide. N.d. Fanucinn esittelysivu Roboguide ohjelmasta. Viitattu 15.1.2015. http://www.fanucrobotics.de/en/products/software/simulation%20and%20development/roboguide

Robotiikka. 2008. Lahden ammattikorkeakoulun verkkodokumentti robotiikasta. Viitattu 16.1.2015. http://miniweb.lpt.fi/automaatio/opetus/luennot/pdf_tiedostot/Robotiikka_yleinen.pdf

Suomen teollisuusrobottitilasto 2012. 2013. Suomen robottiyhdistys ry:n tilasto teollisuusroboteista. Viitattu 12.1.2015. http://www.roboyhd.fi/index.php?option=com_docman&task=doc_download&gid=168&Itemid=66

Teollisuusrobotti. 2014. Wikipedia-artikkeli teollisuusroboteista. Viitattu 12.1.2015. http://fi.wikipedia.org/wiki/Teollisuusrobotti