Juho-Pekka Hietamäki

Jyrsintärobotin etäohjelmointi

Opinnäytetyö
Kevät 2017
SeAMK Tekniikka
Automaatiotekniikan koulutusohjelma

SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Tutkinto-ohjelma: Automaatiotekniikan koulutusohjelma

Suuntautumisvaihtoehto: Sähköautomaatio

Tekijä: Juho-Pekka Hietamäki

Työn nimi: Jyrsintärobotin etäohjelmointi

Ohjaaja: Jarkko Pakkanen

Vuosi: 2017 Sivumäärä: 47

Työn ensisijaisena tavoitteena oli kasvattaa kohdeyrityksen robotin käyttöastetta etäohjelmoinnin avulla. Tarkoituksena oli tehdä uusia robottiohjelmia, joilla pystyttäisiin leikkaamaan yrityksen uudet maskimallit. Maskien leikkaaminen robotilla poistaisi käsikäyttöisen jyrsimen tarpeen. Tätä varten työssä tutkittiin, onko uusien leikkausratojen työstäminen mahdollista solun tämän hetkisellä layoutilla.

Työn välillisenä tavoitteena on leikkauslaadun parantaminen. Leikkauslaatuun vaikuttavat tekijät olivat kohdeyrityksellä ennestään tiedossa ja niihin yritettiin etsiä parempia ratkaisuja etäohjelmointia apuna käyttäen.

Toimeksiantajana työssä toimi Kihniöllä sijaitseva maalivahdinmaskeja valmistava Wall Maskit Oy. Työ tehtiin yrityksen omalle FANUC-merkkiselle teollisuusrobotille, joka leikkaa maskeihin tarvittavat materiaalin poistot.

Avainsanat: etäohjelmointi, FANUC, robotti, ohjelmointi

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Automation Engineering

Specialisation: Electric Automation

Author: Juho-Pekka Hietamäki

Title of thesis: Robot Offline Programming

Supervisor: Jarkko Pakkanen

Year: 2017 Number of pages: 47

The main purpose of this thesis was to increase the use of the target company's industrial robot by offline programming. The company needed new programs for the robot to cut holes and shapes to their new ice hockey mask models. These programs would automate a couple work tasks that are now done by hand. The current cell layout had to be investigated in order to be able to run new programs by a robot.

The other goal of this thesis was to improve the quality of cutting. The company already knew the things that have an effect on cutting quality. Cutting quality was increased by finding better cutting solutions via offline programming.

This thesis was made in co-operation with a company called Wall Maskit. This target company locates in Kihniö. They have their own industrial robot which was used for this thesis.

Keywords: offline programming, FANUC, robot, programming

SISÄLTÖ

Kuvaluettelo

Käytetyt termit ja lyhenteet

ASCII	Tietokoneiden merkistö, nimi on lyhenne englannin kielen sanoista American Standard Code for Information Interchange
CF-kortti	Vuonna 1994 esitelty muistikortti, nimi tulee englannin kielen sanoista Compact Flash
CNT-liike	Robotin ohjelman liikekomennon tarkkuusasetus, käytetään KAREL-ohjelmointikielessä
FINE-liike	Robotin ohjelman liikekomennon tarkkuusasetus, käytetään mm. KAREL- ja RAPID-ohjelmointikielissä
Joggaus	Robotin liikuttaminen online-tilassa esim. Teach pendantin avulla
KAREL	FANUC-robottien käyttämä ohjelmointikieli
Kinematiikka	Robotin nivelet ja niiden kytkökset toisiinsa sekä nivelten liikesuunnat
KRL	KUKA-robottien käyttämä ohjelmointikieli, lyhenne sanoista Kuka Robot Language
Layout	Eri laitteiden ja koneiden fyysinen sijainti tehtaan lattialla
Offline-ohjelmointi	Etäohjelmointi
Offset	Robotin työkalupisteen siirtäminen määrättyyn suuntaan, siirron suunta perustuu työskentely-ympäristön eli maailman koordinaatistoon
OLP	Lyhenne englannin kielen sanoista offline programming eli etäohjelmointi
Online-ohjelmointi	Manuaalinen ohjelmointi robotin teach pendantin avulla

PC	Tietokone, nimi on lyhenne englannin kielen sanoista Personal Computer
PCMCIA	Korttilaajennuspaikka, nimi on lyhenne englannin kielen sanoista Personal Computer Memory Card International Association
RAPID	ABB-robottien käyttämä ohjelmointikieli
Sekvenssi	Robottiin ohjelmoidun ohjelman työkierto
Sensori	Robottiin kytketty tunnistin, joka lähettää havaitsemiaan tietoja robotille
Singulariteetti	Kinemaattisessa singulariteetissa robotin nivelet ovat asennossa, jossa robotti ei enää pysty liikkumaan eteenpäin
Teach pendant	Robotin kontrollerin kannettava ohjelmointilaite
Touch up	Teach pendantin komento, jonka avulla koodissa valittuna oleva paikkatieto päivitetään robotin nykyisten nivelien arvoilla
Työkalun offset	Robotin työkalupisteen siirtäminen määrättyyn suuntaan, siirron suunta perustuu työkalupisteen koordinaatistoon
Työkalupiste	Robottiin kiinnitetyn työkalun pään koordinaatit
Työkierto	Robotin suorittamat liikkeet ja toiminnot robottiohjelman alusta loppuun kertaalleen

1 JOHDANTO

1.1 Työn tausta

Työn toimeksiantaja on kihniöläinen jääkiekko- ja salibandymaskien valmistaja Wall Maskit. Yrityksen tuotteet valmistetaan yrityksen toimitiloissa Kihniöllä, ja tarvittavat raaka-aineet ostetaan suomalaisilta toimijoilta.

Tilaustöinä valmistettavien maskien tekeminen vie aikaa, sillä suurin osa maskin valmistuksen työvaiheista tehdään käsityönä. Ainoastaan maskiin leikattavat aukot jyrsitään robotin avulla. Yrityksellä on tällä hetkellä käytössään FANUC-merkkinen kuusinivelinen teollisuusrobotti, joka leikkaa maskeihin ilmanottoaukot sekä kiinnitysreiät muiden osien kiinnittämistä varten. Robotin käyttöaste ei ole kovin suuri, koska se pystyy tekemään vain yhtä työvaihetta. Yritys haluaa kasvattaa robotin käyttöastetta sekä hyödyllisyyttä.

Robotille siirrettävä työvaihe on maskin reunojen kaventaminen uusinta mallia varten. Maskin kavennus tarkoittaa maskin leuan kaventamista sekä maskin takaosan lyhentämistä. Tällä hetkellä robotissa ei ole ohjelmaa leuan eikä takaosan leikkaamiselle, joten näiden leikkaaminen vaatii työstämistä käsikäyttöisellä jyrsimellä. Maskin kaventaminen käsityönä on vaarallista ja hidasta, mikä johtuu maskin kovasta materiaalista. Tämän työvaiheen siirtäminen robotille helpottaisi ja nopeuttaisi tuotantoa.

Robottisolu koostuu tällä hetkellä robotista sekä pneumaattisesta kiinnityspöydästä, johon työstettävä maski kiinnitetään paineilman avulla. Robotin ja kiinnityspöydän sijoittelu solun sisällä on tehty ennen uusia maskimalleja, joten tavoitteena on tutkia, pystyykö robotti leikkaamaan kaikki tarvittavat työstöt vai joudutaanko solun layoutia muuttamaan.

1.2 Työn tavoite

Työn ensisijaisena tavoitteena on kasvattaa robotin käyttöastetta siirtämällä maalivahdin maskin leuka- ja takaosan leikkaaminen robotille. Tätä varten täytyy tutkia, onko uusien leikkausratojen työstäminen mahdollista solun tämän hetkisellä layoutilla.

Uudet leikkausohjelmat robotille tehdään robottien etäohjelmointiin tarkoitetun ohjelman avulla. Ohjelman avulla pystytään tekemään myös robotin ulottuvuustarkastelua, jonka perusteella solun layout voidaan suunnitella tarpeen vaatiessa uudelleen.

Työn välillisenä tavoitteena on jyrsintälaadun parantaminen. Maskin kovasta materiaalista johtuen robotin leikkausterän jälki ei välttämättä ole aina yhtä tasaista. Jyrsintälaadun parantamista varten nykyisiin leikkausohjelmiin tehdään muutoksia. Erityisesti tarkastellaan leikkauksen etenemisen suuntaa sekä leikkauksen ensimmäisten pisteiden aloituskohtia.

1.3 Työn rakenne

Ensimmäisessä luvussa on johdanto, jossa käydään läpi työn tausta, tavoitteet sekä kerrotaan työn toimeksiantajasta. Toisessa luvussa käsitellään robotin ja tuotannon teoriaa teollisuuden näkökulmasta. Kolmannessa luvussa perehdytään robotin ohjelmointitapoihin sekä selvennetään niiden eroavaisuuksia. Neljännessä luvussa käydään läpi työn eri vaiheet suunnittelusta toteutukseen. Viidennessä luvussa tarkastellaan tutkimuksen tuloksia ja pohditaan opinnäytetyön onnistumista sekä prosessin aikana tehtyjä ratkaisuja. Kuudes luku sisältää yhteenvedon koko työstä.

1.4 Yritysesittely

Wall Maskit on Euroopan johtava jääkiekkomaskien valmistaja. Yrityksen muita tuotteita ovat salibandy- ja maahockeymaskit. Toimenkuvaan kuuluu myös vanhojen

maskien huolto ja korjaus sekä varaosamyynti. Wall Maskit valmistaa maskit tilausohjatusti asiakkaan toiveiden mukaisesti. Maskeja valmistetaan vuosittain noin 2000 kappaletta. Yritys työllistää 4–5 henkilöä ja sen vuosittainen liikevaihto on noin puoli miljoonaa euroa. Suurin osa liikevaihdosta tulee ulkomaanviennistä. Yrityksen tärkeimpiä markkina-alueita ovat Pohjois-Amerikka ja Eurooppa. (Karvinen 2017).

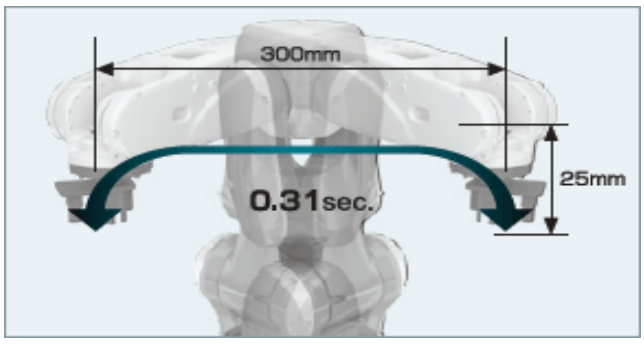
2 ROBOTIT TEOLLISUUDESSA

Robotit ovat tärkeässä asemassa teollisuudessa. Robotit pystyvät suoriutumaan tietyistä työtehtävistä ihmistä tehokkaammin, eivätkä ne tarvitse mitään mukavuuksia kuten ihmiset. Robotin hankinta vaatii kuitenkin paljon aikaa ja rahaa ennen kuin robotti on toimiva osa tuotantoa. (Arora & Gupta 2009, 308.)

2.1 Robottityypit

Erilaisia robotteja on useita, on tärkeää osata valita oikea robotti oikeaan tehtävään (Gill & Krar 2003, 752). Tietyt robotit on suunniteltu tietynlaisia tehtäviä varten. Robotin kykyä suoriutua eri tehtävistä pystymään lisäämään erilaisten työkalujen avulla. Robottien niveliä liikutetaan useimmiten sähkömoottoreiden avulla, mutta välillä käytetään myös hydrauliikkaa nivelten liikuttamiseen. (Spilsbury & Spilsbury 2017, 6–8.)

Robotin nopeutta mitataan standardisoidulla syklillä, jossa robotti siirtää kappaleen toiseen paikkaan. Yksi robotin tekemä sykli tarkoittaa, että robotti poimii kappaleen, vie sen toiseen paikkaan, laskee kappaleen ja palaa takaisin alkupisteeseen. Standardisyklissä robotin edestakaisin kulkema etäisyys kappaleen poiminnassa ja pudotuksessa on yksi tuuma, kun taas siirtymän etäisyys on 12 tuumaa. (Robotpark Academy 2015.) Monet valmistajat ilmoittavat robottien nopeudet standardisyklille sekä pidennetylle syklille. ABB ilmoittaa robottiensa pidennetyn syklinsä poiminta- ja pudotusetäisyydeksi 90 millimetriä ja siirtymän etäisyydeksi 400 millimetriä. (ABB 2017e.)



Kuva 1. Robotin standardisykli
(NACHI 2016)

2.1.1 Nivelrobotti

Nivelrobotti on yleisin teollisuudessa käytettävä robotti. Näillä roboteilla on yleensä kuusi vapausastetta ja ne pystyvät liikkumaan tiettyyn pisteeseen eri nivelkonfiguraatiolla. Suurimpien nivelrobottien ulottuvuus on yli 3,5 metriä sekä nostokyky yli 1000 kilogrammaa. (Wilson 2015, 24.)

Nivelrobotin suurin etu on sen liikkumisen joustavuus. Sen nivelet voidaan helposti suojata, joten se voi työskennellä pölyisissä ja kosteissakin olosuhteissa. Heikkoutena nivelrobotilla on sen huono tarkkuus sekä ohjelmoinnin vaikeus. (Arora & Gupta 2009, 325.)

Kuva 2. ABB IRB 2400 -nivelrobotti
(ABB 2017b)

2.1.2 SCARA-robotti

SCARA-robotti on neljän vapausasteen robotti. Näiden robottien etuina ovat suuret nopeudet sekä tarkkuus, minkä vuoksi niitä käytetään monesti muun muassa pakkaus- ja kokoonpanotehtävissä. SCARA-robotin ulottuvuus sekä nostokyky on pienempi kuin useimmilla nivelrobotilla. (Wilson 2015, 25–26.) Sen lisäksi SCARA-robotti voi liikkua haluttuun pisteeseen vain kahdella eri nivelkonfiguraatiolla. Robotin online-ohjelmointi on suhteellisen helppoa, mutta etäohjelmointi on haastavampaa. (Arora & Gupta 2009, 326.) Maailmanlaajuisesti SCARA-roboteilla on noin 12 prosentin markkinaosuus teollisuudessa (Wilson 2015, 25–26).

Kuva 3. ABB IRB 910SC SCARA -robotti
(ABB 2017c)

2.1.3 Lineaarirobotti

Lineaarirobotit ovat toiseksi yleisimpiä robotteja noin 22 prosentin markkinaosuudella. Lineaarirobotteihin kuuluvat niin pienemmät lajittelurobotit kuin isommat ristikkopalkkirobotit. Useimmissa lineaariroboteissa on vain kolme vapausastetta, niitä käytetään yleensä kappaleiden käsittelyssä sekä kokoonpanotehtävissä. Isojen ristikkopalkkirobottien työskentelykorkeus voi olla useita metrejä, minkä vuoksi robotin alapuolelle jää hyvin työskentelytilaa. (Wilson 2015, 26–27.)

Lineaarirobottien etuna ovat rakenteen jäykkyys sekä ohjelmoinnin helppous. Heikkoutena taas ovat avoimet nivelet, jonne esimerkiksi pöly pääsee helposti kerääntymään. Lineaarirobotin toiminta-alue on käytännössä melkein sama kuin sen viemä fyysinen pinta-ala tehtaan lattialta. (Arora & Gupta 2009, 322.)

Kuva 4. Lineaarirobotti
(LPR Global 2017)

2.1.4 Deltarobotti

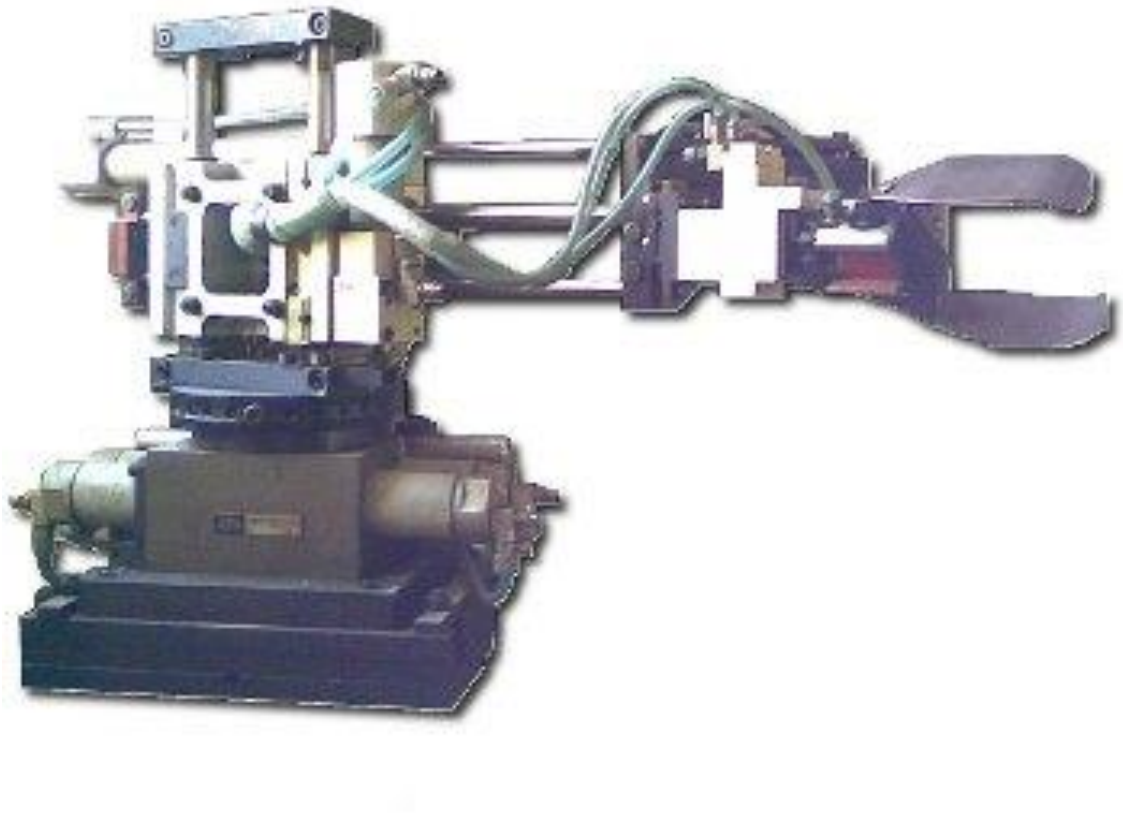
Deltarobotti on suhteellisen uusi robottityyppi. Se on suunniteltu kiinnitettäväksi esimerkiksi kattoon tai kehikkoon. Deltarobotti on SCARA-robotin tapaan hyvin nopea ja liikkeiden kiihtyvyydet ovat suuria. Tämä johtuu delta-robotin rakenteesta, jossa moottorit sekä suurin osa painosta sijaitsee alustassa. (Wilson 2015, 27–28.) Nopeimmat deltarobotit pystyvät tekemään 200 standardisykliä minuutissa (ABB 2017e). Markkinaosuus deltarobotilla on noin yhden prosentin luokkaa. Robotin tärkeimpänä käyttökohteena ovat kuljetushihnalla kulkevien tuotteiden lajittelu sekä pakkaus. (Wilson 2015, 27–28.)

Kuva 5. FANUC M-1iA/0.5A -deltarobotti
(FANUC [Viitattu 4.4.2017])

2.1.5 Sylinterirobotti

Sylinterirobottia käytetään pääasiassa elektroniikkateollisuuden kokoonpanossa. Robotin rakenne on yksinkertainen ja se koostuu lineaari- ja kiertoakseleista. Robotin markkinaosuus on noin kaksi prosenttia. (Wilson 2015, 28.)

Sylinterirobotit pystyvät vankkarakenteisuutensa takia siirtämään painaviakin kuormia työalueellaan. Sylinterirobotti pystyy ulottumaan kaikkialle itsensä ympärillä, mutta ojentuvien nivelien puutteen takia se ei pysty ulottumaan itseään pidemmälle. Rakenteen yksinkertaisuuden vuoksi robotin huoltaminen on helppoa, mutta koneisto on altis pölylle ja kosteudelle. (Arora & Gupta 2009, 323.)



Kuva 6. Sylinterirobotti
(All On Robots [Viitattu 4.4.2017])

2.1.6 Yhteistyörobotti

Yhteistyörobotti on alan uusin innovaatio. Se on kehitetty työskentelemään ihmisen rinnalla. ISO 10218 -standardin mukaan yhteistyörobotin täytyy täyttää vähintään yksi seuraavista turvallisuusominaisuuksista:

1. Valvottu turvapysäytys
2. Valvottu turvanopeus
3. Rajoitettu käyttövoima
4. Käsin viemällä -ohjelmointi. (Mogab 21.9.2016.)

Näillä neljällä turvallisuusominaisuudella on eroja. Ainoastaan käyttövoimaltaan rajoitetut robotit voivat täysin työskennellä ihmisen kanssa yhteistyössä. (Mogab 21.9.2016.)

Kuva 7. ABB IRB 14000 YuMI -robotti
(ABB 2017d)

2.2 Automatisoitu tuotanto

Teollisuuden tuotantoautomaatio voidaan jakaa kolmeen eri ryhmään:

1. Normaali tuotantolinja
2. Ohjelmoitava tuotantolinja
3. Joustava tuotantolinja. (Kuttan 2007, 4.)

Normaali tuotantolinja tarkoittaa, että linjasto on suunniteltu nimenomaan tietyn tuotteen valmistukseen. Ohjelmoitavaa tuotantolinjaa taas käytetään, kun tuotantomäärät ovat vähäisempiä, mutta valmistettavia tuotteita on enemmän. Tässä tapauksessa tuotteet valmistetaan sarjoissa tuote kerrallaan ja linjastoon on ohjelmoitu eri asetukset eri tuotteille. Joustavassa tuotantolinjassa voidaan tehdä useampaa eri tuotteita yhtä aikaa samassa linjastossa. (Kuttan 2007, 4.)

2.3 Robottien kehitys

Teollisuusrobottien kehityksestä puhuttaessa käytetään termiä sukupolvi. Tällä hetkellä teollisuudesta löytyy kolmen eri sukupolven robotteja. Näiden lisäksi voidaan luokitella myös neljäs sekä viides sukupolvi, mutta niihin luokiteltavat robotit ovat vielä fiktiivisellä tasolla. (Kuttan 2007, 6.)

Ensimmäisen sukupolven robotit eivät reagoi ympäristöön tai sen muutoksiin mitenkään. Lisäksi ne noudattavat tarkasti niihin ohjelmoitua sekvenssiä eli työkiertoaan riippumatta siitä, onko sen edessä esimerkiksi edes työstettävää kappaletta. (Kuttan 2007, 6.)

Toisen sukupolven roboteissa on erilaisia sensoreita, joiden avulla ne pystyvät ottamaan ympäristöään huomioon. Myös robotin laskentateho on parempi kuin ensimmäisen sukupolven roboteissa, jotta ne pystyvät muuttamaan työkiertoaan toimintojen välissä. Tyypillinen esimerkki tästä on kuljetuslinja, jota pitkin kulkee erilaisia kappaleita robotin käsiteltäväksi. (Kuttan 2007, 6.)

Kolmannen sukupolven robotit ovat teknologialtaan edistyneimpiä. Ne pystyvät tekemään päätöksiä sekä ratkaisemaan ongelmia. Tämä prosessointi perustuu keinoälyyn sekä robotin sensoreilta tuleviin tietoihin. (Kuttan 2007, 6.)

Robottien kehityksen seuraavat vaiheet olemassa olevien kolmen sukupolven lisäksi ovat neljäs ja viides sukupolvi. Neljännen sukupolven robotti olisi inhimillinen, eli se pystyisi fyysisesti sekä älyllisesti samaan kuin ihminen. Viides sukupolvi puolestaan tarkoittaisi yli-inhimillistä robottia, joka pystyy tekemään päätöksiä ihmistä paremmin ja nopeammin. (Kuttan 2007, 6.)

3 ROBOTIN OHJELMOINTI

3.1 Ohjelmointi

Robotin ohjelmointi tarkoittaa yksinkertaisuudessaan eri komentojen tallentamista robotin kontrollerille. Kontrolleri suorittaa valittua ohjelmaa rivi kerrallaan ylhäältä alaspäin liikuttaen robottia sen mukaan. (Arora & Gupta 2009, 369.) Robotin ohjelmointi voidaan jakaa kahteen pääryhmään, online-ohjelmointiin ja offline-ohjelmointiin (Nof 1999, 337).

3.2 Robotin liikekomennot

Robotti pystyy liikkumaan ohjelman seuraavaan tallennettuun pisteeseen kolmella eri liikekomennolla. Liikekomento tarkoittaa robotin siirtymän laskentaa sille asetetun työkalupisteen suhteen. Liikekomennot luokitellaan seuraavasti:

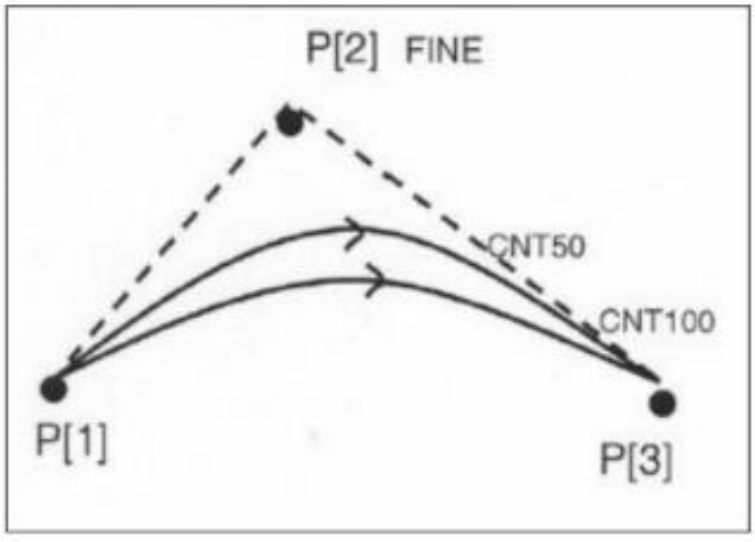
1. Nivelliike
2. Lineaariliike
3. Ympyräliike. (Nof 1999, 225.)

Liikuttaessa pisteestä pisteeseen nivelliikkeellä robotti liikuttaa niveliään laskennallisesti helpoimmalla tavalla, jotta robotin työkalupiste saadaan liikutettua haluttuun paikkaan. Robotin kontrolleri laskee työkalupisteen sijainnin suoralla kinematiikalla robotin nivelarvojen perusteella. Nivelliikkeessä robotin nivelet liikkuvat yhtä aikaa niin, että jokainen nivel liikkuu mahdollisimman lyhyen siirtymän. Nivelliike ei vaadi robotin kontrollerilta niin paljon laskentatehoa kuin lineaari- tai ympyräliike. (Nof 1999, 225.)

Lineaariliikkeessä robotti ajaa sille asetettua työkalupistettä lineaarisesti pisteestä pisteeseen halutulla nopeudella. Lineaariliikettä käytetään yleensä pienissä ja tarkoissa siirtymissä. Lineaariliikkeen aikana robotin kontrolleri laskee robotin nivelarvoja käänteisen kinematiikan avulla. (Nof 1999, 225.)

Ympyräliike on kuin lineaariliike, jossa siirtymä seuraavaan pisteeseen tapahtuu kauttakulkupisteen kautta halutulla nopeudella. Ympyräliikettä varten täytyy määrittää alkupiste, loppupiste ja kauttakulkupiste. Ympyräliikkeessä nämä kolme pistettä muodostavat ympyrän kaaren muotoisen liikeradan. Kontrolleri laskee robotin nivelarvot käänteisellä kinematiikalla. (Nof 1999, 226.)

Robotin liikekomennolle asetetaan aina erikseen liikenopeus ja pistetarkkuus. Liikenopeus tarkoittaa, kuinka nopeasti robotti liikkuu ohjelmaan tallennettuun pisteeseen. Liikekomennon pistetarkkuus määritellään eri tavoin robottimerkistä riippuen. Esimerkiksi FANUC-roboteilla pistetarkkuus annetaan joko FINE- tai CNT-liikkeen muodossa. FINE-liikkeellä robotti ajaa työkalupisteen haluttuun pisteeseen ja pysähtyy täysin, ennen siirtymistä seuraavaan pisteeseen. CNT-liikkeessä robotti ajaa haluttuun pisteeseen pysähtymättä ja jatkaa sen jälkeen seuraavaan pisteeseen. CNT-liikkeelle määritetään arvo nollasta sataan, sen perusteella, kuinka paljon robotin halutaan hidastavan vauhtiaan kyseisen pisteen kohdalla. Mitä lähempänä arvo on nollaa, sitä tarkempi robotti on. Suurella CNT-arvolla robotti on nopeampi, mutta sen tarkkuus kärsii. (Shazar & Tabactsidis 2015, 15.)



Kuva 8. FINE- ja CNT-liikkeet
(Marquina 2010, 98).

3.3 Online-ohjelmointi

Online-ohjelmoinnissa operaattori ohjaa robottia teach pendantin avulla. Robotti jogataan eli ajetaan haluttuun pisteeseen ja tämän jälkeen paikkatieto tallennetaan robotin kontrollerin muistiin. Käsiajo-tilassa robotti on turvanopeudella ja tästä syystä operaattori voi olla robotin lähellä ohjelmoidessaan sitä. (Arora & Gupta 2009, 369–370.)

Online-ohjelmoinnissa on myös huonoja puolia. Tämä ohjelmointitapa vaatii robotin fyysisen läsnäolon. Lisäksi robotin käsittely vaatii teknistä osaamista sekä robotin käyttämän ohjelmointikielen hallintaa. Ohjelmoinnin ajaksi robotti täytyy irrottaa tuotannosta, mikä voi aiheuttaa koko tuotantolinjan pysähtymisen ja siten tulla kalliiksi yritykselle. (Low 2007, 349.)

3.4 Offline-ohjelmointi

Offline-ohjelmoinnissa robottia ei tarvitse irrottaa tuotannosta ohjelmoinnin ajaksi. Tämän vuoksi offline-ohjelmoinnin tärkeys kasvaa koko ajan. Suurimmat taloudelliset säästöt tulevat siitä, että huollon jälkeen robotti voidaan ottaa nopeasti käyttöön ja tarvittaessa ohjelmoida uudelleen. (Low 2007, 107.)

3.4.1 Todellisen robotin ja robottimallin kalibrointi

Todellinen robotti ei ikinä vastaa täysin simulointiympäristön robottimallia. Tämä tarkoittaa sitä, että jokaisella robotilla on yksilölliset nivelarvot, jotka eroavat robottimallin oletusarvoista. Näiden arvojen muuttumiseen tai eroavaisuuteen voivat vaikuttaa ympäristö ja robotin normaali kuluminen. Myös robottiin kohdistuvat iskut ja kolhut voivat muuttaa sen nivelarvoja. (Low 2007, 130.)

Todellisen robotin nivelarvojen virheet korjataan etäohjelmointia varten tietokoneella, sen sijaan että muutettaisiin robotin rakennetta millään tavalla. Todellisen robotin ja robottimallin nivelarvojen erot voidaan kompensoida kalibrointiprosessilla, jossa mallirobotin nivelarvoja valituissa pisteissä verrataan

todellisen robotin nivelarvoihin samoissa pisteissä. Ilman nivelarvojen kalibrointia todellinen robotti ei pysty suorittamaan ohjelmaa halutulla tavalla, mikä näkyy tallennettujen pisteiden etäisyyserossa todellisessa ympäristössä. (Elatta 2004, 74.)

3.4.2 Offline-ohjelmoinnin mahdollisuudet ja riskit

Offline-ohjelmointiin tarkoitettuja ohjelmia on eri hintaisia. Kalliit ohjelmistot voivat olla suuri hankinta pienelle yritykselle. Ohjelmiston käyttö vaatii osittain samaa teknistä tietämystä robotista kuin online-ohjelmointi, mutta tämän lisäksi myös ohjelmiston sisäisten komentojen ymmärtämistä. (Arora & Gupta 2009, 374.)

Onnistunutta offline-ohjelmointia varten tarvitaan hyvin tarkka klooni oikeasta tuotantosolusta. Kloonia varten tarvitaan sekä robotin ja sen työkalupisteen että työstettävän kappaleen 3D-mallit. Kun kaikki edellä mainitut osat kalibroidaan toisiinsa nähden oikein, offline-ohjelmoinnin tuloksena saatu robottiohjelma toimii samoin myös oikeassa tuotantosolussa. Robotin työkalupisteen kalibrointi voidaan tehdä itse robotilla, jonka jälkeen työkalupisteen arvot voidaan syöttää ohjelman robotille. (Gan & Tang 2011, 93.) Ohjelmassa on myös mahdollista valita, millaisilla nivelkonfiguraatioilla robotti ajaa haluttuun pisteeseen. Valmista robottiohjelmaa pystytään simuloimaan tietokoneella, ja mahdolliset virheet voidaan havaita ennen ohjelman lataamista oikealle robotille. (Arora & Gupta 2009, 373.)

3.4.3 Offline-ohjelmointiin tarkoitetut ohjelmat

FANUC ROBOGUIDE on etäohjelmointiin tarkoitettu ohjelma, josta löytyy yrityksen kaikki robottimallit ja sisäänrakennettu virtuaalinen robottiohjain eli teach pendant. Ohjelmassa voidaan simuloida robottia 3D-maailmassa ja nähdä, miten se toimii eri tilanteissa. (FANUC 2017.)

Siemens Process Simulate on helppokäyttöinen simulointiin ja etäohjelmointiin tarkoitettu ohjelma, sillä voidaan tehdä robottiohjelmia useille eri robottimerkeille. Process Simulate ei sisällä minkään merkin robottien 3D-malleja eikä virtuaalista kontrolleria. Process Simulate -ohjelmassa voidaan analysoida työkierron nopeutta

ja havaita robotin törmäyskohtia. Myös kinemaattisten mallien, kuten robottien ja resurssien mallinnus onnistuu. (Siemens 2017.)

ABB RobotStudio on PC:lle tarkoitettu simulointi- ja etäohjelmointiohjelma. Ohjelman avulla voidaan ohjelmoida ABB:n robotteja ja harjoitella niillä. RobotStudio-ohjelmassa on sisäänrakennettu virtuaalinen kontrolleri, joka toimii samalla tavalla kuin oikea ABB-kontrolleri. Virtuaalisella kontrollerilla voidaan tehdä robottiohjelmia, näiden ohjelmien simuloinnin realistisuus vastaa lähes todellista tilannetta tehtaalla. (ABB 2017a.)

3.5 Ohjelmointikielet

Eri robottivalmistajat käyttävät eri ohjelmointikieliä. Ei ole olemassa yleistä ohjelmointikieltä, jota kaikki robotit ymmärtäisivät. Esimerkiksi FANUC-merkkiset robotit ymmärtävät KAREL-ohjelmointikieltä (Kandray 2010, 371), kun taas KUKA-yrityksen valmistamat robotit ymmärtävät omaa KRL-kieltään (Braumann & Brell-Cockan 2013, 301).

ABB käyttää roboteissaan prosessipainotteista RAPID-ohjelmointikieltä. RAPID-ohjelmointikieli on pitkälle kehittynyt ohjelmointikieli, joka sisältää toiminnallisia muuttujia, robotin työjärjestyksen hallintaan vaikuttavia toimintoja ja virheiden tunnistustoiminnon. (Tarn, Chen & Zhou 2007, 92.)

4 JYRSINTÄROBOTIN ETÄOHJELMOINTI

4.1 Lähtötilanne

Yrityksellä on käytössä FANUC M-6i -teollisuusrobotti, jonka kontrollerina toimii FANUC R-J3. Robotin työkaluna toimii servomoottori, johon on kiinnitetty leikkaamiseen tarkoitettu jyrsinterä. Työkalulla on oma ohjausyksikkö, joka on kytketty robottiin. Servomoottori voidaan siis kytkeä päälle sekä sammuttaa myös robotin teach pendantista ohjaamalla robotin lähtöjä.

Tällä hetkellä robotilla voidaan leikata vain vanhempia kypärämalleja, koska uusille malleille ei ole tehty robottiohjelmia kavennuksia varten. Uudemmat mallit ovat lapsille ja nuorille tarkoitettuja malleja, jotka valmistetaan aikuiselle tarkoitettua kypärämallia pienentämällä.

Ohjelmaa uudempia malleja varten on aikaisemmin yritetty tehdä käsin, mutta se on epäonnistunut, koska robotin nivelrajat ovat tulleet vastaan eikä leikkausjälki ole ollut tarpeeksi tasaista. Tästä syystä tutkitaan, onko solun layoutiin tehtävä muutoksia, jotta uudet leikkausohjelmat voidaan toteuttaa.

Robotin leikkausjälki on tällä hetkellä vain kohtalaista ja sitä halutaan parantaa. Leikatun aukon reuna on epätasainen ja rosoinen. Leikkausjäljen lopputulokseen vaikuttavia tekijöitä on monia. Tärkeimmiksi näistä ovat yrityksen kokemuksen perusteella osoittautuneet jyrsinnän syvyys sekä kitkan määrä leikkaavan terän ja työstettävän materiaalin välillä.

Työn käytännön osuudessa päätettiin ohjelmoida ensin Seinäjoen Ammattikorkeakoulun laboratoriossa sijaitseva FANUC R-2000iB -teollisuusrobotti, jonka jälkeen siirryttiin kohdeyrityksen FANUC M-6i -robotin ohjelmointiin. Tämän seurauksena työtä voitiin viedä ensin eteenpäin Seinäjoella. Lisäksi koulun robotti on tarkoitettu opetuskäyttöön tuotannon tehtävien sijaan, joten sen avulla voidaan tutustua kyseisen valmistajan robottien ominaisuuksiin ja ohjelmointikieleen ilman tuotannollisia keskeytyksiä.

Ohjelmaksi työtä varten valittiin simulointiin ja etäohjelmointiin tarkoitettu Siemens Process Simulate, koska sen käytöstä oli kertynyt kokemusta aikaisemmin ja se oli todettu toimivaksi työkaluksi. Tämän ohjelman käyttö työssä ei myöskään aiheuttaisi ylimääräisiä kustannuksia, koska ohjelma on asennettu osalle Seinäjoen Ammattikorkeakoulun tietokoneista.

4.2 Siemens Process Simulate 13.0

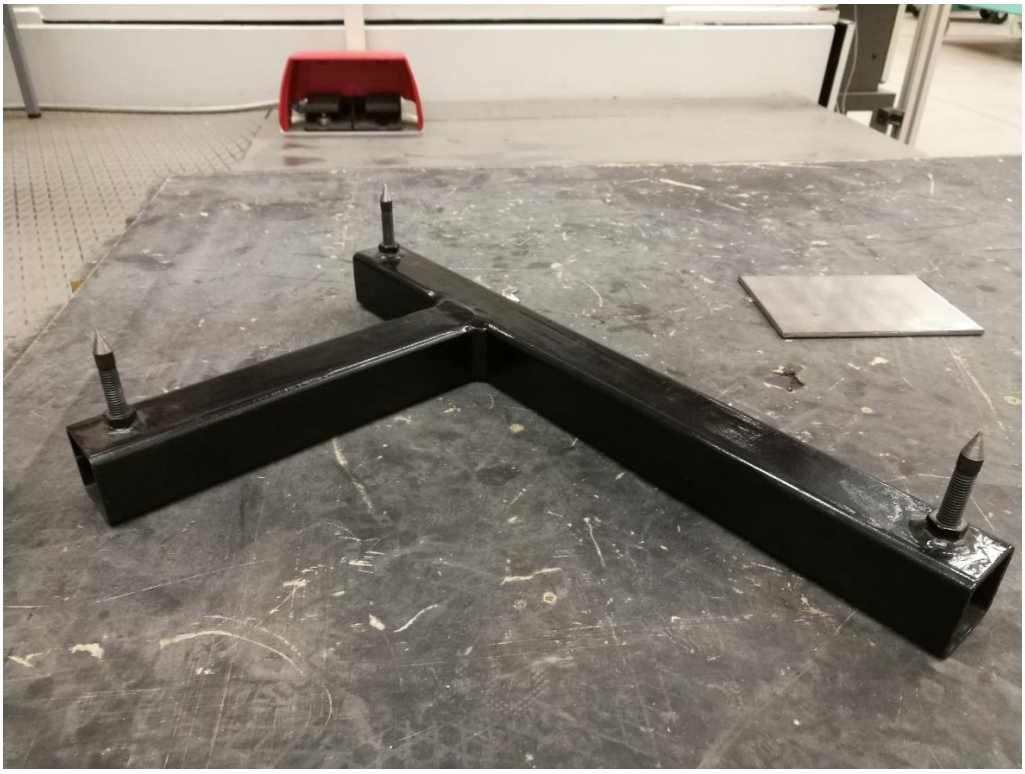
Siemens Process Simulate on tehokas suunnittelun työkalu. Se on kolmannen osapuolen ohjelmointiohjelma, jolla voidaan tehdä robottiohjelmia useille eri roboteille. Ohjelmassa voidaan tehdä joko aikaan perustuvaa simulointia tai tapahtumiin perustuvaa simulointia. (Siemens 2017.) Molemmissa simulointityyleissä on puolensa. Esimerkiksi, jos tietokone on kytketty erilliseen PLC-logiikkaan, signaaleja voidaan lähettää ohjelman ja logiikan välillä vain tapahtumiin perustuvassa simuloinnissa. Process Simulate sisältää paljon robotin ohjaukseen liittyviä toimintoja, kuten OLP-koodiosion. OLP-osioon voidaan kirjoittaa koodia suoraan robotille, kuten laittaa jokin robotin lähtö päälle.

Process Simulate -ohjelmassa ei ole valmiiksi robottien kinemaattisia malleja, joten robotit täytyy lisätä sinne itse. Esimerkiksi ABB:n sivuilta voi ladata yrityksen omien robottien 3D-malleja, mutta FANUC ei tarjoa tätä mahdollisuutta. Yrityksen sivuilta löytyi kuitenkin kaikki tekniset tiedot roboteista, kuten robotin fyysiset mitat sekä nivelten liikekulmat.

FANUC R-2000iB -robotin 3D-malli saatiin www.grabcad.com -nimiseltä sivustolta. Sivusto on tarkoitettu 3D-mallinnuksesta kiinnostuneille henkilöille, sinne voi ladata itse luomiaan 3D-malleja muiden saataville sekä ladata itselleen muiden tekemiä 3D-malleja. Robotin malli tuotiin Process Simulate -ohjelmaan ja sen omalla mittaustyökalulla varmistettiin mallin todenmukaisuus. Tämän jälkeen robotille rakennettiin kinematiikka. Process Simulate -ohjelmassa kinematiikan rakentamisessa määritellään erikseen kaikki liikkuvat nivelet ja niiden kytkökset muihin niveliin. Tämän jälkeen määritellään nivelten väliset kiinnityskohdat sekä nivelten liikesuunnat. Liikesuunnille asetetaan samat minimi- ja maksimiarvot kuin oikealla robotilla. Sen

jälkeen simulaatiossa oleva robotti ei voi ylettyä paikkaan, johon oikea robottikaan ei ylety.

Kinemaattisen mallin rakentamisen jälkeen 3D-malli täytyi kalibroida oikean robotin kanssa. Kalibrointi tapahtui robotin työkaluun kiinnitettävän kalibrointipiikin ja -työkalun avulla. Kun kalibrointiohjelma oli valmis, se siirrettiin Process Simulate -ohjelmaan. Kalibrointityökalun mittauspisteistä otettiin fyysiset mitat ja näiden perusteella tehtiin 3D-malli kalibrointityökalusta. Simulaatiossa tehtiin samat kalibrointipisteet simuloidun robotin ja kalibrointityökalun 3D-mallin avulla. Kun molemmat ohjelmat olivat valmiita, ne syötettiin piste kerrallaan Process Simulate -ohjelman kalibrointityökaluun. Kun kaikki oikeat pisteet ja simulointiympäristön pisteet olivat taulukossa, ohjelma laski simuloidulle robotille korjausarvot, jotta se vastaa kinematiikaltaan täysin koulun laboratoriossa olevaa robottia.



Kuva 9. Kalibrointityökalu

Työtä varten tarvittavasta jääkiekkomaskin mallista löytyi 3D-malli, joka saatiin käyttöön leikkausratojen piirtämistä varten. Maskimalli käännettiin eri tiedostomuotoon Process Simulate -ohjelmaa varten ja 3D-malli tuotiin ohjelmaan. Maskimalli ei valitettavasti sisältänyt kiinteää geometriaa, vaan kyseessä oli pelkkä pintamalli. Tästä

syystä Process Simulate ei tunnistanut 3D-mallin geometriaa, joten maskimallin pinnalle ei voinut suoraan tehdä leikkausratoja. Ongelmasta raportoitiin kohdeyritykselle ja pyydettiin kiinteää 3D-mallia maskista. Ongelma sivuutettiin tässä vaiheessa ja robottiohjelmien teko Process Simulate -ohjelmalla aloitettiin.

Process Simulate tarvitsee erillisen asennuspaketin, jotta se voi muuttaa ohjelmaan tallennetut pisteet sekä OLP-komennot FANUC-robottien käyttämälle KAREL-ohjelmointikielelle. Koululta saatiin FANUC-roboteille tarkoitettu asennuspaketti ja se lisättiin Process Simulate -ohjelman tiedostokansioon. Sen jälkeen tehtiin testiohjelma, jota oli tarkoitus testata koulun laboratorion robotilla. Valmis testiohjelma tallennettiin USB-muistitikulle, koska laboratorion R-30iA -kontrollerissa oli tuki USB 2.0 -liitännälle. Robotin kontrolleri ei kuitenkaan tunnistanut USB 3.0 -muistitikkua, jonka pitäisi olla yhteensopiva USB 2.0 -portin kanssa. Muistitikkua kokeiltiin alustaa eri tiedostojärjestelmille, mutta kontrolleri ei silti tunnistanut muistitikkua. Koululta löytyi kourallinen vanhoja USB 2.0 -tikkuja, joista kontrolleri lopulta onnistui tunnistamaan yhden Kingston DataTraveler -merkkisen muistitikun. Testiohjelma tallennettiin kyseiselle muistitikulle, mutta kontrolleri ei kuitenkaan pystynyt löytämään testiohjelmaa tikulta.

Ongelmaa tutkiessa saatiin selville, että FANUC R-J3 -kontrolleri ymmärtää vain .tp-päätteisiä tiedostotyyppejä, kun taas Process Simulate pystyy tuottamaan vain .ls-tiedoston. Tarvittavia .tp-tiedostotyypin ohjelmia pystytään tuottamaan FANUC-yhtiön omilla etäohjelmointiin tarkoitetuilla ohjelmistoilla. Toinen vaihtoehto ohjelmien siirtämiseen robotille olisi ostaa FANUC-yhtiöltä kontrollerille erikseen asennettava ASCII-lisäominaisuus, joka pystyy muuttamaan .ls-tiedostot kontrollerin haluamaan .tp-muotoon. Kun tämä tieto varmistui, päätettiin luopua Siemens Process Simulate -ohjelman käytöstä ja siirtyä käyttämään pelkästään FANUC-robottien etäohjelmointiin tarkoitettua ROBOGUIDE-ohjelmaa.

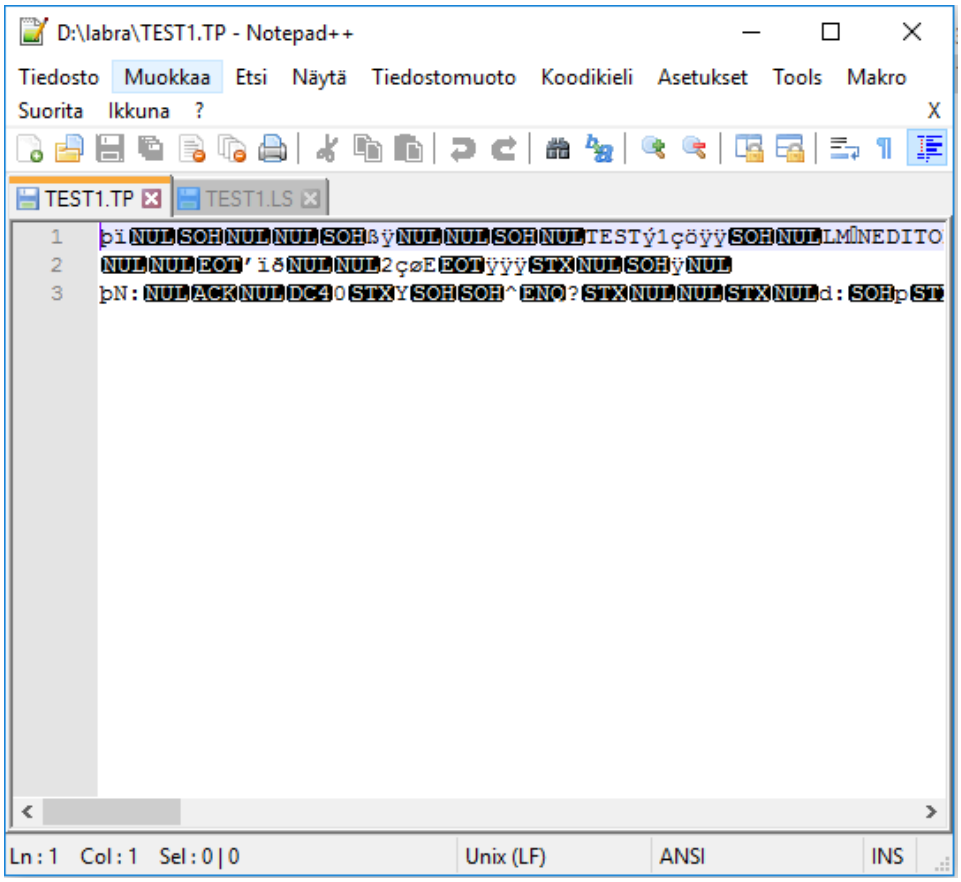
Kuva 10. FANUC R-2000iB 165F -robotin kalibrointi

4.3 FANUC ROBOGUIDE

ROBOGUIDE on FANUC-yhtiön oma ohjelma, jossa on robottisimuloinnin lisäksi myös mahdollisuus tehdä robottiohjelmia suoraan virtuaalisen teach pendantin avulla. Ohjelmasta löytyy valmiiksi kaikkien FANUC-merkkisten robottien kinemaattiset 3D-mallit, joten M-6i- ja R-2000iB-robottimallit saatiin ohjelmaan ilman erillistä mallintamista.

ROBOGUIDE-ohjelma sisältää virtuaaliset teach pendantit jokaiselle FANUC-merkkiselle kontrollerille. Robotin etäohjelmointia varten täytyy ohjelmassa valita saman mallin kontrolleri kuin oikeassa robottisolussa käytettävä kontrolleri on.

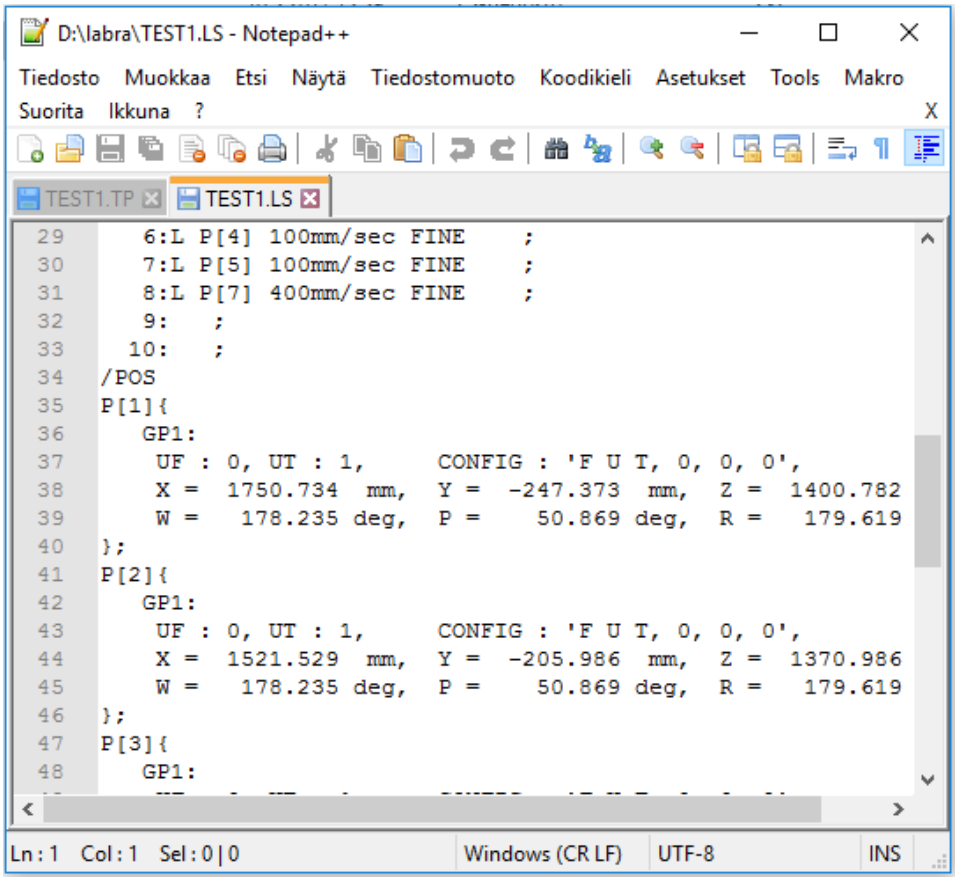
Ohjelman avulla robottiohjelma voidaan tallentaa suoraan .tp-tiedostomuotoon. Robotin kontrolleri ei siis tarvitse ASCII-lisäominaisuutta, jotta robotti ymmärtäisi koodia.



Kuva 11. Esimerkki .tp-tiedostosta

4.4 Seinäjoen Ammattikorkeakoulun FANUC R-2000iB -robotti

Työ aloitettiin luomalla ROBOGUIDE-ohjelmaan solu, jossa oli samat asetukset kuin oikeassa laboratoriosolussa. Seuraavaksi oikealla robotilla tehtiin testiohjelma, joka tallennettiin ainoalle yhteensopivaksi todistetulle USB-muistitikulle. Laboratorion R-30iA -kontrollerilla on tallennukseen kaksi eri vaihtoehtoa: DO_SAVE- ja PRINT-komennot. Molemmilla komennoilla voidaan tallentaa valittu ohjelma suoraan muistitikulle. DO_SAVE-komento tallentaa ohjelman .tp-muotoisena tiedostona, joka sisältää ohjelman tiedot robotin ymmärtämällä kielellä, kun taas PRINT-komento tallentaa saman ohjelman .ls-tiedostoksi. Käyttäjän on helpompi lukea ja muokata KAREL-ohjelmointikieltä sisältävää .ls-tiedostoa, koska se on kirjoitettu ASCII-merkistön merkeillä.



Kuva 12. Esimerkki .ls-tiedostosta

Itse testiohjelma sisälsi vain muutamia tallennettuja pisteitä, joihin R-2000iB 165F -robotin oli helppo liikkua, joten myös virtuaalisen mallin pitäisi ylettyä niihin ilman ongelmia. Testiohjelma tuotiin ohjelmaan, mutta sitä ei voitu kuitenkaan simuloida ohjelman ilmoittamien virheiden takia. Virhekoodien syyt etsittiin ja kaikki ongelmat liittyivät oikean kontrollerin asetusten ja virtuaalisen kontrollerin asetusten eroavaisuuksiin. Ongelmat saatiin ratkaistua ja robottiohjelman simulaatio toimi moitteettomasti.

Kommunikointi kontrollerilta ROBOGUIDE-ohjelmalle toimi hyvin, ja seuraavaksi ohjelmien siirtoa testattiin toisinpäin. Tätä varten tehtiin uusi testisohjelma ROBOGUIDE-ohjelmalla ja siihen asetettiin samat asetukset kuin oikealla robotilla. Testiohjelma tallennettiin .tp-tiedostoksi ja saatiin siirrettyä robotille kontrollerin LOAD-komennolla, jossa kontrolleri kopioi omaan muistiinsa halutun tiedoston. Kopiointi onnistuu vain, jos tiedosto on oikeaa tiedostotyyppiä ja sen nimi on kirjoitettu täsmälleen samalla tavalla kuin se on muistitikulla. Testiohjelma ajettiin läpi oikealla robotilla ilman ongelmia. Tässä vaiheessa laboratorion R-2000iB-robotilla ei voitu

enää tuoda merkittävää lisäarvoa tutkimukselle, jonka vuoksi päätettiin siirtyä ohjelmoimaan kohdeyrityksen M-6i-robottia. Jääkiekkomaskin leikkausratojen ohjelmoimisesta R-2000iB-robotille ei olisi ollut hyötyä, koska ohjelmat eivät olisi olleet siirrettävissä eteenpäin M-6i-robotille. Tämä johtui siitä, että robottiohjelmaan tallennetut pisteet sisältävät aina kyseisen robotin eri nivelten arvot, eri kokoisilla roboteilla pisteet eivät olisi olleet samassa kohdassa.

4.5 Kohdeyrityksen FANUC M-6i -robotti

Kohdeyrityksellä on käytössään FANUC M-6i -robotti, jossa on kontrollerina FANUC R-J3 V5.30. Kontrolleri on vanhempi kuin laboratorion R-30iA, joten siinä on USB-portin sijasta PCMCIA-portti. Seuraavaksi selvitettiin, mitä osia tarvitaan tiedonsiirtoon PC:n ja R-J3-kontrollerin välillä. FANUC-konsernin R-J3-kontrollerin esitteestä selvisi, että kyseessä on tyypin 2 PCMCIA-väylä. Työtä varten tilattiin tyypin 2 PCMCIA-adapteri, jolla pystyttiin tallentamaan tiedostoja tyypin 1 CF-kortille. Seinäjoen Ammattikorkeakoululta saatiin käytettäväksi 32 megatavun kokoinen tyypin 1 CF-muistikortti sekä CF-muistikortinlukija, joka kytkettiin PC:hen USB-portin välityksellä.

Kuva 13. PCMCIA-adapteri ja CF-muistikortti

4.5.1 Solun mallinnus

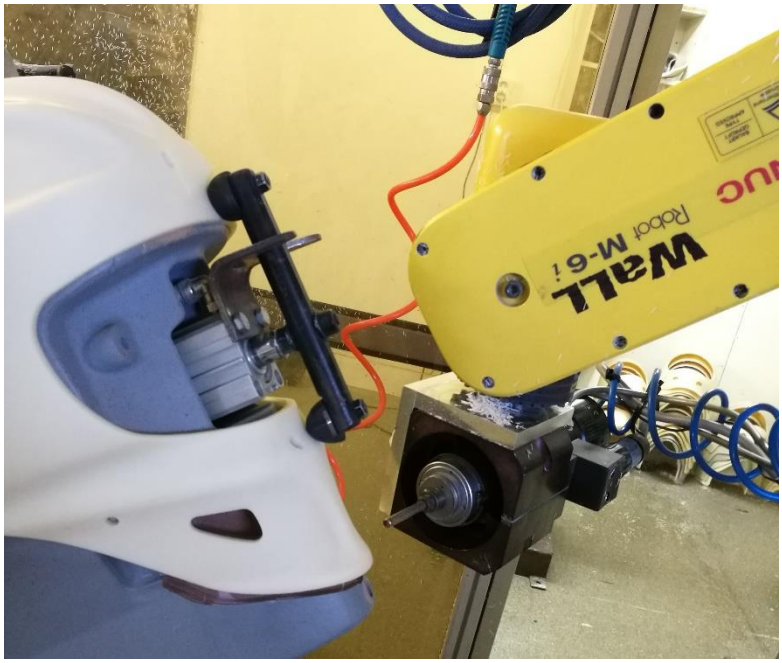
Robotin 3D-malli saatiin suoraan ROBOGUIDE-ohjelman omasta kirjastosta ja sen kontrolleriksi valittiin R-J3 V5.30. Oikean robotin työkalupisteen koordinaattien tiedot saatiin kontrollerista ja samat arvot asetettiin robotin 3D-mallille. ROBOGUIDE-ohjelma näyttää robotille asetetun työkalupisteen vihreänä pisteenä. Siirrettäessä työkalupistettä myös vihreä piste siirtyy visuaalisesti oikeaan kohtaan, mistä oli paljon apua simulaation hahmottamisessa.

Edellisen kerran yritettäessä jääkiekkomaskin 3D-malli ei toiminut halutulla tavalla Process Simulate -ohjelmassa. ROBOGUIDE-ohjelmakaan ei pystynyt tunnistamaan jääkiekkomaskin geometriaa. Mallia ei saatu käännettyä onnistuneesti millekään ROBOGUIDE-ohjelman hyväksymälle formaatille. Ensin maskin pintamallia koitettiin muokata Siemens Solid Edge ST8- ja Autodesk 3ds Max -ohjelmilla, mutta geometriaa ei saatu muokattua kunnolla. Seuraavaksi testattiin Siemens NX 10 -ohjelmaa, josta löytyi mahdollisuus tallentaa malli ROBOGUIDE-ohjelman hyväk-

symään muotoon. Kyseinen suunnitteluohjelma oli jo entuudestaan tuttu, ja pintamallista saatiin tehtyä kiinteä 3D-malli. ROBOGUIDE-ohjelma tunnisti jääkiekkomaskin 3D-mallin geometrian, ja malli oli tarpeeksi tarkka etäohjelmointia varten.

4.5.2 Virtuaalisen solun kalibrointi

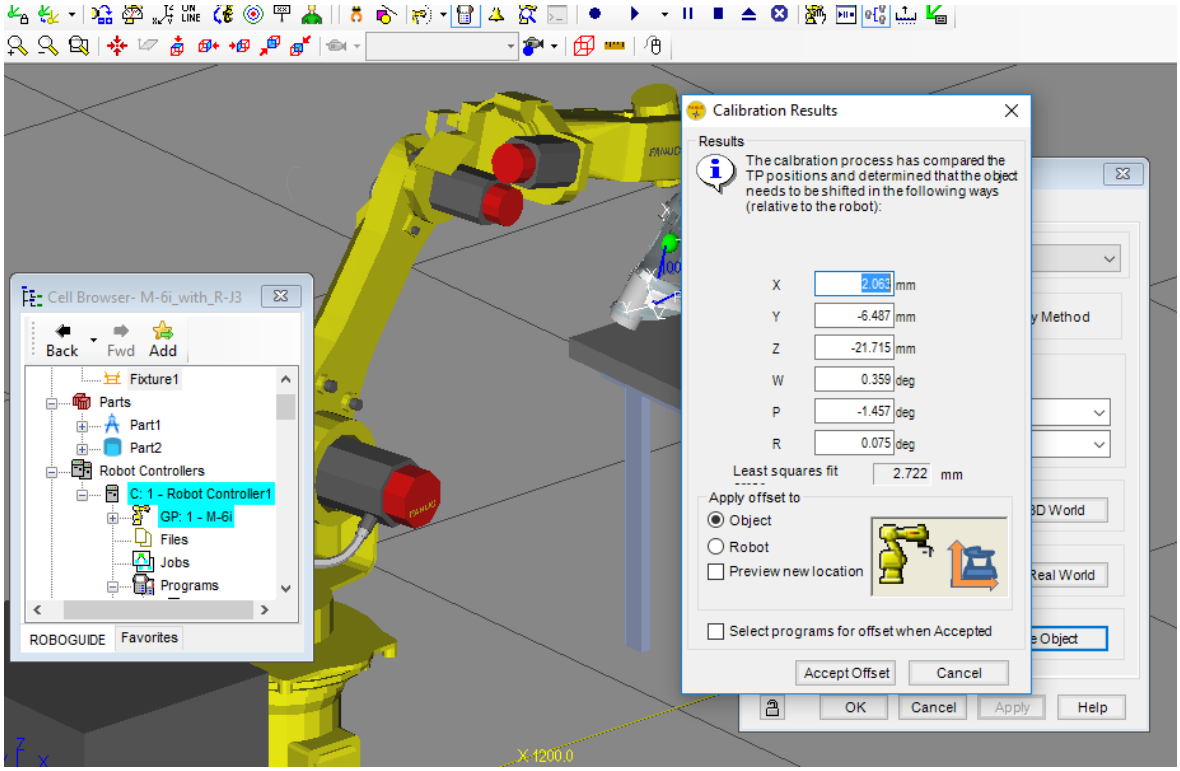
Etäohjelmointia varten virtuaalisen solun täytyy vastata tarkasti oikeaa solua. ROBOGUIDE-ohjelmasta löytyi erillinen työkalu, jolla virtuaalinen solu saatiin paikoitettua oikean solun mukaiseksi. Paikoitustyö oli kolmevaiheinen. Ensiksi virtuaalisella robotilla tehtiin kalibrointiohjelma, johon tallennettiin kolme pistettä eri puolilta simulaatiota. Pisteet valittiin maskin reikien reunoilla, jotta oikean solun maskista olisi helppo löytää samat kohdat pisteiden kalibrointia varten. Kalibrointiohjelma tallennettiin käynnissä olevan projektin tiedostokansioon ja kopioitiin muistikortille.



Kuva 14. Oikean solun kalibrointi

Seuraavassa työvaiheessa muistikortille kopioitu kalibrointiohjelma tuotiin oikealle kontrollerille, ja robotin työkalupiste ajettiin oikean maskin pinnalla tarkasti samoihin kohtiin kuin simuloidussa kalibroinnissa. Kalibrointiohjelman kolmen kalibrointipisteen koordinaatit päivitettiin uusiin koordinaatteihin käyttämällä TOUCH UP -komentoa. Viimeisessä työvaiheessa päivitetty kalibrointiohjelma tuotiin muistikortilla

PC:lle ja se tallennettiin samaan ROBOGUIDE-projektin tiedostokansioon kuin aikaisemmin. Uusi kalibrointiohjelma korvasi vanhan, ja ROBOGUIDE-ohjelmalla laskettiin näiden kahden kalibrointiohjelman pisteiden väliset eroavaisuudet. Laskelman lopputuloksena simuloidun robottisolun osat siirtyivät oikeille kohdille ja robottisolu saatiin vastaamaan todellista robottisolua.



Kuva 15. Solun kalibroinnin tulokset

Solun kalibroinnin jälkeen simulaatiossa voitiin varmistaa robotin ulottuvuus. Robotti ylettyi hyvin kaikkiin tarvittaviin paikkoihin, joten layoutin muuttaminen ei ollut tarpeellista. Kuitenkin asia täytyi vielä varmistaa oikealla robotilla, koska aikaisemmin M-6i-robottia ohjelmoitaessa oli päädytty tulokseen, että maskin leuan leikkaamista ei voida suorittaa robotilla. Tämä johtopäätös oli johtunut pääosin robotin nivelkulmien aiheuttamista kinemaattisista singulariteetti-ongelmista, jonka vuoksi robotti ei enää pystynyt liikkumaan lineaarisesti seuraavaan haluttuun pisteeseen. Maskin takaosan kaventamista ei oltu aikaisemmin yritetty ohjelmoida robotille.

Todellisen robotin työkalupiste ajettiin lähelle haluttua leikkauslinjaa terä pystysuorassa maskia kohden. Huomioon täytyi ottaa robotin työkaluun tulevat paineilmaletku ja sähköjohto, jotta ne eivät lähde irti tai joudu liian tiukalle. Robotti ajettiin

maskin luo erilaisilla nivelkonfiguraatioilla ja löydettiin konfiguraatio, jolla robotti ei ajaudu kinemaattiseen singulariteettiin. Lisäksi katsottiin, että nivelarvot eivät ole lähellä minimi- tai maksimiarvoja, jotta robotilla jää hyvin liikkumisvaraa. Robotti ylettyi myös maskin takaosaan ilman ongelmia.

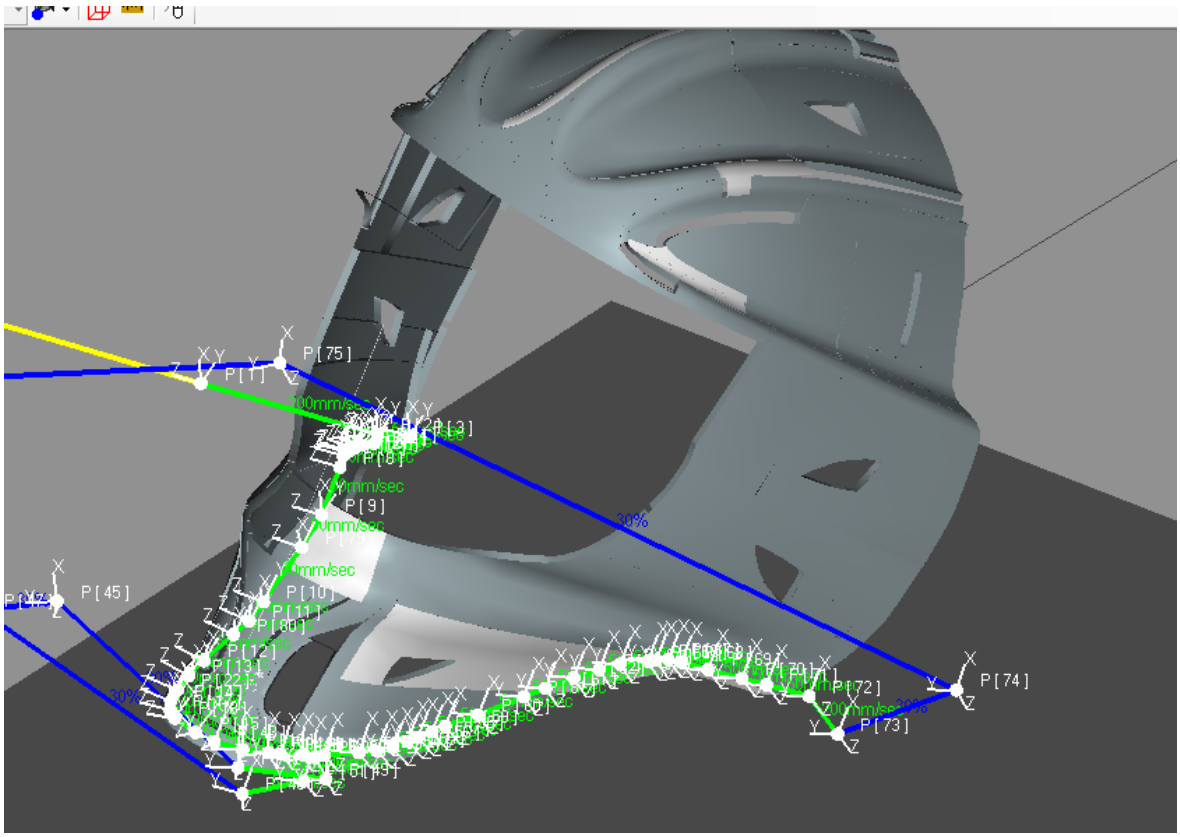
4.5.3 Ohjelmointi ja testaus

Robotilla leikatun leuan täytyy olla samanlainen kuin käsijyrsimellä leikattu, jotta varmistetaan lopputuotteen pysyminen mahdollisimman samankaltaisena. Tämän vuoksi valmis, käsin jyrsitty maski kiinnitettiin hydrauliseen työpöytään, jotta sen ulkoreunan avulla voitiin tehdä kohdistuspisteitä ROBOGUIDE-ohjelmalla tehtävää leikkausrataa varten. Uuteen ohjelmaan tehdyt pisteet tuotiin ROBOGUIDE-ohjelmaan ja ne toimivat perustana leukaosan leikkaamiselle. ROBOGUIDE-ohjelmassa on piirtotoiminto, jolla voidaan piirtää pisteitä monella eri tavalla kappaleiden pinnalle. Piirtotoiminnolla piirretylle leikkausradalle voidaan määrittää, miten työkalun orientaatio on suhteessa kappaleen pintaan sekä määrittää työkalun kääntyminen siirryttäessä robottiohjelman seuraavaan pisteeseen. Leikkausrata koostuu useasta toisiaan lähellä olevasta pisteestä, joihin yleensä liikutaan lineaariliikkeillä.

Valmiiseen leikkausrataan lisättiin vielä ROBOGUIDE-ohjelmassa tarvittavat lähestymis- ja poistumispisteet, jonka jälkeen ohjelmaa voitiin testata oikeassa solussa. Lähestymis- ja poistumispisteet ajettiin FINE-asetuksella, jolloin robotti menee haluttuun pisteeseen ja pysäyttää liikkeen täysin ennen seuraavaan pisteeseen siirtymistä. Kaikki ohjelman leikkauspisteet tehtiin CNT100-asetuksella, jossa robotti jatkaa seuraavaan pisteeseen ilman pysähdystä, ylläpitäen pisteessä määrätyn liikenopeuden.

Ensimmäistä testausta varten jyrsinterä irrotettiin robotista kokonaan vahinkojen välttämiseksi. Ohjelmaa ajettiin hitaalla käsiajonopeudella läpi ja etsittiin samalla virheitä ohjelmasta. Ohjelmaa ajettaessa huomattiin, että letkut eivät olisi kääntyneet robotin työkalun mukana koko leikkausreittiä, joten leikkaus jouduttiin tekemään kahdessa osassa. Ensimmäisen leikkauksen jälkeen robotti nostaa terän pois kypärästä ja robotti kääntää nivelet parempaan asentoon, jonka jälkeen se palaa jatkamaan leikkauksen loppuun. Kun nämä leikkauspisteet saatiin hiottua kuntoon,

ohjelma toimi hyvin maskin pinnalla. Jyrsinterä saatiin pureutumaan maskin sisälle määrittämällä työkalulle oma offset, jonka mukaan työkalupiste siirtyy maskin pinnalta sen sisälle. Offset-asetuksia voidaan määrittää teach pendantilla ja ne tallennetaan kontrollerin Position Registers -kansioon. Ohjelmaa tehtäessä voidaan määrittää aktiivisena oleva offset kutsumalla se Position Registers -kansiosta. Kutsuttu offset voidaan lisätä halutessa jokaisen ohjelman liikekomennon perään, jolloin pisteen paikka siirtyy offset-komennon asetusten mukaisesti. Aktiivinen offset-asetus ei kuitenkaan automaattisesti siirrä ohjelmaan tallennettuja pisteitä. Jyrsinterän offset-asetukseksi asetettiin siirtymistä -5 millimetriä Z-akselin suuntaan. Leuanleikkaus-ohjelman leikkaavien pisteiden perään haluttu offset lisättiin komennolla Tool_Offset, jolloin haluttu offset-siirtymä liikutaan työkalukoordinaatiston mukaan. Tässä tapauksessa työkalukoordinaatiston Z-akseli oli jyrsinterän suuntainen, ja leikatessa jyrsinterä on pystysuorassa suhteessa maskiin, joten leikkauksessa tarvittiin vain yhtä offset-asetusta.



Kuva 16. Valmis ohjelma leukaosan leikkaamiseen

Ohjelmaan lisättiin vielä komennot fyysisten lähtöjen päälle asettamiselle sekä ajastin. Ajastin odottaa, että jyrsinterä on saavuttanut täyden pyörimisnopeuden ennen itse leikkauksen alkamista.

Leikkauslinja leualle oli hyvä, mutta leikkausjälki hieman rosoinen. Leikkauksen jälkeen maskit hiotaan hiekkapaperilla maalausta varten, mutta hyvä leikkausjälki vähentäisi jälkikäsittelyyn kuluvaa aikaa. Leukaosan leikkausta varten päätettiin tehdä kaksi eri leikkausohjelmaa. Ensimmäinen leikkaus kulkisi muutaman millimetrin päästä lopullisesta leikkauslinjasta ja toinen leikkaus viimeistelisi leuan oikeaan lopputulokseen.

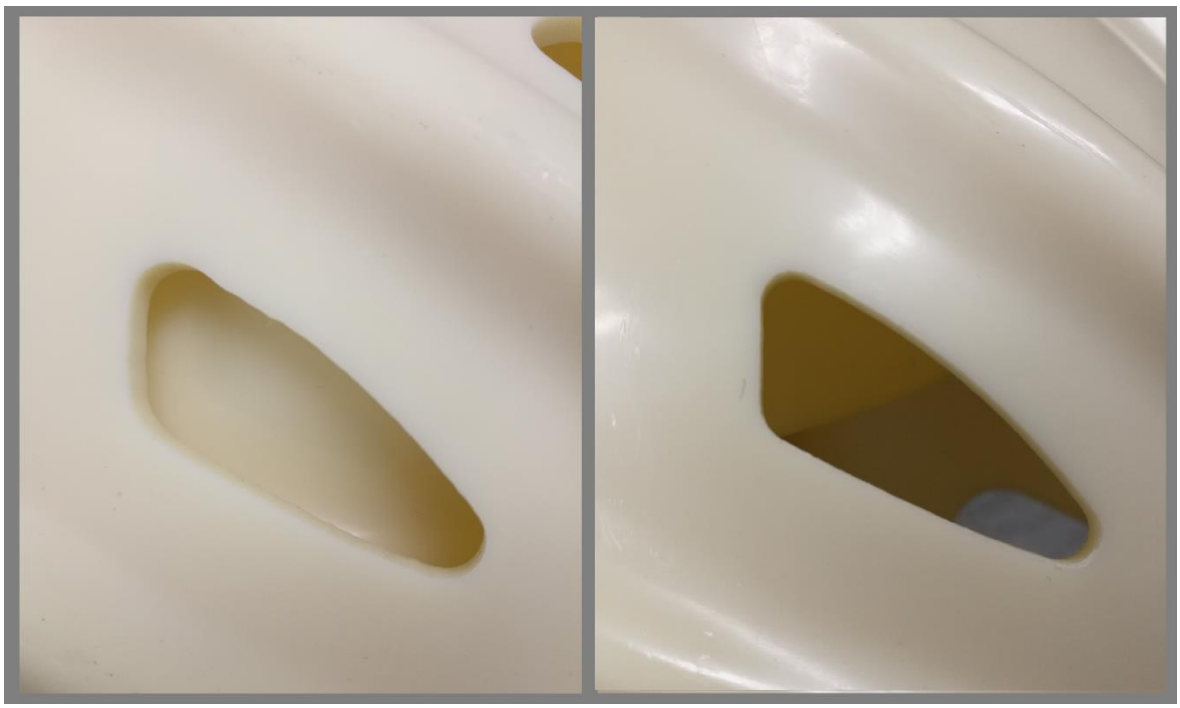
Leikkausohjelma kopioitiin ja tallennettiin uudella nimellä. Uusi ohjelma toimii aliohjelmana lopulliselle leikkausohjelmalle. Ohjelman kaikki pisteet pysyivät samoina, mutta niiden sijaintia muutettiin offset-toiminnon avulla. Ensimmäistä leikkausta varten tehtiin uusi offset-asetus, joka siirtää työkalupisteen muutaman millimetrin päähän lopullisesta linjasta. Lopullista leikkausohjelmaa muutettiin niin, että se kutsuu ja suorittaa ensimmäisen leikkauslinjan tekevän ohjelman ensin, jonka jälkeen leikataan viimeistelevä leikkauslinja. Lopputuloksena leikkausjälkeä onnistuttiin parantamaan paljon.

Maskin takaosan leikkaamista varten tehtiin uusi ohjelma. Ensin muutama reunapiste tallennettiin käsin, jonka jälkeen ohjelma tuotiin ROBOGUIDE-ohjelmaan. Maskin takaosa ei ollut niin monimutkainen kuin maskin leuka, ja robotin letkut olivat paremmassa asennossa. Tästä syystä takaosa pystyttiin leikkaamaan yhtäjaksoisesti samalla nivelkonfiguraatiolla. Maskin takaosa oli paksuudeltaan ohuempaa kuin maskin leukaosa, joten jo yhdellä leikkauksella saatiin hyvä lopputulos.

Leikkauslaatua parannettiin myös maskin ilmanottoaukoissa. Testiin otettiin reikä, jossa oli eniten epätasaisuutta. Reiän leikkausohjelma tallennettiin kontrollerilta muistikortille ja tuotiin ROBOGUIDE-ohjelmaan. Kun ohjelma avattiin simulaatiossa, pystyttiin huomaamaan leikkauksessa syntyvien epätasaisuuksien syyt.

Reiänleikkausohjelman pisteitä pystyttiin helposti muokkaamaan, ja leikkausradasta tehtiin tasaisemman näköinen. Reiän leikkausta varten tehtiin myös aliohjelma, joka leikkaa muutaman millimetrin lopullista reikää pienemmän aukon. Samoin kuin leuanleikkausohjelmassa, lopullisessa leikkauksessa jyrsinterä poistaa materiaalia

vain muutaman millimetrin. Lopputuloksena robotin leikkausjälkeä saatiin parannettua huomattavasti.



Kuva 17. Vasemmalla M-6i-robotin leikkausjälki ennen etäohjelmointia ja oikealla etäohjelmoinnin jälkeen

5 TULOKSET JA POHDINTAA

Opinnäytetyön tavoitteena oli tehdä uudet leikkausohjelmat robotille käyttäen etäohjelmointia. Uudet ohjelmat parantaisivat robotin käyttöastetta ja samalla poistaisivat käsikäyttöisen jyrsimen tarpeen. Sen lisäksi haluttiin parantaa vanhojen ohjelmien tarkkuutta ja robotin leikkausjälkeä.

Työn tuloksena saatiin aikaan kokonaan uudet leikkausohjelmat, joita voidaan käyttää uusien maskimallien leikkaamiseen. Tämän ansiosta uusien, lapsille ja nuorille tarkoitettujen maalivahdinmaskien jyrsinnät voidaan tehdä täysin robotilla. Kaikki leikkaukset ja jyrsinnät on tehty eri ohjelmiksi, jotta halutun maskin ohjelmaan voidaan kutsua kaikki kyseisen maskin valmistukseen tarvittavat työstöohjelmat.

Uusien ohjelmien leikkausjälki saatiin halutulle tasolle, ja vanhojen ohjelmien leikkausjälkeä ja tarkkuutta onnistuttiin parantamaan. Kohdeyrityksellä oli tiedossa jyrsinnän laatuun vaikuttavat tekijät, joista suurimpiin etsittiin ratkaisuja. Tärkeimpänä ongelmana oli kitkan vaikutus jyrsinnässä. Maskia työstettäessä täydellä jyrsinterän leveydellä jyrsinterään kohdistui paljon voimaa, mikä aiheutti tärinää ja epätarkkuutta. Ongelmaan keksittiin ratkaisu, jossa yksi reikä leikattiin kahdella eri työstöllä. Ensimmäinen työstö teki esileikkauksen 1–2 millimetrin päästä lopullista reikää. Toisella työstöllä jyrsinterä poisti vain ohuen kerroksen materiaalia, jonka vuoksi jyrsinterää vastustava voima ei ollut suuri. Leikkausjälki parani huomattavasti pelkästään tällä uudistuksella, joten muita muutoksia robottiin tai työkaluun ei ollut tarpeen tehdä.

Etäohjelmointi on mielestäni hyvä menetelmä robotin ohjelmointiin. Varsinkin tarkkojen ja erilaisten muotojen työstöt on helpompi tehdä etäohjelmoinnilla kuin online-ohjelmoinnilla. Työssä aluksi käytetty Process Simulate -ohjelma ei soveltunut FANUC-robottien etäohjelmointiin. Ohjelma oli kaikin puolin toimiva ja selkeä, mutta sen avulla ei pystytty luomaan FANUC-kontrollerin hyväksymää .tp-tiedostoa. FANUC käyttää omaa tiedostomuotoaan, jota ei voida tuottaa kuin FANUCIN omilla tuotteilla. Tämä poistaa mahdollisuuden tehdä robottiohjelmia millään muulla etäohjelmointiohjelmalla. Jos robotin kontrolleriin on asennettu ASCII-lisäominaisuus, voitaisiin myös .ls-tiedostoja ladata robotille. Silloin etäohjelmointiin voitaisiin käyttää esimerkiksi Process Simulate -ohjelmaa.

Työssä eniten aikaa kului tiedonhankintaan sekä ohjelmien käytön opetteluun. Erilaisia ongelmia, kuten robotin virhekoodien ilmenemistä sekä niiden ratkaisua varten apua saatiin apua robotin käyttöohjeista sekä verkosta. Etäohjelmoinnissa on monta eri vaihetta, ja osa vaiheista täytyy tehdä oikeassa järjestyksessä. Esimerkiksi robotin työkalupiste täytyy olla tiedossa ennen ohjelman tekoa, sillä muuten simuloinnissa lasketut robotin nivelkulmat ovat väärin ja tämä voi aiheuttaa robotin törmäämisen. Etäohjelmoitaessa on tärkeää ottaa kaikki asiat huomioon. Työssä käytetyssä ROBOGUIDE-ohjelmassa ei esimerkiksi ollut mahdollisuutta simuloida johtojen liikettä robotin kääntyessä, joten johtojen asento selvisi vasta robottiohjelmaa testatessa. Johdot voivat helposti jäädä jonkun nivelen taakse jumiin, mikä voi aiheuttaa johtojen vääntymisen ja työkalun hajoamisen. Tämän vuoksi robotin nivelten asentoihin täytyy kiinnittää erityistä huomiota ohjelmien suunnittelussa ja testauksessa. Koska robotti työskentelee ilman valvontaa, ohjelmoidun reitin ja robotin nivelien asennot ovat entistä tärkeämmässä asemassa.

Etäohjelmoinnissa yksi tai useampi piste voidaan tallentaa ohjelmaan monella eri tavalla, ja on tärkeää miettiä, mikä tapa on tehokkain. Tämän opinnäytetyön edetessä pohdittiin eri leikkausmenetelmiä, joita voitiin testata etäohjelmoinnin avulla. Lopputulos ja leikkausjäljen lopullinen laatu nähtiin vasta, kun robotti sai maskin leikkauksen valmiiksi.

Opinnäytetyön aihe oli mielestäni todella mielenkiintoinen ja työn tekeminen oli koko ajan mielekästä. Mielekkään aiheen ansiosta kiinnostus ei lopahtanut missään vaiheessa, vaikka uusia ongelmia tuntui tulevan vastaan heti, kun jotain saatiin aikaiseksi. Työtä tehdessä tein muistilistan, johon arvioin jäljellä olevia työvaiheita projektin onnistumiseen asti. Yritin myös ennakoida vastaan tulevia ongelmia, jotka täytyisi ratkaista. Koin tämän työskentelytavan hyvänä, koska aluksi minulla ei ollut paljon tietoa robotin ohjelmoimisesta ja uutta tietoa löytyi koko ajan lisää. Työn aikana vein eteenpäin useaa projektin eri osa-aluetta samalla hakien aiheesta lisää tietoa. Mielestäni tämän ansiosta ongelmat saatiin ratkaistua suhteellisen nopeasti ja työprosessi rullasi koko ajan eteenpäin ainakin jollain osa-alueella.

Tällä hetkellä yritys ei tarvitse lisää robottiohjelmia, mutta mahdollisesti tulevaisuudessa uusien maskimallien ja robotin työtehtävien lisääntyessä voi tarve uusille ro-

bottiohjelmille kasvaa. Silloin yritys voi miettiä esimerkiksi tässä työssä käytetyn ROBOGUIDE-ohjelman hankintaa. Jos yritykselle ilmenee tarve muuttaa jotain nykyistä robottiohjelmaa, pienet muutokset voidaan tehdä aika nopeasti myös online-ohjelmoimalla. Suurempien muutosten kohdalla voidaan miettiä ohjelmointipalvelun ostamista muualta tai oman etäohjelmointiohjelman hankintaa. Oman etäohjelmointiohjelman hankinta edellyttää kuitenkin paljon ohjelman käytön opettelua ja robotin eri toimintojen ymmärtämistä.

6 YHTEENVETO

Tämän työn tekijän mielestä opinnäytetyössä parhaaseen lopputulokseen päästiin etäohjelmoinnin ja online-ohjelmoinnin yhteistyöllä. Molempien ohjelmointitapojen hyviä puolia käytettiin hyödyksi, sen sijaan että painopiste olisi ollut täysin etäohjelmoinnissa. ROBOGUIDE-ohjelma soveltui tähän opinnäytetyöhön loistavasti ja tarvittava osaaminen ohjelman käyttöä varten opittiin nopeasti.

Opinnäytetyölle asetetut tavoitteet olivat selkeät ja ne pysyivät samana koko opinnäytetyöprosessin ajan. Tärkein tavoite oli saada uudet leikkausohjelmat valmiiksi uusia malleja varten. Tämä tavoite saavutettiin ja uudet ohjelmat onnistuttiin tekemään etäohjelmoinnin avulla. Kun tärkein osuus saatiin valmiiksi, seuraavaksi parannettiin reikien leikkausjälkeä. Leuka- ja takaosan leikkauksessa kiinnitettiin jo huomiota leikkausjälkeen, joten toimintaperiaate reikien leikkausjäljen parantamiseen oli tiedossa.

Uutta tietoa robotin ohjelmoinnista tuntui tulevan koko opinnäytetyöprosessin ajan, mikä tarkoitti uusien ongelmien selvittämistä ja ratkaisemista. Työn ansiosta FANUC-roboteista ja niiden ohjelmoimisesta opittiin paljon, mutta paljon jäi varmasti vielä opittavaa.

LÄHTEET

ABB. 2017a. RobotStudio [Verkkosivu]. ABB Oy. [Viitattu 20.3.2017]. Saatavissa: http://new.abb.com/products/robotics/robotstudio

ABB. 2017b. IRB 2400 -robotti. [Kuva]. ABB Oy. [Viitattu 4.4.2017]. Saatavissa: http://new.abb.com/products/robotics/industrial-robots/irb-2400

ABB. 2017c. IRB 910SC SCARA. [Kuva]. ABB Oy. [Viitattu 4.4.2017]. Saatavissa: http://new.abb.com/products/robotics/industrial-robots/irb-910sc

ABB. 2017d. IRB 14000 YuMI -robotti. [Kuva]. ABB Oy. [Viitattu 4.4.2017]. Saatavissa: http://new.abb.com/products/robotics/industrial-robots/yumi

ABB. 2017e. Technical data for the IRB 360 industrial robot. [Verkkosivu]. ABB Oy. [Viitattu 17.4.2017]. Saatavissa: http://new.abb.com/products/robotics/industrial-robots/irb-360/irb-360-data

All On Robots. Ei päiväystä. Lineaarirobotti. [Kuva]. All On Robots. [Viitattu 4.4.2017]. Saatavissa: http://www.allonrobots.com/cylindrical-robot.html

Arora, S. & Gupta, A. 2009. Industrial Automation and Robotics. New Delhi: University Science Press.

Braumann, J. & Brell-Cockan, S. 2013. Rob|Arch 2012: Robotic Fabrication in Architecture, Art and Design. Wien: Springer Science & Business Media.

Elatta, A. 2004. An Overview of Robot Calibration. [Verkkojulkaisu]. Information Technology Journal 3 (1). [Viitattu 1.3.2017]. Saatavissa: http://docsdrive.com/pdfs/ansinet/itj/2004/74-78.pdf

FANUC. Ei päiväystä. M-1iA/0.5A -deltarobotti. [Kuva]. FANUC CORPORATION. [Viitattu 4.4.2017]. Saatavissa: http://www.fanuc.eu/fi/fi/robotit/robottisuodatin-sivu/m1-sarja/m-1ia-05a

FANUC. 2017. ROBOGUIDE - FANUC Simulation Software. [Verkkosivu]. FANUC CORPORATION. [Viitattu 20.3.2017]. Saatavissa: http://robot.fanucamerica.com/products/vision-software/roboguide-simulation-software.aspx

Gan, Z. & Tang, Q. 2011.Visual Sensing and its Applications: Integration of Laser Sensors to Industrial Robots. New York: Springer.

Gill, A. & Krar, S. 2003. Exploring Advanced Manufacturing Technologies. New York: Industrial Press Inc.

Kandray, D. 2010. Programmable Automation Technologies. New York: Industrial Press Inc.

Karvinen, P. 2017. Toimitusjohtaja. Wall Maskit. Haastattelu 26.4.2017.

Kuttan, A. 2007. Robotics. New Delhi: International Publishing House Pvt. Ltd.

Low, K-H. 2007. Industrial Robotics: Programming, Simulation and Applications. Mammendorf: Pro literatur Verlag Robert Mayer-Scholz

LPR Global. 2017. Lineaarirobotti. [Kuva]. LPR Global, Inc. [Viitattu 4.4.2017]. Saatavissa: http://www.uskoreahotlink.com/products/factory-automation/gantry-robots-rack-and-pinion/

Marquina, J. 2010. Curso FANUC I: Programación TPE. [Verkkokirja]. PSA Peugeot Citroën. [Viitattu 17.4.2017]. Saatavissa: http://myslide.es/documents/curso-fanuc-i-m07-programacion-2010.html

Mogab, I. 21.9.2016. 4 Types of Collaborative Robots to Increase Productivity. [Verkkosivu]. Patti Engineering. [Viitattu 4.4.2017]. Saatavissa: http://pattiengineering.com/blog/4-types-collaborative-robots/

NACHI. 2016. Robotin standardisykli. [Kuva]. NACHI-FUJIKOSHI CORP. [Viitattu 17.4.2017]. Saatavana: http://www.nachi-fujikoshi.co.jp/eng/rob/hand/mz07a.htm

Nof, S. 1999. Handbook of Industrial Robotics. 2. p. New York: John Wiley & Sons, Inc.

Robotpark Academy. 2015. DICTIONARY of ROBOTICS. [Verkkosivu]. Robotpark. [Viitattu 17.4.2017]. Saatavissa: http://www.robotpark.com/academy/robotics-learning-center/dictionary-robotics/

Shazar, A. & Tabactsidis, P. 2015. Computer Integrated Manufacturing: Integration Project Report. [Verkkojulkaisu]. [Viitattu 17.4.2017]. Saatavissa: https://www.slideshare.net/AvivShazar/cimintegrationfinalreport

Siemens. 2017. Process Simulate. [Verkkosivu]. Siemens AG. [Viitattu 20.3.2017]. Saatavana: https://www.plm.automation.siemens.com/en_us/products/tecnomatix/manufacturing-simulation/assembly/process-simulate.shtml

Spilsbury, L. & Spilsbury, R. 2017. Incredible Robots in Industry. Oxford: Raintree.

Tarn, T-J., Chen, S-B. & Zhou, C. 2007. Robotic Welding, Intelligence and Automation. New York: Springer.

Wilson, M. 2015. Implementation of Robot Systems. Amsterdam: Elsevier Inc.