# Offline-ohjelmiston soveltuvuus järjestelmäintegraattorin liike-toimintaan

Joonas Tikka

OPINNÄYTETYÖ
Maaliskuu 2020

Konetekniikka
Koneautomaatio

**TIIVISTELMÄ**

Tampereen ammattikorkeakoulu
Konetekniikka
Koneautomaatio

TIKKA, JOONAS:
Offline-ohjelmiston soveltuvuus järjestelmäintegraattorin liiketoimintaan

Opinnäytetyö 52 sivua, joista liitteitä 1 sivu
Maaliskuu 2020

---

Opinnäytetyössä selvitettiin robottien offline-ohjelmointityökalun käyttömahdollisuuksia kappaleen viimeistelyyn liittyvissä asiakastapauksissa ja tutkittiin ohjelmiston investointikannattavuutta. Ensimmäinen asiakastapaus oli asiakkaan robottisolun käytettävyyteen liittyvä kehitysprojekti. Toinen asiakastapaus oli esiselvitys robotisoidun jäysteenpoiston toteuttamisesta offline-ohjelmistolla. Ohjelmiston investointikannattavuutta on tutkittu työssä asiakastapauksia laajemmasta näkökulmasta. Työ on tehty empiirisenä tutkimuksena ja työn toimeksiantajana toimi JTA Connection Oy

Tuloksena saatiin käsitys ohjelmiston tärkeimmistä ominaisuuksista sekä niiden rajoitteista. Myös ohjelmiston soveltuvuudesta JTA Connectionin kaltaisen järjestelmäintegraattorin liiketoimintaan saatiin riittävän hyvät tulokset. Kaikki tuloksissa saadut testiajot ajettiin KUKA KR 60 HA nivelvarsirobotilla. Investoinnin takaisinmaksun osalta esitetyt tulokset ovat viitteellisiä.

Lopputuloksena todettiin, että ohjelmisto ei ole tällä hetkellä sopiva yritykselle, koska yrityksessä on vahva osaaminen robotiikasta, mutta ei CAM-ohjelmoinnista. Ohjelmisto soveltuu paremmin yrityksille, joissa työstettävillä tuotteilla on suuri volyymi, tuotteet vaihtelevat ja osaamista robotiikasta ei ole. Tutkimuksen jatkokehitystä on suositeltavaa jatkaa kattavalla CAM-ohjelmointikokemuksella.

---

Asiasanat: robotiikka, offline-ohjelmointi

**ABSTRACT**

Tampere University of Applied Sciences
Degree programme of Mechanical Engineering
Machine Automation

TIKKA, JOONAS:
Suitability of Offline Programming Software in a System Integration Business
Bachelor's thesis 52 pages, appendices one page
March 2020

---

The purpose of this study was to evaluate possibilities of an offline programming software combined with robots in machining applications, and in addition to examine the cost-efficiency of the software. This study was done as a case study, which consisted of two different cases. The first case was a development project of a human machine interface which was aimed to improve the usability of the current solution. The second case was preliminary examination of automatizing the deburring of a complex body using offline programming software. This study was commissioned by JTA connection Oy.

As a result, a comprehensive understanding of the advantages and disadvantages of the software was obtained. Also, the test results gave adequate data to determine the applicability of the software for a system integrator's such as JTA Connection. The test drives were done using the KUKA KR 60 HA industrial robot. All the exact results of cost-efficiency are suggestive.

The findings indicate that the software is not an appropriate solution to either of the cases. This is due to cost-efficiency and lack of CAM programming skills in comparison to traditional robot programming within the company. The software is more suitable for companies that do not have extensive knowledge of robot programming, have knowledge of CAM programming and preferably manufacture high quantities of various products. For further studies it is highly recommended that they should be based on an extensive knowledge of CAM.

---

Key words: robotics, offline programming

**SISÄLLYS**

## LYHENTEET JA TERMIT

| | |
|---|---|
| CAD | Computer-Aided Design |
| CAM | Computer-Aided Manufacturing |
| CNC | Computer Numerical Control |
| KRL | KUKA Robot Language |
| NC | Numerical Control |
| PTP | Point to Point |
| TCP | Tool Center Point |

# 1 JOHDANTO

Työstettyjen kappaleiden viimeistely on usein vaarallista ja uuvuttavaa työtä. Nykypäivänä tämän kaltaisia töitä pyritään automatisoimaan mahdollisimman pitkälle esimerkiksi hyödyntämällä robotiikkaa. Erilaisista kappaleista on mahdollista poistaa jäysteitä onnistuneesti jo nykyisillä menetelmillä, mutta opinnäytetyön toimeksiantajalla JTA Connectionilla on tarve kehittää näitä menetelmiä kustannustehokkaammiksi ja modulaarisemmiksi.

Jäysteenpoiston toteuttaminen roboteilla on mahdollista, mutta perinteisesti robotin käsiohjaimella liikepisteiden opettaminen on hidasta ja vaatii robotiikan osaamista, jota ei välttämättä ole asiakasyrityksellä. Opinnäytetyön tarkoituksena on tutkia offline-ohjelmistoa, jonka avulla robotin liikeratojen opettaminen nopeutuisi. Tavoitteena on myös selvittää, millaisia investointeja ja osaamista ohjelmiston käyttöönottamien edellyttää.

Tutkimus on toteutettu tapaustutkimuspohjaisena empiirisenä tutkimuksena, jossa on ollut kaksi erilaista tapausta. Ensimmäisessä tapauksessa tarpeena oli selvittää ohjelmiston soveltuvuus muovilaatikon aukotukseen jo olemassa olevan järjestelmän tilalle. Toisessa tapauksessa tutkittiin monimutkaisen kappaleen jäysteenpoiston automatisointia. Kaikki testiajot suoritettiin KUKA KR 60 HA nivelvarsirobotilla.

# 2 Robottien ohjelmointi

## 2.1 Ohjelmointitekniikoita

Robottien ohjelmointi yleisesti tarkoittaa niiden työtehtävien opettamista robotille, joita sen oletetaan suorittavan. Eri valmistajilla on roboteille erilaisia ohjelmointikieliä, jotka voivat erota toisistaan huomattavasti. Ohjelmointikielten erilaisuudesta huolimatta, robottien ohjelmointi voidaan pääsääntöisesti jakaa kolmeen eri tyyliin: kirjoittamalla ohjelmointiin, käsiohjaimella opettamiseen ja graafiseen ohjelmointiin. (Tirloni, Fassi, Legnani 2012, 18.)

Kaikki kolme ohjelmointitapaa voidaan jakaa edelleen online- ja offline-ohjelmointiin. Offline- ja online-ohjelmointi eroavat toisistaan niin, että offline-ohjelmoinnissa ohjelmointi suoritetaan erillisellä tietokoneella, joka on robotista riippumaton. Online-ohjelmointi puolestaan tarkoittaa, että robotin ohjelma tehdään suoraa robotin käsiohjaimella. Offline-ohjelmoinnin aikana robotti voi työskennellä tuotannossa toisin kuin online-ohjelmoinnin aikana. (Tirloni ym. 2012, 19.)

### 2.1.1 Kirjoittamalla ohjelmointi

Kirjoittamalla ohjelmointi tehdään joko online-tilassa tai offline-tilassa. Tyypillisesti kirjoittamalla ohjelmoinnissa käytetään tietokoneen näppäimistöä, mutta kirjoittamalla ohjelmointia voi tehdä myös robotin käsiohjaimella. Käsiohjaimella ohjelmoidessa peruskäskyt löytyvät usein valmiista valikoista, eikä niitä tarvitse erikseen kirjoittaa. (Tirloni ym. 2012, 20.)

Eri robottivalmistajien välisistä eroavaisuuksista huolimatta, monista robottien ohjelmointikielistä löytyy yhtäläisyyksiä. Osa ohjelmointikielistä muistuttaa Basicia, kun taas jotkut ohjelmointikielet ovat Pascal, C ja C++ pohjaisia. Siitä huolimatta, että ohjelmointikielten välillä voi olla toiminnallisia eroja sekä erilainen syntaksi, robottien ohjelmoinnissa esiintyy kuitenkin aina liikekäskyt, funktiot ja datatyypit. (Tirloni ym. 2012, 19.)

Kirjoittamalla ohjelmoidessa robotille syötettävät peruskomennot sisältävät muun muassa paikkatiedon ja nopeuden kuvan 1 mukaisesti. Kirjoittamalla ohjelmoidessa pystytään myös kommunikoimaan ulkoisten laitteiden kanssa, kuten erilaisten anturien, logiikoiden ja toimilaitteiden kanssa. Ulkoisten laitteiden kanssa kommunikointi mahdollistaa robotin liikkeiden koordinoimisen muiden laitteiden kanssa sekä auttaa vikatilojen selvittämisessä. (Tirloni ym. 2012, 19.)

```
$BASE={X 938.1865,Y 1113.6939,Z 776.1112,A 90.0000,B 0.0000,C 0.0000}
$TOOL={X 288.6800,Y -0.6230,Z 53.5810,A 0.0000,B 90.0000,C 0.0000}

$ADVANCE = 5
PTP  {A1 -90.0000,A2 -116.7900,A3 116.3500,A4 0.0000,A5 80.6700,A6 0.0000}
PTP  {X 98.9022,Y -62.0903,Z 57.5786,A 62.8826,B -12.7066,C 164.4259,S 2,T 35}
PTP  {X 102.2577,Y -61.4282,Z 48.1817,A 62.8826,B -12.7066,C 164.4259,S 2,T 35}
PTP  {X 117.3575,Y -58.4486,Z 5.8955,A 62.8826,B -12.7066,C 164.4259,S 2,T 35}
$VEL.CP = 0.0002  ;0.2mm/s
LIN  {X 119.0352,Y -58.1176,Z 1.1971,A 62.8826,B -12.7066,C 164.4259} C_DIS
```

KUVA 1. Esimerkki KUKAn liikekäskyistä.

### 2.1.2 Käsiohjaimella ohjelmointi

Käsiohjaimella ohjelmointi on online-ohjelmointitapa, jossa robottia ajetaan käsiajolla haluttuihin paikkoihin. Käsiajolla robotin liikuttaminen tapahtuu yleensä sauvaohjaimella tai painikkeilla esimerkiksi kuvan 2 mukaisella ohjaimella. Käyttäjä määrittää itse haluamansa paikat visuaalisesti, jonka jälkeen paikkapiste tallennetaan robotin ohjaimen muistiin. Robotin paikkapisteiden opettaminen tapahtuu manuaalitilassa. Kun halutut pisteet ovat opetettu, voidaan robotti vaihtaa automaattitilaan, jolloin robotti ajaa opetetut pisteet järjestyksessä. (Tirloni ym. 2012, 21.)


KUVA 2. KUKA smartPAD käsiohjain

Käsiohjaimella ohjelmointi on offline-ohjelmointimenetelmiin verrattuna hyvin hidasta, työlästä ja joissain tapauksissa kallista. Jotkut sovellukset saattavat tarvita useita liikepisteitä, joiden opettaminen käsin on hyvin työlästä ja aikaa vievää (Maw 2019). Ohjelmoinnin korkeisiin kustannuksiin vaikuttaa joissakin tapauksissa myös se, että robottia ei voi käyttää tuotannossa, kun sitä ohjelmoidaan käsiohjaimella. Online ohjelmoinnin haittana on myös se, että robotin ohjelmointia ei voi aloittaa ennen kuin robotti ja muu laitteisto on hankittu.

### 2.1.3 Graafinen ohjelmointi

Useimmille robottimerkeille on tarjolla graafisia simulaatio-ohjelmia. Robottivalmistajien omat ohjelmistot eivät usein tue muita kuin oman brändin robotteja. Graafiset ohjelmointityökalut mahdollistavat offline-ohjelmoinnin, sillä ohjelmointi suoritetaan ulkoisella PC:llä (Tirloni ym. 2012, 22). Koska ohjelmointi tehdään ulkoisella PC:llä, robotteja on mahdollista ohjelmoida keskeyttämättä tuotantoa. Ohjelma voidaan luoda hyvin pitkälle simulaatiossa ja siirtää oikealle robotille vasta, kun ohjelma on simulaatiossa valmis.

Simulaatiossa pystytään tarkastamaan muun muassa robotin liikeratojen lisäksi mahdolliset törmäykset, robotin ulottuvuudet ja työsyklin vaatima aika. Esimerkiksi osa mahdollisista törmäyksistä voidaan poistaa ohjelmasta jo simulaatiossa, jolloin mekaanisten vahinkojen riski pienenee. Simulaatioista näkee myös hyvin, pitääkö robottisolun asettelua muuttaa vielä ennen sen kokoamista. (Tirloni ym. 2012, 24.)

Robottivalmistajien omien ohjelmistojen lisäksi on myös tarjolla ulkoisten yritysten valmistamia ohjelmistoja. Ulkopuolisen ohjelmiston käyttämisestä voi olla hyötyä, jos esimerkiksi käytössä on useita eri robottimerkkejä. Näissä tapauksissa on mahdollista käyttää ulkopuolista ohjelmistoa kuten Robotmasteria. Robotmaster tukee kaikkia yleisimpiä robottimerkkejä, joita ovat esimerkiksi ABB, Fanuc ja KUKA. Eri valmistajien omat simulaatio-ohjelmat, kuten ABB:n RobotStudio tai Fanucin RoboGuide eivät tue muita, kuin oman brändin robotteja. (Maw 2019.)

## 2.2 Graafisen ohjelmoinnin hyödyt ja haitat

Graafisen ohjelmoinnin hyötyinä pidetään sen nopeutta verrattuna käsiohjaimeen, mahdollisuutta ohjelmoida pysäyttämättä tuotantoa ja ohjelmoinnin aloittamista ennen kuin robotti tai oheislaitteet ovat hankittu (Maw 2019). Myös robotin työkalut voidaan tuoda suoraa simulaatioon, koska mitat ja robotin työkalupisteen pystyy määrittämään suoraa CAD-mallista. Simulaatio-ohjelmat usein myös tukevat ulkoisten laitteiden, kuten liukuhihnojen mallintamista.

Simulaatiot eivät kaikista eduistaan huolimatta vastaa aina todellisuutta. Simulaatiot eivät esimerkiksi huomio fysikaalisia ilmiöitä ja niiden vaikutusta robotin toimintaan täysin realistisesti. Robotti, joka siirtää raskasta kuormaa ei todellisuudessa pysähdy heti, kun seis painiketta painetaan. Robotin pysähtyminen saattaa kestää huomattavasti pidempään kuin simulaatiossa. Kiihtyvyyden muutos vaikuttaa myös robotilla siirrettävään kappaleeseen, joka voi nopean jarrutuksen seurauksena irrota robotin tarttujasta. Kaikki simulaatio-ohjelmat eivät myöskään pysty simuloimaan ympäristössä olevia esineitä kuten robotin kaapelointia, jotka robotti saattaa liikkuessaan katkaista (Maw 2019).

Robottien liikeradat luodaan pääsääntöisesti suhteessa määritettyihin koordinaatistoihin. Tämä mahdollistaa muun muassa sen, että ohjelmoija opettaa vain yhden pisteen ennalta määritettyyn koordinaatistoon ja luo loput pisteet siirtämällä opetettua pistettä esimerkiksi 20 millimetriä johonkin XYZ-koordinaatiston suuntaan (Maw 2019). Simulaatiota oikeaan maailmaan siirtäessä tämä menetelmä voi aiheuttaa ongelmia, mikäli kappaleet eivät ole samoilla paikoilla oikeassa robottisolussa ja sen simulaatiossa.

# 3 KUKA

## 3.1 KUKA ohjelmointi

KUKAn robotit ohjelmoidaan käyttäen KUKA Robot Languagea (KRL), joka on KUKAn kehittämä ohjelmointikieli. Tyypillisesti ohjelmointikielissä esiintyvien muuttujien määrittelyn, ehtolauseiden ja silmukoiden lisäksi KRL tarjoaa roboteille ominaisia komentoja, joiden avulla määritellään liikekäskyt, liikkeiden parametrit ja robotin työkalut. KRL on Pascalia muistuttava ohjelmointikieli. (Mühe, Angerer, Hoffman, Reif 2010.)

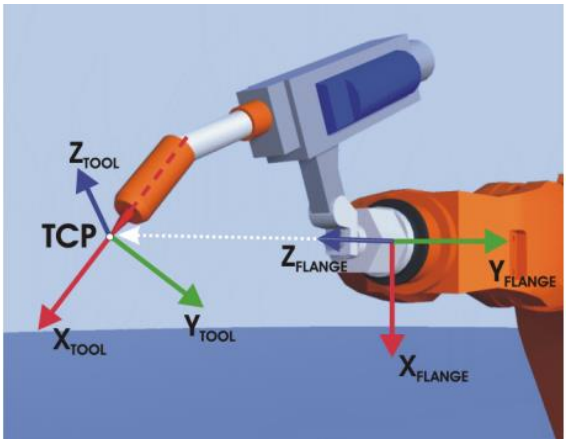
KRL ohjelma koostuu usein SRC- ja DAT-tiedostosta. DAT-tiedostoon tallennetaan vain muuttujia ja liikepisteiden sijainteja (KUKA GmbH 2015, 359). SRC-tiedosto pitää sisällään esimerkiksi liikekäskyt, ehtolausekkeet, silmukat ja digitaalisten tulojen ja lähtöjen ohjaukset. Kuten kuvassa 3 esitetystä esimerkkiohjelmasta havaitaan, muuttujia on myös mahdollista esitellä SRC-tiedostossa, kunhan esittelyn tekee omassa osiossaan. Esittelyn jälkeen muuttujia voi käyttää lauseosiossa.

```
DEF  b ( )

  DECL INT Laskuri
  ;-----esittely-------
  ;FOLD INIT
  Laskuri = 1 ;alustus

  ;ENDFOLD (INIT)

IF Laskuri < 1 THEN
PTP HOME  VEL= 100 % DEFAULT
ENDIF


END
```

KUVA 3. KUKA KRL muuttujan esittely

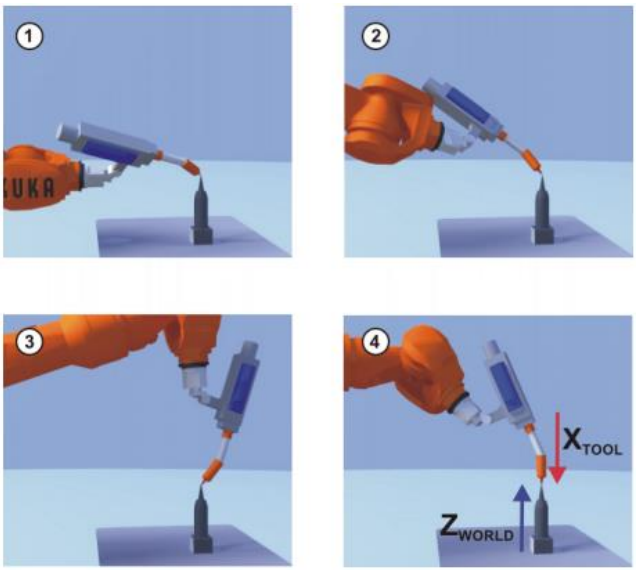
### 3.2 Työkalupisteen määrittäminen

Tool Center Point (TCP) eli työkalupiste tarkoittaa robotin työkalulle määriteltyä origoa (kuva 4). TCP:n määrittämisen voi suorittaa monilla tavoilla, joista tässä esitetään ainoastaan neljän pisteen menetelmä. Kyseinen menetelmä on oleellisin, sillä sitä on käytetty myös koeajoissa. Työkalupisteen määrittäminen tarkoittaa käytännössä käytettävän työkalun kärkipisteen sijainnin opettamista robotille suhteessa robotin 6-akselin päätylaippaan kuvan 4 mukaisesti (KUKA GmbH 2015, 125). KUKA pystyy pitämään muistissaan samanaikaisesti kuudentoista eri työkalun koordinaattidatan, jotka tallentuvat muuttujaan TOOL_DATA[1...16].




KUVA 4. Työkalupisteen sijainti robotin 6-akselin laipasta (KUKA GmbH 2015, 125).

Neljän pisteen menetelmässä robotti ajetaan manuaalisesti referenssipisteeseen neljästä eri suunnasta kuvan 5 mukaisesti. Referenssipiste voi olla mikä tahansa piste, esimerkiksi pöytään kiinnitetyn ruuvimeisselin kärki. Työkalupiste lasketaan 6-akselin laipan eri sijainneista, joten työkalun orientaatiota on suositeltavaa muuttaa jokaisen opetettavan pisteen kohdalla (KUKA GmbH 2015, 126). KUKAn ohjain ei hyväksy liian lähelle toisiaan tallennettuja pisteitä.




KUVA 5. Työkalupisteen opetuksen vaiheet neljän pisteen menetelmällä (KUKA GmbH 2015, 126).

### 3.3 Työkoordinaatiston määrittäminen

Robotin työkoordinaatiston voi määrittää kahdella tavalla, jotka ovat kolme piste metodi ja epäsuora metodi (KUKA GmbH 2015, 132). Työkoordinaatiston opettaminen on suositeltavaa, koska se helpottaa robotin ajamista manuaalitilassa esimerkiksi työstettävän kappaleen läheisyydessä. Myös liikepisteiden opettaminen työkoordinaatiston suhteen osoittautuu usein hyödylliseksi. Mikäli työstettävää kappaletta siirretään, kaikkia liikepisteitä ei tarvitse opettaa uudestaan, kun ne ovat luotu suhteessa työkoordinaatistoon. Tällöin riittää, kun työkoordinaatisto siirretään uuteen sijaintiin, koska liikepisteet liikkuvat työkoordinaatiston mukana. KUKA mahdollistaa 32 eri työkoordinaatiston tallentamisen muuttujaan BASE_DATA[1...32].

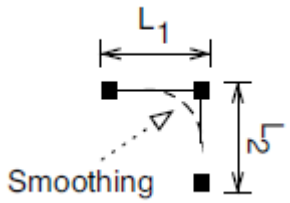
Kolmen pisteen metodissa uuden työkoordinaatiston opettaminen tapahtuu käyttämällä sellaista työkalua, joka on jo opetettu robotille. Työkalu ajetaan manuaalisesti koordinaatiston haluttuun origoon, jonka jälkeen piste tallennettaan. Tämän jälkeen työkalu ajetaan halutun positiivisen x-akselin suuntaan, jolta opetetaan toinen piste. Kolmas piste opetetaan halutulta positiiviselta y-akselilta. (KUKA GmbH 2015, 132.)

### 3.4 KUKA perusliikekäskyt

Point to Point (PTP) eli pisteestä pisteeseen liike on liiketyyli, jossa robotti liikuttaa TCP:n nopeinta mahdollista rataa pitkin pisteestä pisteeseen. PTP-liikkeen rata ei välttämättä ole suora, koska lyhin rata ei aina ole välttämättä nopein rata. Tämä on seuraus siitä, että robotin akselien liikkeet ovat kiertyviä, jonka takia mutkaisia ratoja on mahdollista ajaa nopeammin kuin suoria ratoja. LIN-liikekäsky tarkoittaa liikettä, jossa robotin TCP siirtyy suoraa reittiä pisteestä pisteeseen. CIRC-liikekäskyllä pystytään ajamaan kaarevia ratoja. CIRC-käskyllä kaari määräytyy aloituspisteen, apupisteen ja loppupisteen mukaan. Apupiste tarkoittaa pistettä, joka on alku ja loppupisteen välissä. (KUKA GmbH 2015, 277-278.)

### 3.5 Advance run

KRL lukee ohjelmaa läpi käyttäen kahta ohjelmalaskuria. Toinen ohjelmalaskuri lukee ohjelmaa ennakkoon, joka mahdollistaa liikekäskyjen laskemisen ennakkoon. Advance run toimintoa käytetään erityisesti sulauttamaan peräkkäisiä liikkeitä toisiinsa, jolloin liikekäskyjen välissä olevaan liikepisteeseen ei ajeta tarkasti kuvan 6 esimerkin mukaan (Mühe ym. 2010). Advance run parametrin maksimi arvo on 5, vakioarvo 3 ja pienin sallittu arvo likimääräistä paikotusta käytettäessä 1.




KUVA 6. Liikkeiden likimääräinen paikoitus (Mühe ym. 2010).

Kuvan esittämä oikaisu määritellään liikekäskyn perässä erillisellä parametrilla C_DIS. Tälle parametrille annetaan arvo järjestelmämuuttujalla $APO.CDIS, jonka arvo määrittää sen etäisyyden radan päätepisteestä, jolloin oikaisu voi aikaisintaan alkaa (KUKA GmbH 2015, 372). Liikekäskyjen välissä oikaisemalla säästää huomattavia määriä aikaa, sillä robotti pyrkii aina pysähtymään tarkaksi

määrätyn liikekäskyn loppupisteeseen. Tällöin aikaa kuluu, kun robotti jarruttaa ja kiihdyttää pisteiden välissä (Mühe ym. 2010).

## 3.6 KUKA CNC

KUKA CNC (Computer Numerical Control) on KUKAn julkaisema ohjelmistopaketti, joka mahdollistaa NC eli Numerical Control ohjelmien ajamisen suoraa KUKAn KR C4-robottiohjaimella. NC-ohjelmia pystytään lukemaan ja luomaan ohjaimella suoraa ilman, että NC-ohjelmaa tarvitsee kääntää ensin KRL-ohjelmaksi. NC-ohjelmat voidaan tehdä suoraa offline-ohjelmoinnilla käyttämällä CAD/CAM ohjelmistoja. (KUKA n.d.)

KUKA CNC mahdollistaa isojenkin ohjelmien lukemisen. KUKA CNC kasvattaa Advance run ominaisuuden ennakointikyvyn 150 pisteeseen saakka, joka parantaa ratojen tarkkuutta ja sulavuutta huomattavasti robotilla. (KUKA n.d.) KUKA kuitenkin suosittelee ohjelmistopakettia käytettäväksi vain työtehtävissä kuten vesileikkaus ja laserleikkaus sekä pehmeiden materiaalien, kuten muovin, puun tai alumiinin jyrsinnässä (KUKA CNC 2.1 2013, 7). Advance run ominaisuuden maksimi ennakointikyvyn kasvattamisesta huolimatta, robotit eivät suoriudu tarkkuutta vaativista koneistuksista.

### 3.6.1 NC-ohjelman suorittaminen

NC-ohjelman suorittamisen voi tehdä kahdella tavalla. NC-ohjelmaa pystyy kutsumaan KRL-ohjelman sisällä käyttäen funktiota gCodeExecute(), jolloin pystytään hetkellisesti käyttämään CNC-ohjausta. Toinen tapa suorittaa NC-ohjelmia, on käyttää kuvan 7 mukaista ohjelmaa, jossa kutsutaan cncMotion() funktiota. CncMotion funktiota käyttäessä robotti siirtyy pysyvästi CNC-tilaan, jossa se on niin kauan, kunnes ohjelma pysäytetään (KUKA CNC 2.1 2013, 53).

```
DEF cnc()
; this procedure can be called as a main program for activating the interactive CNC-mode

INI


PTP P11 VEL=100 % PDAT10 TOOL[1]:HARJ BASE[32]:CNC_TRAIN

; start-positions can be teached here e.g. a position over the milling table.
; the base of the last teached motion stays active while operating in CNC-mode
; while the tool may be overridden by the reference tool cncReferenceTool or by
; using the "T<cncToolIndex> M6" command in a nc-program


; enter CNC-mode
cncMotion()

-END
```

KUVA 7. CNC-tilaan siirtyminen

Oleellisesti NC-ohjelman suorittamiseen kuuluu mekaanisten välysten kompensointi. Mekaanisten välysten kompensointi tarkoittaa välyksistä johtuvan oikean ja lasketun sijainnin eron kompensoimista ohjelmallisesti. Kompensointia ei ole pakko suorittaa ollenkaan, mutta sen suorittaminen parantaa tarkkuutta etenkin tapauksissa, joissa liikkeen suunta muuttuu päinvastaiseksi. Tästä syystä mekaanisten välysten vaikutuksen työstöön voi havaita etenkin työstön aikana tapahtuvissa suunnanvaihdoissa. (ISG Stuttgart 2020.)

# 4 MASTERCAM JA ROBOTMASTER

## 4.1 Mastercam

Mastercam on maailman käytetyin CAD/CAM ohjelmisto, eli tietokoneavusteiseen suunnitteluun ja valmistukseen suunnattu ohjelmisto. Mastercamilla CAD-malliin pystytään tekemään erilaisia työstöratoja, joita voidaan simuloida ja kääntää suoraa NC-ohjelmaksi. Mastercam tarjoaa erilaisille työstökonetyypeille eri ohjelmistopaketteja, kuten jyrsintä, sorvaus, moniakseliset työstökoneet ja rautalankasahaus. Mastercamissa on myös tuki kolmansien osapuolien ohjelmistoille kuten Robotmasterille. (Mastercam n.d.)

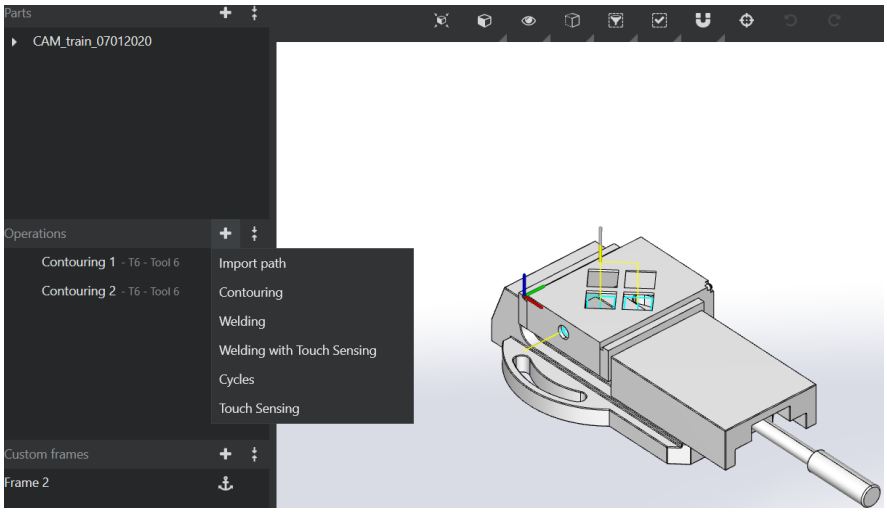
## 4.2 Robotmaster

Robotmaster on Hyperterm Inc:n luoma robottien CAM-ohjelmisto. Robotmaster integroi offline-ohjelmoinnin, simuloinnin ja ohjelman tuottamisen. Tarkoituksena on nopeuttaa ohjelmointia ja päästä perinteisestä pisteiden opettamisesta eroon, sekä luoda valmis robottiohjelma suoraa CAM-ohjelman pohjalta. Hyperterm Inc:n mukaan tämä mahdollistaa, että ohjelmointiin ei vaadita robotiikan asiantuntijaa. (Robotmaster n.d.)

Robotmasterista on käytännössä kaksi erilaista versiota, joista toinen toimii CAD/CAM ohjelmiston lisäosana ja toinen on itsenäinen ohjelma. Itsenäinen Robotmaster V7 julkaistiin 2018 ja sen tarkoituksena on, että käyttäjien ei tarvitse investoida kalliisiin CAD/CAM ohjelmistoihin ja opetella käyttämään niitä (Robotmaster V7, 2018). Tässä tutkimuksessa on käytetty Robotmaster V7:ää, sekä Mastercamin rinnalla toimivaa Robotmaster V6.6:tta.

## 4.3 Työstöradat

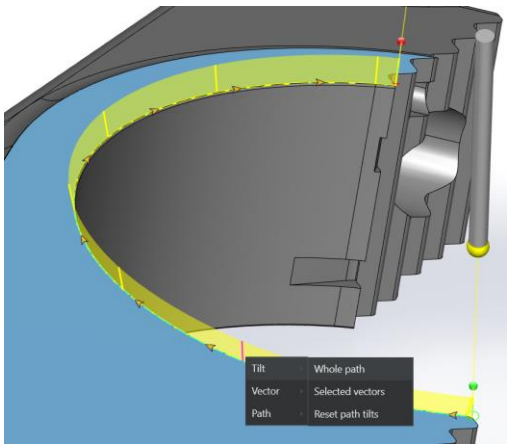
Työstöratojen luominen Robotmaster V6:lla ja V7:llä poikkeavat toisistaan merkittävästi. V7 on hyvin suoraviivainen ja ominaisuuksia on huomattavasti vähemmän kuin V6.6:ssa. Kuvassa 8 on esitettynä V7 työstöratojen luomistyökalut, jotka ovat Import path, Contour, Welding, Welding with touch sensing, Cycles ja Touch sensing.




KUVA 8. Robotmaster V7 työstöratavalikko.

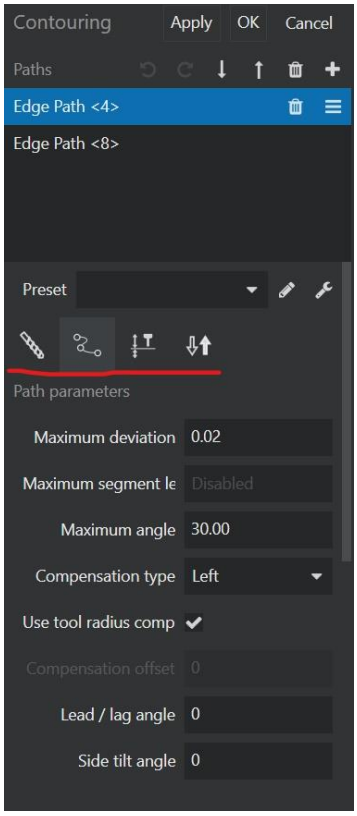
### 4.3.1 Contour

Ratojen luominen esimerkiksi Contour-työkalulla tapahtuu kuvan 9 mukaan valitsemalla työstettävästä kappaleesta haluttu reuna tai muu piirre, jolloin ohjelma luo kuvan mukaisen keltaisen radan valinnan ympärille. Radan ominaisuuksiin pystyy vaikuttamaan esimerkiksi valitsemalla radan hiirellä, jolloin on mahdollista muun muassa kallistaa koko rataa tai sen tiettyjä osia, kääntää rata ympäri ja vaikuttaa sen alku- ja loppupisteisiin.




KUVA 9. Contour-rata.

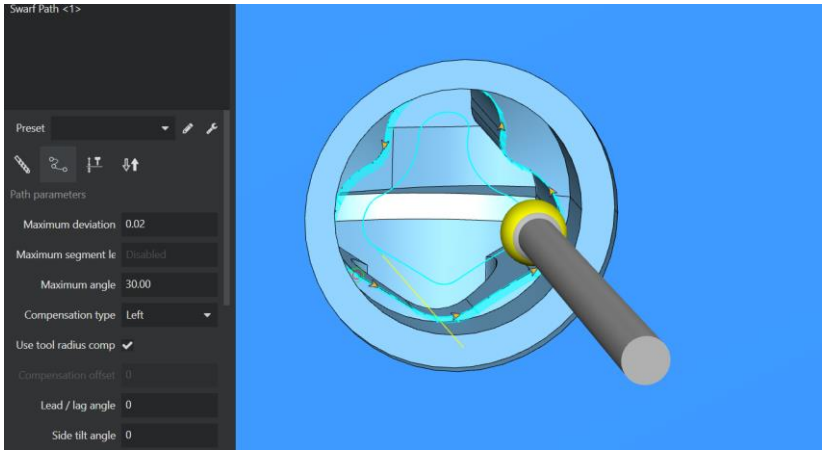
Graafisen radan muokkauksen lisäksi, radan ominaisuuksiin on mahdollista vaikuttaa sen parametrien arvoja muokkaamalla neljästä kuvaan 10 merkitystä valikosta. Valikosta löytyy työstön kannalta oleellisimmat parametrit, kuten nopeus- ja syvyysparametrit, pikaliikkeiden etäisyys ja liiketyyppi, lähestymis- ja poistumisparametrit, sekä työkalun kallistuma, työkalun säteen kompensointi ja sallittu poikkeama radalta.




KUVA 10. Työstöratojen parametrien muokkaaminen.

### 4.3.2 Swarf

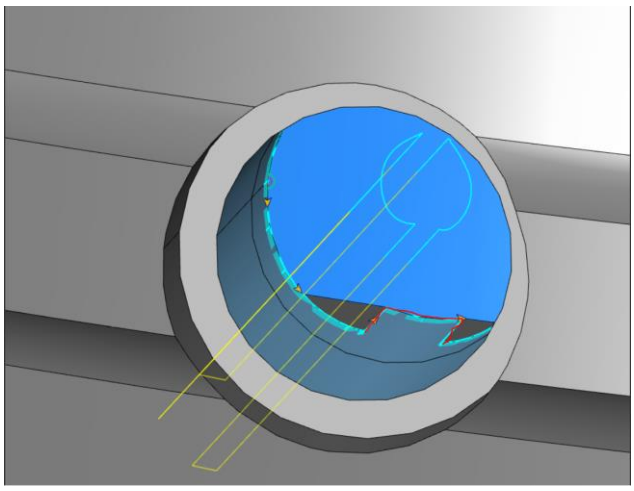
Swarf-radat ovat ratoja, joissa työstö tapahtuu työkalun sivulla työkalun kärjen sijasta. Työstöradan parametrit ovat lähes samat kuin Contour-radassa, mutta työstettävän alueen valitseminen tapahtuu eri tavalla. Kuvassa 11 on esitetty kappaleen sivu, jossa rata halutaan käydä läpi työkalun sivulla. Kyseinen rata on myös mahdollista luoda Contour-toiminnolla. Tällöin radan parametrejä kuten kallistumaa pitää muokata paljon.




KUVA 11. Swarf-radan luonti

Kappaleen geometrialla on suuri vaikutus Swarf-radan luomisessa, sillä työstettävän pinnan valinta tapahtuu valitsemalla ensin jokin kappaleen alue, jonka jälkeen valitaan työstettävä reuna. Kuvassa 12 on esimerkki, jossa valittava alue on ympyrä, joka sisältää punaisella merkityn ulokkeen sekä ylä- että alapinnassa. Swarf luo radan automaattisesti kaikkiin alueen reunoihin siitä huolimatta mitä reunoja on tarpeellista työstää.

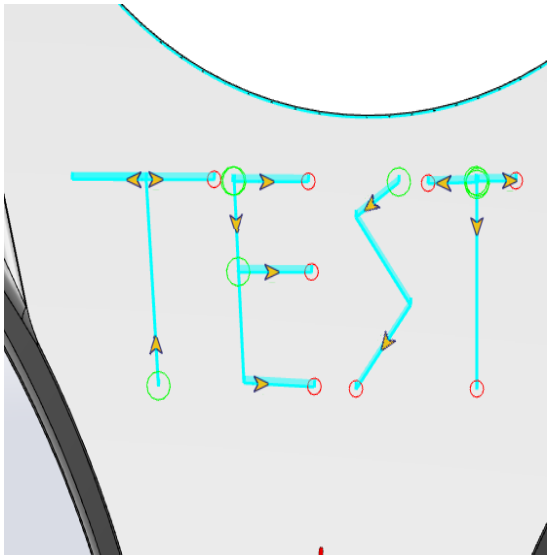
Radan parametrejä ei myöskään pysty muokkaamaan vain tietyistä kohdista, jonka seurauksena rataa täytyy katkoa useampiin osiin ja muokata kaikki radan parametrit erikseen. Kuvassa 12 näkyvät siniset puoliympyrät ovat esimerkiksi luotu Swarf-toiminnolla niin, että rata on katkaistu ulokkeiden kohdalta, jonka seurauksena on syntynyt kaksi Swarf-rataa.



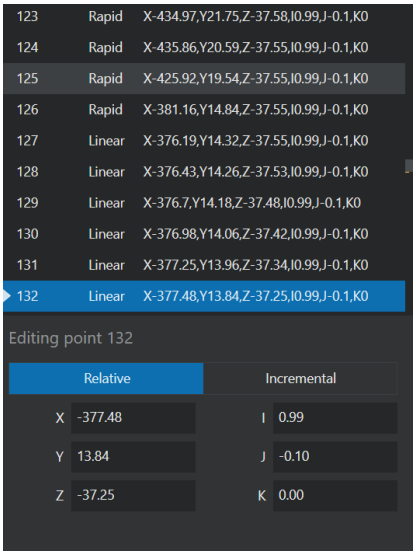
KUVA 12. Muokattu Swarf-rata

### 4.3.3 Manuaaliset radat

Ratoja on mahdollista tehdä myös kuvan 13 mukaisesti pintoihin, jossa ei ole reunoja. Manuaaliset radat tarjoavat paljon vapauksia käyttäjälle, mutta niiden luominen on myös huomattavasti työläämpää kuin Contour- tai Swarf-ratojen luominen. Kaikkien ratojen pisteitä on myös mahdollista muokata manuaalisesti kuvan 14 osoittamalla tavalla. Manuaalisessa muokkauksessa käyttäjä valitsee radan pisteen valikosta ja asettaa uudet koordinaatit millimetreinä. Manuaalisesti pisteiden muokkaus on kuitenkin hyvin työlästä, etenkin jos pisteitä on paljon.



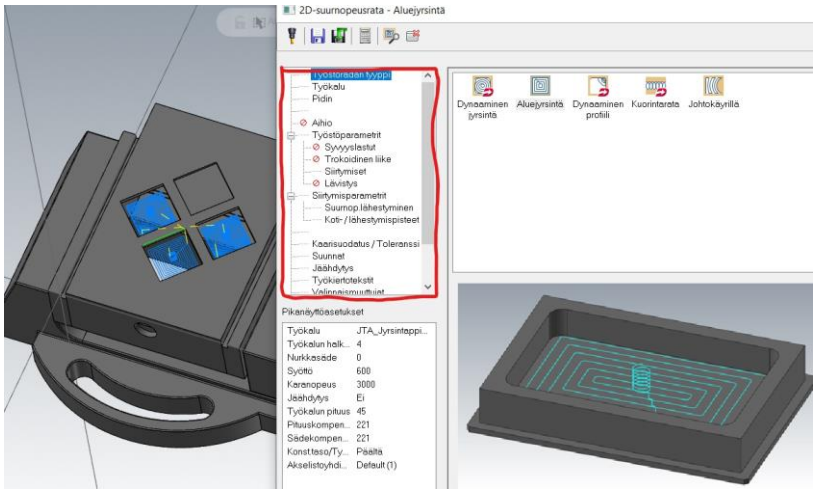
KUVA 13. Manuaalisesti luotu rata




KUVA 14. Pisteen muokkaus

### 4.3.4 Robotmaster V6.6

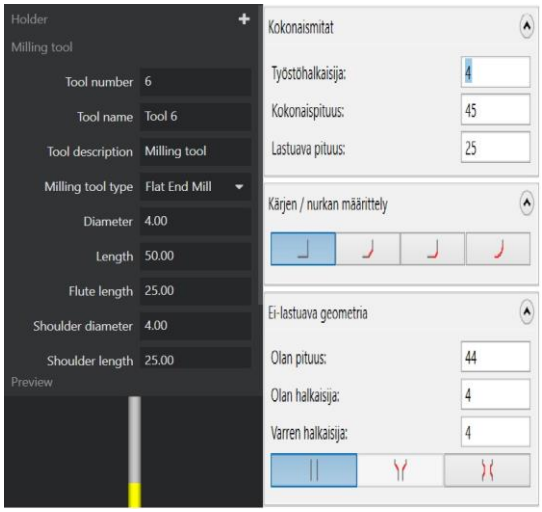
Robotmaster V6.6:ssa ratojen luominen tapahtuu suoraa Mastercamissa. Kuvaan 15 on merkattu valikot, joista 2D-aluejyrsintää koskevia parametrejä voi muokata. Toisin kuin Robotmaster V7:ssä Mastercamissa kaikki työstöradan parametrit säädetään numeerisesti. Tämä edellyttää CAM-ohjelmointiosaamista sekä ymmärrystä koneistuksesta. Kuvia 10 ja 15 verrattaessa, voidaan välittömästi havaita, että Mastercam vaatii huomattavasti enemmän osaamista, kuin Robotmaster V7, sillä säädettäviä parametrejä on huomattavasti enemmän. Kuvaan 15 merkatulla alueella on näkyvissä seitsemän valikkoa, joista löytyy useita eri työkalun liikkeeseen liittyviä parametrejä. Kuvassa 10 vastaavia valikoita on neljä.




KUVA 15. Mastercamin työstöradan parametrivalikko.

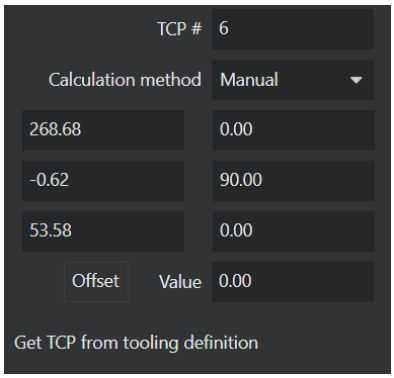
## 4.4 Työkalut ja työkoordinaatistot

Työkalun kuten jyrsintapin luominen tapahtuu molemmissa ohjelmissa samantyylisesti, mutta valikoissa on hieman eroja. Kuvassa 16 on esitetty molempien ohjelmien työkalun muokkausvalikko. Valikossa pystytään muokkaamaan esimerkiksi kuvan 16 jyrsintapin työstöalueen pituutta, kokonaispituutta ja halkaisijaa, sekä muita parametrejä. Kuvassa 16 on lähes samat parametrit, mutta Mastercam tarjoaa myös muita vaihtoehtoja työkalun muokkaamiseen toisin kuin Robotmaster V7.




KUVA 16. Robotmaster V7 ja Mastercamin työkalun muokkaaminen

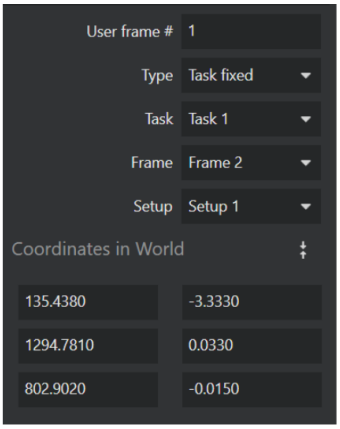
TCP:n luominen työkalun päähän tapahtuu Robotmasterissa kuvan 17 mukaisesti. Työkalupisteen arvot asetetaan joko manuaalisesti tai automaattisesti. Automaattisella valinnalla Robotmaster laskee työkalun geometriaan perustuen arvot TCP:lle. Manuaalisella määrittelyllä arvot pystytään määrittämään itse. Manuaalinen määrittely mahdollistaa, että arvot voidaan hakea suoraa oikealta robotilta, jolloin simulaatio vastaa todellisuutta parhaiten.




KUVA 17. TCP:n arvojen asettaminen

Työkoordinaatiston luominen noudattaa samoja periaatteita kuin työkalun luominen, mutta on yksinkertaisempaa, koska operaatiossa on vähemmän muuttujia. Uusia koordinaatistoja pystyy luomaan molemmissa ohjelmissa osoittamalla koordinaatistolle paikan mallinnetusta ympäristöstä, jonka jälkeen luodulle koordinaatistolle pystyy kertomaan sijainnin maailmakoordinaatistossa. Jotta simulaatiosta saa todellisuutta vastaavan, koordinaattien arvot ovat suositeltavaa hakea oikealta robotilta, jolla koordinaatisto on luotu opettamalla. Koordinaatiston

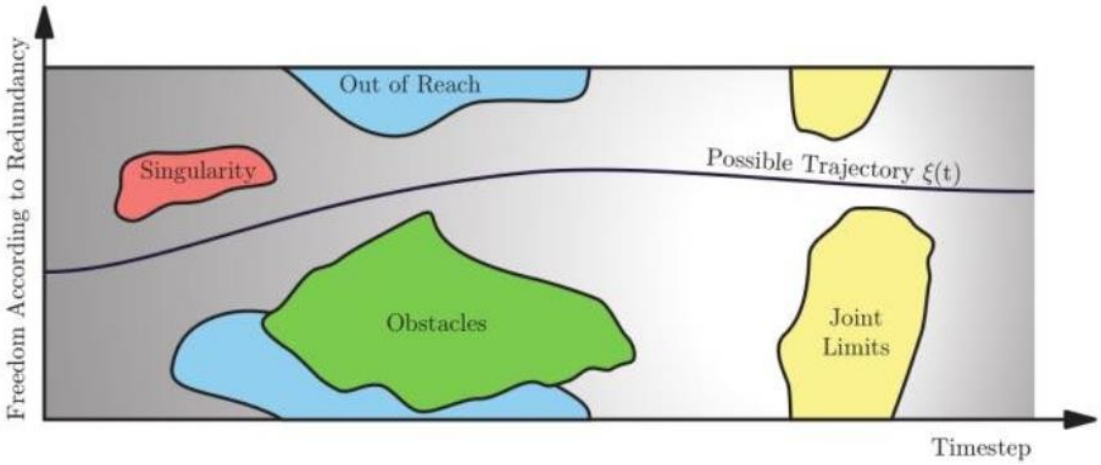
paikoittaminen tapahtuu kuvan 18 mukaisesti samaan tyyliin, kuin TCP:n paikoittaminen jyrsintapille manuaalisesti.




KUVA 18. Työkoordinaatiston sijainnin määrittäminen

## 4.5 Optimointityökalut

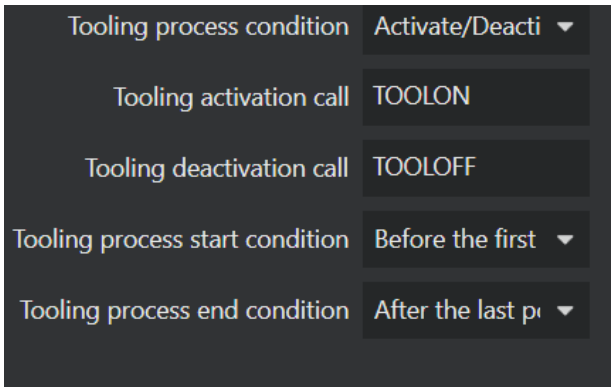
Robotmasterin optimointityökalu laskee robotin liikeradalle mahdolliset törmäykset, singulariteetit, akselien rajat ja ulottuvuusongelmat. Optimointityökalun käyttö tapahtuu graafisesti kuvassa 19 esitettyä viivaa liikuttamalla. Kuvan x-akselilla on kulunut aika, joka on verrannollinen robotin sijaintiin. Y-akselilla voi olla esimerkiksi robotin työkalun kallistuma. Kallistamalla työkalua negatiiviseen tai positiiviseen suuntaan tietyissä kuvan havainnollistamissa paikoissa, on mahdollista luoda toimiva liikerata. Optimointityökalu toimii molemmissa Robotmasterin versioissa ja sisältää samat ominaisuudet.




KUVA 19. Optimointityökalun periaatteellinen toiminta (Petschnigg ym. 2018).

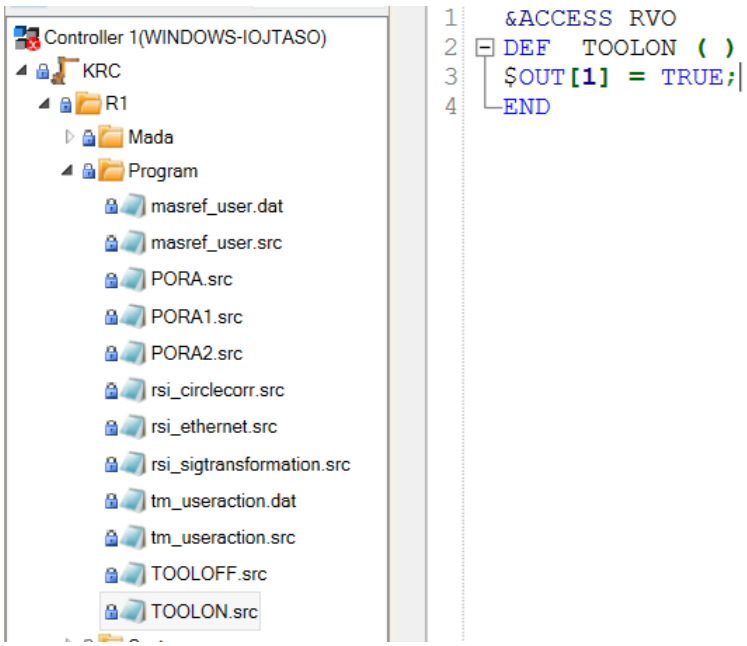
## 4.6 Aliohjelmat

Robotmaster mahdollistaa myös aliohjelmakutsujen kuten työkalun ohjauksen ja työkalunvaihdon luomisen lähes ilman ohjelmointiosaamista. Aliohjelmakutsujen tekeminen Robotmasterissa tapahtuu kuvan 20 mukaisesta valikosta. Kuvassa on esitetty työkalun ohjausta koskeva valikko, jolla työkalu voidaan ohjata päälle. Työkalun ohjaus on sidottu työstöratoihin. Työkalun saa päälle esimerkiksi ennen ensimmäistä kosketusta työstettävään kappaleeseen tai heti ensimmäisen liikkeen jälkeen, joka ei ole pikaliike.




KUVA 20. Aliohjelmakutsujen luominen

Vaikka ohjelmakutsut on mahdollista tehdä ohjelmaan ilman kirjoittamalla ohjelmointia, kutsut eivät kuitenkaan toimi oikealla robotilla pelkällä ohjelmakutsulla. Robotmasterin luomat kutsut ovat tyhjiä, tarvittavia tiedostoja ei löydy robotin ohjaimelta, eikä työkalua ohjaavan toimilaitteen sisään- ja ulostuloja ole osoitettu mihinkään. Nämä toimenpiteet vaativat perustiedot käytössä olevasta robotista sekä sen ohjelmoinnista. Mikäli edellä mainitut toimenpiteet ovat valmiiksi tehty, niin kokematonkin käyttäjä pystyy luomaan toimivia aliohjelmia. Kuvassa 21 on esimerkki, jossa robotin ohjaimelle on luotu aliohjelma, jota voi käyttää Robotmasterissa työkalun ohjaukseen.

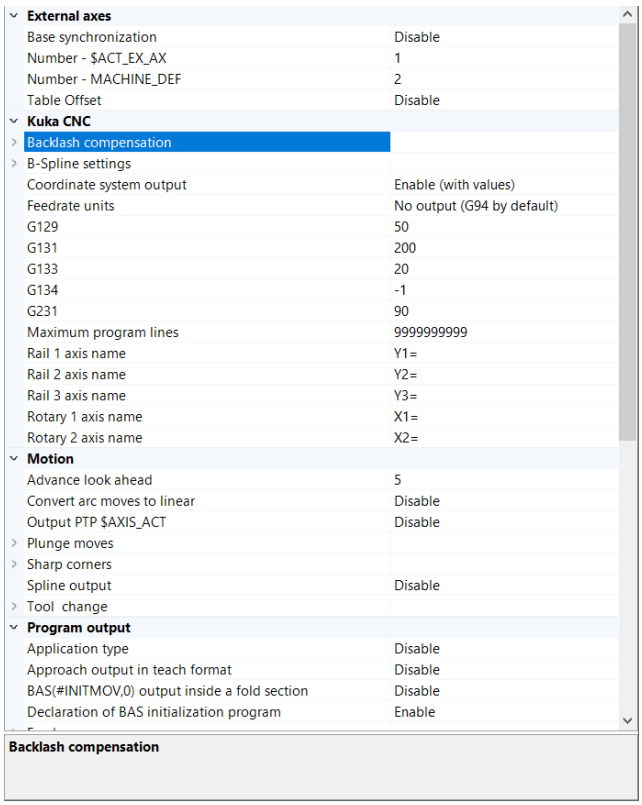



KUVA 21. Aliohjelma TOOLON

## 4.7 Postprosessori

Postprosessori tarkoittaa eräänlaista kääntäjää, joka kääntää CAM-ohjelmiston työstöradat robotilla suoritettavaksi ohjelmaksi. Robotmasterin postprosessorilla työstöradat pystytään kääntämään useiksi erilaisiksi KUKAn ohjelmiksi kuten KRL-ohjelmaksi tai KUKA CNC-ohjelmaksi.

Robotmasterin postprosessorin asetuksia pystyy muokkaamaan molemmissa Robotmasterin versioissa. V6.6 asetuksista pääsee muokkaamaan esimerkiksi seuraavien kokonaisuuksien parametrejä: ulkoiset akselit, KUKA CNC, liike ja ohjelman tulostus. Kuvassa 22 on esitetty V6.6 postprosessorin asetukset, jonka avulla on mahdollista vaikuttaa useisiin parametreihin. Robotmaster V7:ssä postprosessorin asetukset ovat hajautettu useaan eri valikkoon, eivätkä ne ole yhtä kattavat kuin V6.6:ssa. Esimerkiksi kuvassa näkyvät Spline-asetukset puuttuvat.




KUVA 22. Robotmaster V6.6 postprosessorin asetukset

Postprosessori tulostaa oletusasetuksilla kuvan 23 mukaiset liikeparametrit ohjelmaan. Kuvassa nähdään myös työkalun ja koordinaatiston määrittely, sekä liikemääräisen paikotuksen asetukset. Työkalun ja työkoordinaatiston määrittelyjen ollessa heti ohjelman alussa kuvan esittämällä tavalla käyttäjän ei tarvitse puuttua niiden valintaan robotin ohjaimella.

```
;******SETTINGS FOR PTP MOTION*********************
;FOLD PTP $VEL_AXIS AND $ACC_AXIS
FOR I=1 TO 6
    $VEL_AXIS[I] = 50
    $ACC_AXIS[I] = 50
ENDFOR
;ENDFOLD (PTP $VEL_AXIS AND $ACC_AXIS)

;******SETTINGS FOR LIN AND ARC MOTION**************
;FOLD LIN AND ARC MOTION VARIABLES
$VEL.ORI1 = 200
$VEL.ORI2 = 200
$ACC.CP = 3
$ACC.ORI1 = 100
$ACC.ORI2 = 100
;ENDFOLD (LIN AND ARC MOTION VARIABLES)
;******SETTINGS FOR POSITIONING CRITERIA************
;FOLD POSITIONING CRITERIA
  $APO.CDIS = 0.5
  $ORI_TYPE = #VAR
;ENDFOLD (POSITIONING CRITERIA)
$BASE={X 135.4380,Y 1294.7810,Z 802.9020,A -3.3330,B 0.0330,C -0.0150}
$TOOL={X 268.6800,Y -0.6200,Z 53.5800,A 0.0000,B 90.0000,C 0.0000}
$ADVANCE = 5
```

KUVA 23. Postprosessorin generoima KRL ohjelma.

# 5 CASE: LAATIKON AUKOTUS

## 5.1 Vaatimusmäärittely

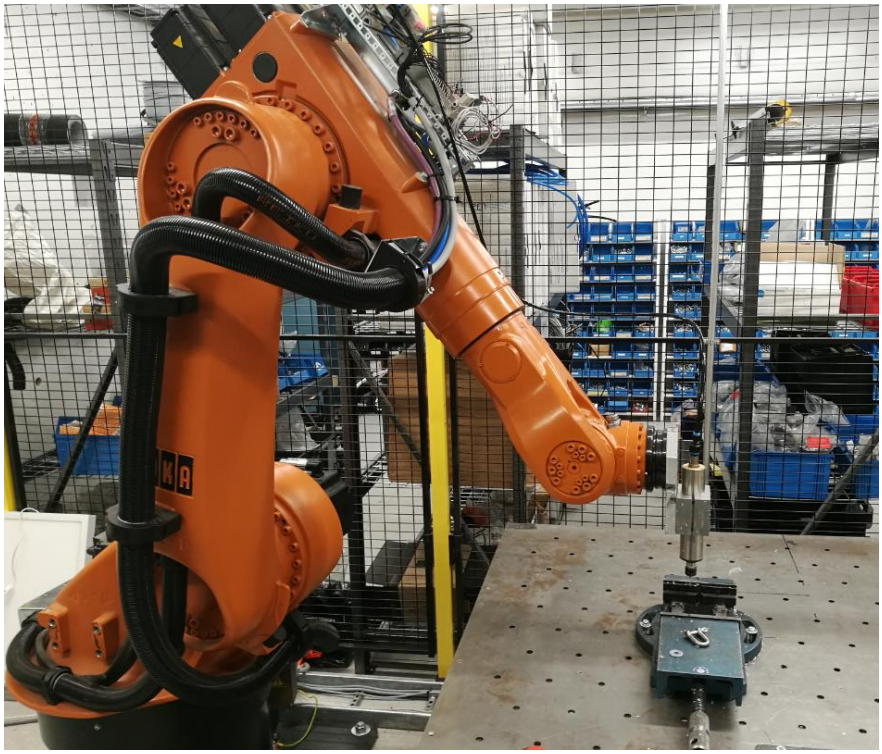
Asiakasyrityksellä on järjestelmä, jolla pystytään jyrsimään robotilla eri kokoisia ja muotoisia reikiä kuvan 24 kaltaisiin muovilaatikoihin. Reikien sijainti ja koko on käyttäjän määritettävissä käyttöpaneelilta. Käyttäjä asettaa paneelilta parametrit, kuten laatikon mitat, reiän sijainnin ja tyypin sekä halutun laatikon sivun. Asiakkaan toiveena on saada käyttöliittymästä visuaalisempi, jotta virheiden määrä laskisi ja käytettävyys paranisi.

KUVA 24. Työstettävä muovilaatikko

Asiakastapauksessa on tarkoitus tutkia Robotmasterin soveltuvuutta kyseiseen tehtävään. Robotmaster tarjoaa visuaalisen tavan tuottaa ohjelma, jossa operaattori voi itse määrittää leikkausparametrit kohtalaisen lyhyellä ohjelmiston opettelulla. Tässä tapauksessa on käytetty molempia Robotmasterin versioita, joista V7 radat ovat luotu leikkaamaan paloja muovista irti, kun taas V6.6 versiolla luoduissa radoissa reiät jyrsitään.

## 5.2 Koeajojen valmistelu

Koeajot suoritettiin kuvan 25 mukaisilla välineillä. Kuvassa oleva robotti on KUKA KR 60 HA-mallinen 6-akselinen nivelvarsirobotti. Robotin päässä oleva työkalu on paineilmakara, jonka päähän on kiinnitetty teräväksi hiottu ruuvi. Kuvassa näkyvä ruuvipenkki on kiinnitetty pöytään kahdella pultilla. Ruuvipenkin tarkoitus työssä on toimia kiinnittimenä muovilaatikolle.



KUVA 25. Testiympäristö

Paineilmakaran päässä oleva ruuvi on niin sanottu opetustyökalu, jolla robotille voidaan opettaa tarkasti esimerkiksi työkoordinaatistoja. Opetustyökalu on hiottu teräväksi, jotta sen opettaminen kappaleessa 3.2 esitetyllä neljän pisteen menetelmällä olisi mahdollisimman tarkkaa. Terävä työkalu on helpompi ajaa referenssipisteeseen tarkasti, joka pienentää opetuksessa tapahtuvaa virhettä.

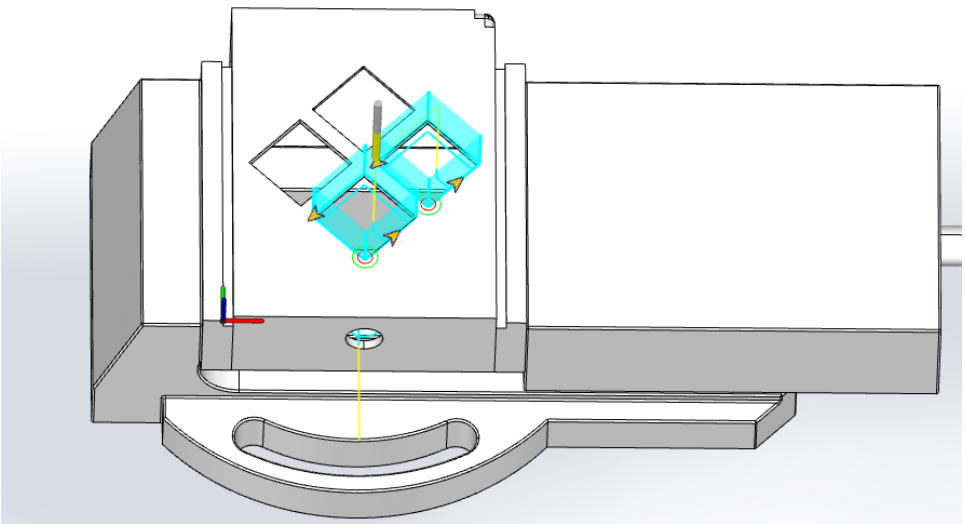
Opetustyökalua on käytetty työssä robotin työkoordinaatiston opettamiseen. Koordinaatisto on opetettu ruuvipenkin reunaan käyttämällä kappaleessa 3.3 esiteltyä kolmen pisteen menetelmää. Ruuvipenkki on kiinnitetty pöytään pulteilla, koska on äärimmäisen tärkeää, että ruuvipenkki ei pääse liikkumaan missään

kohtaa testiajoja tai testiajojen välissä. Mikäli ruuvipenkkiä liikutetaan koordinaatiston opetuksen jälkeen, liikepisteet eivät enää täsmää kappaleen sijaintiin.

Jyrsintapiksi valikoitui tässä tapauksessa halkaisijaltaan neljä millimetriä ja pituudeltaan 45 millimetriä pitkä jyrsintappi. Jyrsintappia ei opetettu robotille, koska jyrsintappi ja ruuvi ovat geometrialtaan samankaltaisia. Ainoastaan työkalun pituus muuttui merkittävästi koeajoja ajatellen. Tästä syystä uuden työkalun opettamisen sijaan, työkalun pituusakselin suuntaista koordinaattia siirrettiin manuaalisesti viisi millimetriä kohti karaa. Toimenpiteen jälkeen jyrsintapin TCP tarkistettiin ajamalla työkalu opetetun työkoordinaatiston origoon. Tällöin robotin käsiohjain näytti TCP:n koordinaateiksi (0,0,0), joten siirto oli onnistunut.

## 5.3 Ratojen luominen

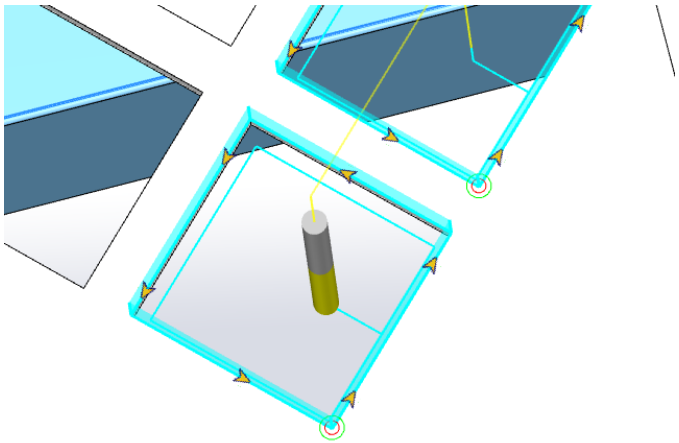
Ratojen luominen Robotmaster V7:llä tapahtui käyttämällä Contour-toimintoa. Laatikkoon piirrettiin kaksi neliön muotoista rataa sekä yksi ympyrän muotoinen rata, jotka ovat esitetty kuvassa 26 sinisellä viivalla. Kuvassa näkyvät keltaiset viivat ovat pikaliikeratoja eli lähestymis- ja poistumisradat. Kuvassa näkyy myös ruuvipenkki, jonka vasemmanpuoleisessa leuassa on koordinaatisto, jonka suhteen kaikki liikkeet tehdään.



KUVA 26. Laatikon Contour-radat

Työstön helpottamiseksi radalle on myös määrätty muita parametrejä Robotmasterilla. Kappaleessa 4.3.1 esitetyssä kuvassa 10 olevista valikoista työstölle valittiin sopivat asetukset. Syöttönopeudeksi valikoitui viisi millimetriä sekunnissa, joka oli aikaisemmissa, tämän tutkimuksen ulkopuolisissa testeissä havaittu sopivaksi nopeudeksi. Porautumissyvyydeksi valittiin kuusi millimetriä, koska materiaalin paksuus oli noin neljä millimetriä. Työkalun säteen kompensoinniksi valikoitui vasen puoli, koska työstörata kiertää vastapäivää.

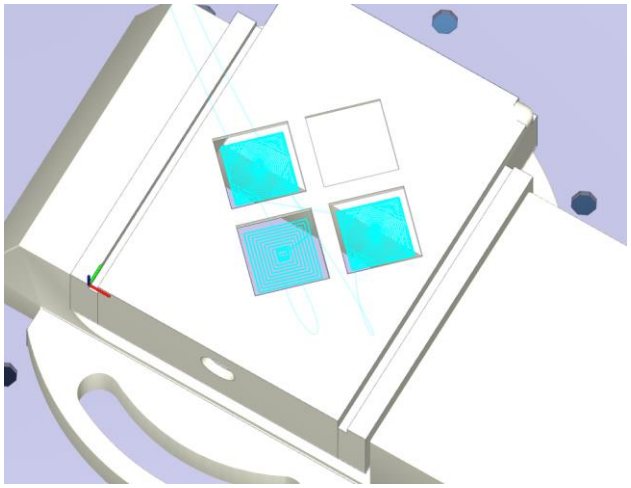
Lähestymisparametreiksi valittiin lähestyminen radan keskipisteestä, jolloin työkalu porataan ensin neliön keskipisteeseen. Keskipisteestä lähestytään varsinaista leikattavaa reunaa kuvassa 27 esitetyllä tavalla. Kaikkein loogisin poistumisrata tässä tapauksessa oli työkalun pituusakselin suuntainen, koska työstöradat ovat kohtisuorassa kappaletta kohden. Poistumisetäisyydeksi valittiin 30 millimetriä työstettävästä pinnasta.



KUVA 27. Lähestyminen radan keskipisteestä

Aukottamista testattiin myös Robotmasterin versiolla 6.6. Tässä tapauksessa reiät luotiin jyrsimällä ne kokonaisuudessaan, eikä leikkaamalla reunoja. Ratojen parametrit ovat muuten samat kuin leikkaustestissä. Testin tavoite oli selvittää, pystyykö KUKAn ohjain lukemaan KRL-ohjelmatiedostoja, joissa liikepisteitä on tuhansia. Tässä testissä liikepisteitä muodostui 8673 kappaletta, jotka aiheutettiin luomalla osalle radoista epärealistiset toleranssit.

Kuvassa 28 on esitetty jyrsintäradat, jotka ovat luotu Mastercamilla ja viety Robotmasteriin. Kuvassa alin rata on tavallinen aluejyrsintä, jonka toleranssi on 0.025 millimetriä, vasemmanpuoleinen rata on dynaaminen aluejyrsintä samalla toleranssilla ja oikeanpuoleinen rata on dynaaminen aluejyrsintä yhden millin toleranssilla. Ohjelmasta noin 90% muodostuu kahdesta ensimmäisestä radasta.

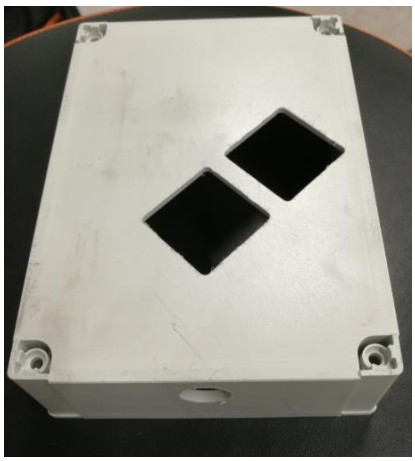


KUVA 28. Jyrsintäradat

## 5.4 Tulokset

### 5.4.1 Leikkaus

V7 Koeajojen tuloksista, jotka ovat esitetty kuvassa 29 nähdään, että tällainen ohjelmisto on toimiva ratkaisu ongelmaan. Robotmaster V7 tarjoaa visuaaliseen käyttöliittymään verrattavissa olevan ratkaisun, jonka käyttäminen ei vaadi kovin paljon koulutusta. Robotmaster myös tarjoaa asiakkaalle mahdollisuuden tehdä minkälaisia reikiä tahansa laatikoihin. Ratojen luominen tietysti edellyttää CAD-mallin, johon reiät ovat piirrettynä, kuten tässäkin tapauksessa on ollut. Tämän lisäksi asiakkaan pitää hallita robotiikan alkeet, sillä ohjelmat täytyy siirtää robotin ohjaimelle PC:ltä.



KUVA 29. Koeajon tulokset

Vastaavan järjestelmän toteuttamista muilla menetelmillä kannattaa kuitenkin harkita, sillä kyseessä on kohtalaisen yksinkertainen robottisovellus. Robotmasterin hinta on noin 10 000 – 40 000 euroa, riippuen mitä ominaisuuksia tarvitsee (Iisac Maw 2019; liite 1). Asiakkaan tarpeiden mukainen robottisovellus on myös täysin mahdollista toteuttaa ohjelmalla, jossa reikien paikat ja muodot ovat määritetty robotin ohjelmassa laskennallisesti. Asiakkaan nykyinen järjestelmä toimii tällä tavalla, joten nykyistä käyttöliittymää kehittämällä visuaalisempaan suuntaan on mahdollista saada täysin asiakkaan tarpeita vastaava robottisolu.

Käyttöliittymän kehittäminen ei kuitenkaan ole ilmaista. Esimerkiksi jos oletetaan, että käyttöliittymän kehittäminen maksaa yritykselle 80 euroa tunnilta, niin Robotmaster muodostuu kannattavaksi asiakkaalle, mikäli käyttöliittymän kehittämiseen käytetään yli 250 tuntia. Tuntimäärä on laskettu yhtälön 1 mukaisesti. Laskelmassa on oletettu, että Robotmaster maksaa 20 000 euroa ja työntekijöiden kouluttamiseen ohjelmiston käytössä ei kuluisi merkittävästi aikaa.

$$\frac{\text{Ohjelmiston hankintahinta}}{\text{Kehityksen tuntihinta}} = \frac{20000\ €}{80\ €/h} = 250\ h \qquad (1)$$

### 5.4.2 Jyrsintä

Robotmaster V6.6:lla tehty jyrsintätesti onnistui myös hyvin (kuva 30). Ennakkooletuksena oli, että toleranssia kasvattaessa liikepisteet ovat jo niin lähellä toisi-

aan, että KUKAn Advance run ominaisuus aiheuttaisi työstössä epätasaista liikettä. Jyrsintä onnistui hyvin CNC-ohjattuna ja KRL-ohjelmana, vaikka NC-ohjelmassa ilmenikin useita ongelmia. Postprosessorista aiheutuvat ongelmat olivat korjattavissa muokkaamalla postprosessorin asetuksia tai muokkaamalla postprosessorin tuottamaa ohjelmaa manuaalisesti.

KUVA 30. V6.6:lla suoritettu jyrsintä

NC-ohjelmaksi kääntämisessä pitää huomioida muun muassa työkalujen ja työkoordinaatistojen erilainen määrittely sekä mekaanisten välysten kompensointi. Robotmasterin kääntämän ohjelman alku on kuvan 31 mukainen G-koodi. Robotmaster lisää kuvan koodin jokaisen NC-ohjelman alkuun vakioasetuksilla, josta aiheutuu ongelmia kuvassa esitetyn ohjelman riveillä 17 ja 18. Rivillä 16 on Robotmasterissa ohjelmoitu robotin kotipiste, jonka Robotmaster suorittaa simulaatiossa aina. Rivit 17 ja 18 puolestaan ovat ne pisteet, jossa robotti suorittaa mekaanisten välysten kompensoinnin. Ongelmaksi muodostuu se, että robotti suorittaa ensin kotiajon, jonka jälkeen robotin 1-akseli kääntyy 90 asetetta todella suurella nopeudella. Robotmaster ei näytä mekaanisten välysten kompensointia simulaatiossa, jolloin robotti pyrkii ajamaan itsensä työstöradalle 1-akselin ollessa 90 asetta väärässä asennossa.

```
4   N1 G90
5   N2 G129 = 50
6   N3 G231 = 90
7   N4 G131 = 200
8   N5 G133 = 20
9   N6 G134 = 20
10  #HSC [BSPLINE PATH_DEV 0.1000 TRACK_DEV 0.5000]
11  N7 #CS OFF ALL
12  N8 #CS DEF[1][135.4380,1294.7810,802.9020,-0.0150,0.0330,-3.3330]
13  N9 #CS ON[1]
14  N10 M3 S3000
15  N11 #MCS ON
16  N12 G0 X-90.0000 Y-90.0000 Z90.0000 A0.0000 B0.0000 C0.0000
17  G1 X-1 F20000
18  G1 X0
```

KUVA 31. Robotmasterin luoman NC-ohjelman alku

Työkalun ja työkoordinaatiston määrittely ei tapahdu NC-ohjelmassa samalla tavalla kuin KRL-ohjelmassa, joka on esitetty kappaleessa 4.7 kuvassa 23. NC-ohjemassa ei ole ollenkaan työkalun määrittelyä, kuten KRL ohjelmassa. Tällöin jää käyttäjän vastuulle pitää huoli siitä, että oikea työkalu on robotin käsiohjaimella valittuna CNC-ohjausta kutsuessa.

Työkoordinaatiston määrittely tulee NC-ohjelmaan automaattisesti, joka näkyy kuvan 31 ohjelmassa rivillä 12. Tästä määrittelystä voi olla myös haittaa. Jos käyttäjä ei ymmärrä CNC-ohjausta kutsuessa valita robotin koordinaatistoksi nollakoordinaatistoa, niin rivin 12 koordinaatit lisätään sen koordinaatiston päälle, joka oli CNC-ohjausta kutsuessa valittuna. Tällöin törmäyksen riski on suuri, sillä robotti voi käytännössä ajaa ihan mihin tahansa.

# 6 CASE: JÄYSTEENPOISTO

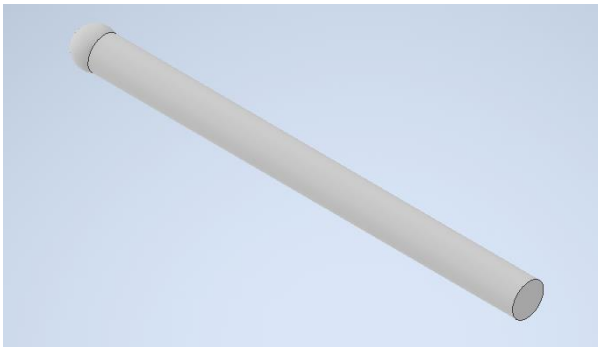
## 6.1 Vaatimusmäärittely

Asiakkaan tavoitteena on pystyä automatisoimaan valurautaisen kappaleen jäysteenpoisto. Kyseinen työ halutaan automatisoida, koska työ on uuvuttavaa ja käsin tehdessä työturvallisuusriski. Kappaleita on eri kokoisia ja niiden yksityiskohdat vaihtelevat. Kappaleista pitäisi pystyä myös poistamaan jäystettä kappaleen sisäpuolelta, jossa jäysteisiä reunoja on paikoittain myös syvissä onkaloissa.

## 6.2 Koeajojen valmistelu

Kappaleen monimutkaisuuden takia tämän kappaleen ratojen luomiseen käytettiin ainoastaan Robotmasterin versiota V6.6 ja Mastercamia. Robotmaster V7:ssä ei ole kappaleen työstön kannalta oleellisia ominaisuuksia, kuten risteävien reikien työstöä. Kappaleen sisäpuolella työstettävät pinnat ovat erittäin haasteellisia luoda Robotmaster V7:llä, jos edes mahdollisia tässä tapauksessa.

Kappaleen käsin työstöön käytetään tällä hetkellä useita eri työkaluja. Työstöön sopivat välineet robotille, kuten servo-ohjattu kara, työkalun pikavaihtaja ja työstöpäät olisivat olleet äärimmäisen iso investointi, joten työstöä päädyttiin simuloimaan 3D-tulostetulla pallopäisellä koneviilalla. Työkalu oli kuvan 32 mukainen ja se tulostettiin muovista, joten tarkoituksena ei ollut ottaa kontaktia työstettävään kappaleeseen.



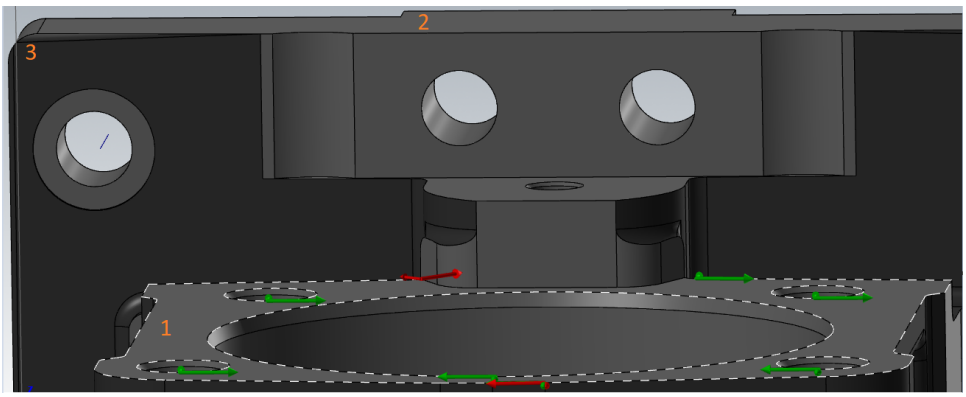
KUVA 32. CAD-malli tulostetusta työkalusta

Kaikki muut välineet pysyivät samana kuin laatikon aukotuksessa. Kappaleen sai kiinnitettyä samaan ruuvipenkkiin tukevasti eikä sitä tarvinnut siirtää. Työkalun pituus kasvoi jyrsintappiin verrattuna 40 millimetriä. TCP:n siirto suoritettiin tässäkin testissä manuaalisesti, jonka jälkeen siirron onnistuminen tarkistettiin työkoordinaatiston origossa. Tarkistukset siis suoritettiin täysin vastaavalla tavalla kuin laatikon aukotuksessa.

### 6.3 Ratojen luominen

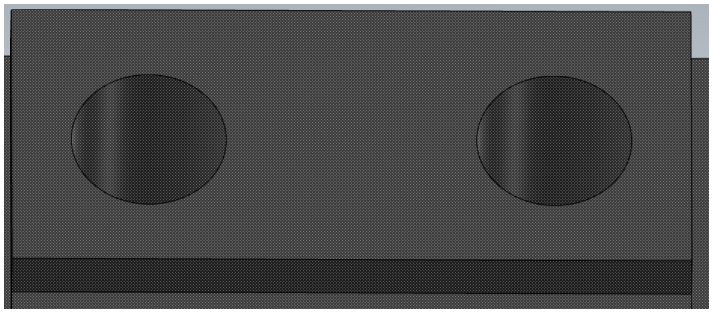
Radat luotiin käyttämällä Mastercamin jäysteenpoisto-ominaisuutta. Kappaleen haastavuuden takia koko kappaletta ei käsitelty. Sen sijaan kappaleesta valittiin viisi erilaista työstökohdetta, jotka olivat haastavuudeltaan erilaisia. Työstöratoja luotiin kappaleen kaikille sivuille ja ne ovat muodoiltaan suurimmaksi osaksi erilaisia. Työkaluksi määritettiin Mastercamissa kuulapääjyrsin, jonka pään halkaisija oli oikean työkalun halkaisijaa millin suurempi. Halkaisija määritettiin todellisuutta suuremmaksi, jotta työkalu ei ottaisi työstettävään kappaleeseen kiinni. Tästä syystä myös jäysteteenpoiston syvyydeksi määriteltiin kaikissa radoissa 0.01 millimetriä.

Ensimmäinen työstöpinta on merkitty kuvaan 33 numerolla 1. Tästä pinnasta käytiin läpi kaikki pyöreät reiät sekä alueen ulkoreunat. Seuraava työstettävä alue on kuvassa numerolla kaksi. Alue kaksi sisälsi suoria viivoja, mutkia, sekä numerolla kolme merkityn pyöristyksen. Työstöpintojen valinta tehtiin manuaalisesti, kuten myös työstettävien reunojen määrittely. Mastercamilla työstettävät reunat pystytään valitsemaan manuaalisesti klikkaamalla reunoja, jolloin useita geometrian kohtia pystyy ketjuttamaan kuvassa 33 näkyvällä katkoviivalla.



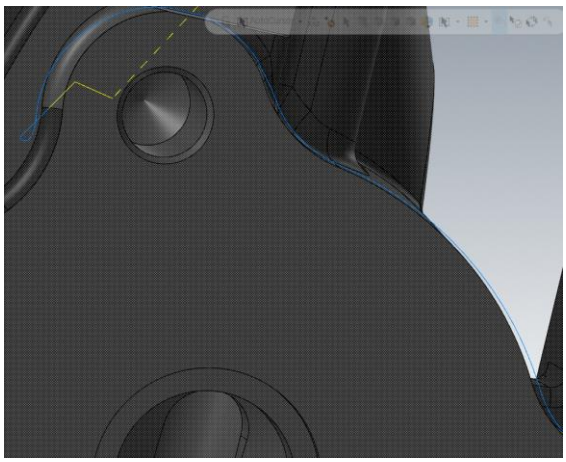

KUVA 33. Työstettävät alueet yksi ja kaksi.

Työstöratoja luotiin myös robotista katsottuna kappaleen takasivulle, jossa jäysteenpoisto suoritettiin kuulapään sivulla eikä kärjellä. Kuvan 34 esimerkissä radat luotiin niin, että robotti käy läpi kaikki suorakaiteen ulkoreunat. Työkalun kallistus määriteltiin Mastercamissa 20 asteeseen pystyakselin suhteen, jolloin Robotmasterilla pystyttiin konfiguroimaan sellaiset akselien asennot, että työstö oli mahdollista suorittaa.



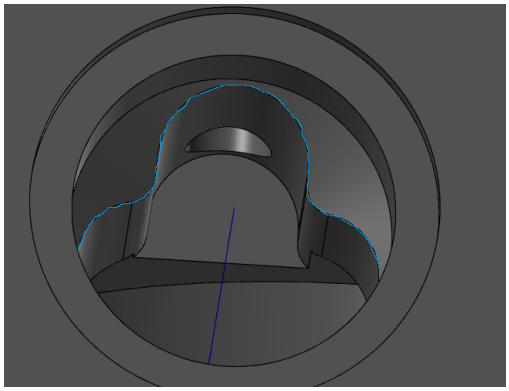
KUVA 34. Kappaleen takasivu

Kuvassa 35 on esitetty osa kappaleen oikeaan reunaan luodusta radasta. Kuvassa sinisenä näkyvä kaarevan radan tarkoituksena oli testata, kuinka hyvin Robotmasterilla luotu ohjelma pystyy ajamaan kaarevan radan. Kappaleen pinnat ja radat valittiin manuaalisesti, jolloin havaittiin, että kappale on piirretty CAM-ohjelmoinnin kannalta huonosti. Reunapintojen väliin jää helposti hyvin pieniä valittavia alueita, joita ei havaitse kuin tarkentamalla erittäin lähelle reunaa.



KUVA 35. Kappaleen oikea reuna

Viimeinen rata oli ehdottomasti kaikkein haastavin, sekä luomista että työstöä ajatellen. Kuvaan 36 on merkattu sinisellä viivalla ensimmäinen työstettävä reuna. Toinen työstettävä reuna on kuvassa näkyvän reiän sisäpuolella oleva reikä. Kappaleessa on useita vastaavia työstettäviä paikkoja, joista osa on vielä huomattavasti haastavampia. Radan luomisessa hyödynnettiin Mastercamin työkalun akselin ohjauksen asetuksia, joilla työkalun suunta määritettiin kuvassa näkyvän tummansinisen viivan suuntaiseksi. Viiva on kohtisuorassa reikään nähden.

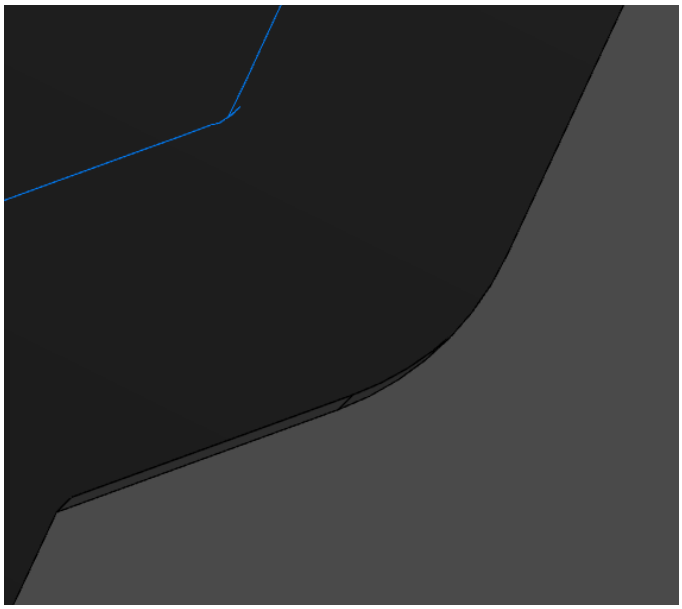


KUVA 36. Viimeinen työstettävä alue

### 6.4 Tulokset

Ensimmäisestä työstettävästä alueesta reikien työstäminen sujui ongelmitta KRL- ja CNC-ohjelmana. Alueen ulkoreunojen ajaminen KRL-ohjelmana ei kuitenkaan onnistunut täysin, sillä robotti pysähtyi hetkeksi ulkoreunassa olevien

pienten mutkien kohdalla. Selitys tälle löytyy KUKAn Advance run ominaisuudesta sekä CAM-ohjelmasta. Kuvassa 37 on tarkennettu ongelmakohtaan, josta nähdään, että sinisellä piirretty työstörata ei ole täysin kaareva. Radassa näkyy pieni yliajo mutkan kohdalla.



KUVA 37. Yliajo CAM-radassa.

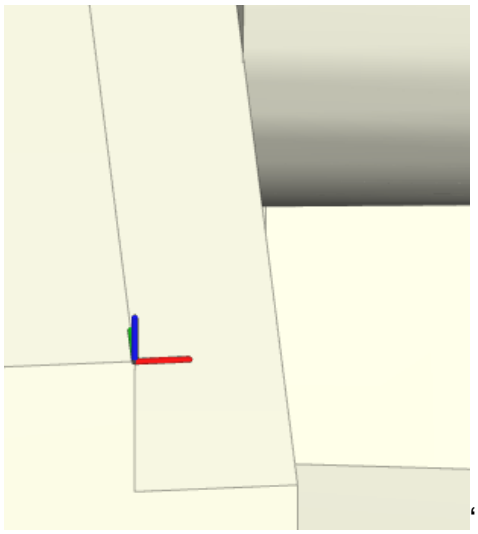
KRL-ohjelmaa suorittaessa tämä näkyy huomattavasti selvemmin kuin NC-ohjelmaa suorittaessa, sillä robotin ohjain ennakoi vain viisi liikepistettä eteenpäin KRL-ohjelmassa. Robotti pyrkii ajamaan mutkassa hieman edestakaisin, joka näyttää aivan lyhyeltä pysähtymiseltä tai tarkalta paikoittamiselta. NC-ohjelmassa robotin ohjain puolestaan lukee koko mutkan jo etukäteen läpi, jolloin yliajoa ei pysty ainakaan visuaalisesti havaitsemaan. Simulaatiossa ongelma ei ole helposti havaittavissa.

Työstörata kaksi, joka on merkattu kuvaan 33 ja kappaleen takasivu kuvassa 34 onnistuivat ongelmitta. Robotista katsottuna kappaleen takasivulla oleva suorakaide vaati akselien konfigurointia Robotmasterissa hieman enemmän kuin muut työstettävät pinnat. Robotmasterin optimointityökaluilla toimenpide osoittautui kuitenkin varsin helpoksi. Käytännössä optimointi tapahtui kappaleessa 4.5 esitetyllä tavalla.

Kappaleen oikealla ja vasemmalla reunalla olleet radat, jotka ovat esitetty kuvissa 35 ja 36 eivät onnistuneet. Ongelmaksi molemmilla sivuilla muodostui robotin törmääminen kappaleeseen. Kappaleen oikealla sivulla, joka on esitetty kuvassa 35, robotti pyrki ajamaan työkalua liian syvälle kappaleeseen. Tämän tyyppinen vika voi johtua useista eri syistä, kuten väärän mittaisesta työkalusta tai kappaleesta, ohjelmointivirheestä tai virheellisestä työkalu- tai koordinaatistodatasta.

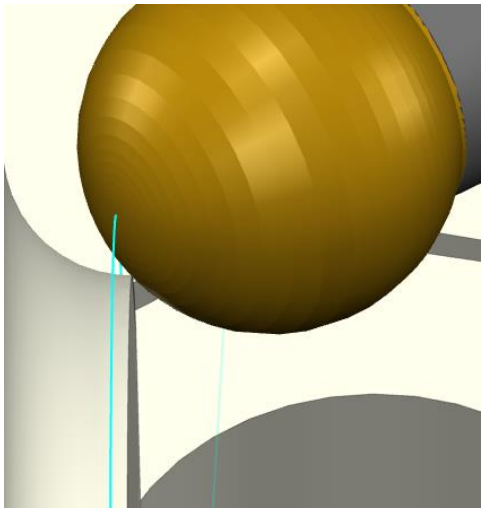
Ensimmäisenä tarkistettiin työkalupisteen sijainti ja kappaleen mitat. Työkalupiste tarkistettiin ajamalla työkalun kärki työkoordinaatiston origoon, josta välittömästi havaittiin työkalupisteen olevan oikeassa kohdassa. Seuraavaksi kappale mitattiin sivuttaissuunnassa, jotta se on varmasti samankokoinen kuin CAM-ohjelmassa käytetty malli. Mitta saatiin otettua työntömitalla koneistettuja pintoja vasten. Mallien koot vastasivat, joten seuraavaksi oli johdonmukaista tarkistaa, että työkoordinaatisto on kaikkialla missä sitä on käytetty samassa pisteessä.

Työkalupisteen tarkistuksen perusteella tiedettiin, missä koordinaatiston origo on robotin ohjaimen mukaan, joten tarkistettavaksi jäi ohjelmisto. Robotmasterissa ja Mastercamissa koordinaatisto oli ruuvipenkin leuan kulmassa kuvan 38 mukaisesti. Kuvasta 38 huomataan myös, että kappale on kiinni ruuvipenkissä. Kaikki testit vastasivat oikeata robottisolua täysin, joten seuraavaksi tarkisteltiin ohjelmaa ja törmäyspisteitä.



KUVA 38. Työkoordinaatiston sijainti ruuvipenkissä

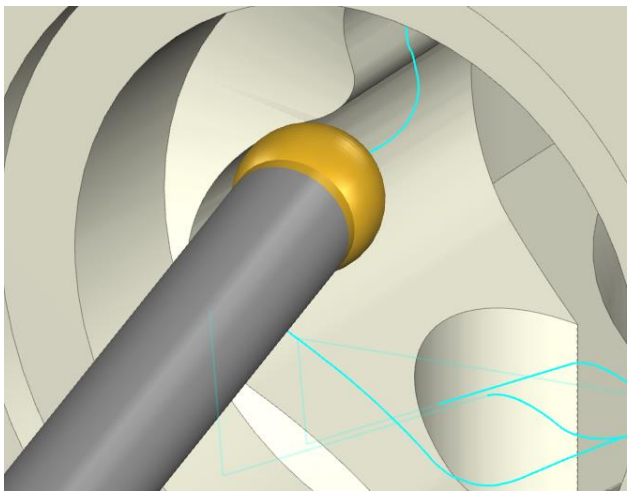
Törmäyspistettä ja kappaleen mittoja verratessa huomattiin, että kyseiseen pisteeseen ajaminen ei ole mahdollista ilman törmäystä. Pisteet ovat viisi millimetriä liian syvällä. Ensimmäinen oletus virheen juurisyystä oli CAM-ratojen luomisessa tapahtunut virhe. Robotmasterin simulaatiota tarkastellessa työkalu liikkuu kuvan 39 mukaan kuitenkin oikeassa kohdassa rataa. Robotmaster on myös muistettu synkronoida Mastercamin kanssa, joten ratojen pitäisi olla yhdenpitävät.



KUVA 39. Oikeassa reunassa oleva rata simulaatiossa.

Varsinaista juurisyytä ei tässä tapauksessa löytynyt. Rata haluttiin siitä huolimatta ajaa, joten työkalun pituusakselin suuntaista TCP:n koordinaattia siirrettiin viisi millimetriä pidemmälle, jolloin rata pystyttiin ajamaan. TCP:n siirron jälkeen radan ajo onnistui ilman ongelmia.

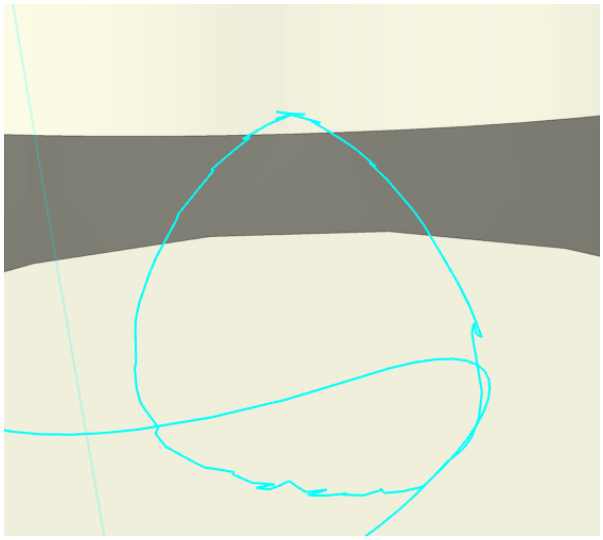
Viimeisin rata, joka on aiemmin esitetty kuvassa 36 ei myöskään toiminut toivotulla tavalla. Simulaatiossa työkalu pysyi kuvan 40 esittämällä tavalla reunojen ulkopuolella, mutta todellisuudessa rata poikkesi kuvan esittämässä kohdassa reunan sisäpuolelta, aiheuttaen törmäyksen. Reikä on koneistettu ja mittausten perusteella samankokoinen ja -muotoinen kuin CAD mallissa.



KUVA 40. Vasemman reunan työstöradat simulaatiossa

Reiän sisäpuolella olevan reiän ajossa ei aiheutunut törmäyksiä, mutta radassa oli paljon samantyyppistä pysähtyvää liikettä kuin ensimmäisessä radassa. Ensimmäisenä testattiin, pystyykö radan ajamaan tasaisesti CNC-tilassa. CNC-puolella ongelma ei hävinnyt, joten ruvettiin tarkastelemaan Robotmasteriin vietyä CAM-ohjelmaa.

Robotmasterissa oleva rata on esitetty kuvassa 41 ja siitä havaitaan heti, että radassa on paljon poikkeamia. Radan pitäisi olla yksi tasainen viiva, mutta siinä on kuvan mukaisia notkahduksia. Radan parametreista pyrittiin nostamaan toleranssien arvoja, saamatta haluttua lopputulosta. Tässä kohtaa ajon epäonnistumiseen vaikuttaa suoraan CAM-osaamisen puute.



KUVA 41. Epäonnistunut työstörata

# 7 INVESTOINTI

Ohjelmiston investoinnin kannattavuutta pystytään tarkastelemaan useasta eri näkökulmista. Tässä tapauksessa keskitytään tarkastelemaan investoinnin kannattavuutta asiakkaan ja JTA Connectionin näkökulmasta. Laskemissa huomioidaan myös, että Robotmaster on ennen kaikkea suunniteltu ohjelmistoksi, jonka tarkoituksena on nopeuttaa suunnittelua ja mahdollistaa robottiohjelmien luominen ilman laajaa robottiosaamista. Laskennassa käytetyt suunnittelukustannukset ovat arviota. Ohjelmistoa koskevat arviot pohjautuvat lähteissä mainittuun artikkeliin The What Why and How of Industrial Robot Simulation Software for Offline Programming ja liitteeseen 1.

Oletukset laskennalle:

- Omakustannehinta suunnittelulle: 65 €/h
- Suunnittelun hinta asiakkaalle ALV 0%: 80 €/h
- Ohjelmiston hinta: 20 000 €
- Ohjelmistoa ei tarvitse päivittää 3 vuoden aikana

Robotmasterin kotisivujen mukaan Robotmaster vähentää offline-ohjelmointi aikaa jopa 70% (Success stories n.d). Tulokseen vaikuttaa todella paljon kappaleen muotojen haastavuus ja ohjelmoijan kokemus ohjelmoinnista eri tavoilla. Tästä syystä suoritettiin testejä, joissa ohjelmoitiin kuvassa 33 näkyvien katkoviivojen mukainen rata. Ohjelmointi tehtiin ensin KUKAn käsiohjaimella, jonka jälkeen sama rata ohjelmoitiin Robotmaster V6.6:lla sekä Mastercamilla. Ohjelmointia suoritettaessa suoritusten ajat mitattiin sekuntikellolla, joka pysäytettiin, mikäli ratojen luomisessa syntyi tietotaidosta johtuvia ongelmia tai teknisiä ongelmia. Lopputuloksena Robotmaster oli noin 55 % nopeampi.

Tehdään oletus, että ohjelmoitavat kappaleet ovat haastavuudeltaan sellaisia, että niiden ohjelmointi vie noin 20 tuntia suunnittelun resursseja ilman Robotmasteria. Oletetaan, että Robotmaster on 55% nopeampi, jolloin suunnittelun resursseja säästetään yhtälön 2 mukaan kappaletta kohti

$$20\ h - (20\ h * 0.55) = 9\ h. \quad (2)$$

Yhtälön 2 mukaan 20 tuntia resursseja kuluttava kappale kuluttaisi Robotmasterilla vain 9 tuntia. Suunnittelun omakustannehintaa ajatellen Robotmasterin avulla säästöä per kappale kertyy yhtälön 3 nojalla

$$20\ h \cdot 65\frac{€}{h} - 9\ h \cdot 65\frac{€}{h} = 715\ € \quad (3)$$

Suunnitteluun käytettävien tuntien vähentämisellä yritys ei kuitenkaan luo kassavirtaa, sillä palkkakustannukset ovat kiinteät eivätkä riipu suunnitteluun käytetyistä tunneista. Investoinnin takaisinmaksua ei siis voi laskea suoraa yhtälöä 3 hyödyntämällä, jos oletetaan että kaikki työntekijät ovat kuukausipalkkaisia. Työntekijöiden ollessa kuukausipalkkaisia, takaisinmaksuun tarvittava raha pitäisi muodostaa karsimalla suunnittelun resursseja.

Robotmaster kasvattaa suunnittelun tuottavuutta, jonka parantuessa myös kapasiteetti suunnittelulle kasvaa. Jos oletetaan, että Robotmasterille sopivia tilauksia on paljon, niin säästetyt resurssit pystytään käyttämään hyödyksi. Kapasiteetin kasvattaminen ei kuitenkaan tuo säästöä, koska suunnittelun kustannukset ovat kiinteät. Myös asiakkaalta veloitettava suunnitteluaika pienenisi. Tuottavuuden kasvamisen näkökulmasta takaisinmaksu pitäisi saada laajentuneen liiketoiminnan tuomista voitoista.

Suunnittelunopeuden kasvattaminen voisi joissain tilanteissa nopeuttaa järjestelmien toimitusta. Huomioon on kuitenkin otettava, että projektien läpiviennin nopeuteen vaikuttaa useat asiat, kuten tarvittavien osien saatavuus ja toimitusajat. Liikekäskyjen ohjelmointiin käytetty aika on koko projektiin kuluvaan aikaan verrattuna melko marginaalinen osuus. Liikekäskyjen ohjelmoinnin nopeuttamisella tilausta olisi mahdollista nopeuttaa vain Robotmasterin säästämän ajan verran eli 11 tuntia.

Asiakkaan näkökulmasta tilanne on edullisempi, sillä suunnittelun ollessa halvempaa, myös robottisovellusten ostohinta laskee. Vielä suuremman säästön

asiakas voi luoda investoimalla Robotmasteriin itse. Mikäli asiakas luo itse robottien liikeohjelmat niin säätöä kertyy kappaletta kohden yhtälön neljä mukaan

$$80\ \frac{€}{h} \cdot 20\ h - 65\ \frac{€}{h} \cdot 9\ h = 1015\ €, \tag{4}$$

jossa on oletettu, että asiakkaan omakustannehinta suunnittelulle on myös 65 €/h.

Yhtälön 5 perusteella asiakas pystyisi maksamaan investoinnin takaisin jo

$$\frac{20000\ €}{1015\ €} = 19{,}7 \approx 20 \tag{5}$$

ohjelmoidun kappaleen jälkeen. Yhtälössä 5 ei ole huomioitu, että online-ohjelmoinnin aikana tarvitaan työvoimaa suorittamaan robotin työtehtävät, ellei tuotantoa pysäytetä. Työvoimakustannuksista kertyvät säästöt nopeuttavat takaisinmaksua, joten todellinen takaisinmaksuaika voi olla vielä nopeampi.

Asiakkaan näkökulmasta ohjelmistoon investointi voi siis olla hyvinkin kannattavaa. Investoinnin kannattavuudessa oleellista tietysti on, että kokemusta robotiikasta ei ole ja erilaisia kappaleita on riittävästi. Tämän lisäksi Robotmaster tukee useiden eri robottivalmistajien tuotteita. Tämä mahdollistaa kaikkien käytössä olevien robottien ohjelmoinnin samalla ohjelmistolla. Valmistavasta metalliteollisuudessa on myös ohjelmiston edellyttämää CAM-ohjelmointiosaamista, joten koulutuskustannukset olisivat minimaaliset.

# 8 POHDINTA

Tavoitteena oli perehtyä Robotmasterin ominaisuuksiin ja selvittää voiko JTA Connection hyötyä ohjelmistosta. Ohjelmistoa tutkittiin etenkin kappaleiden viimeistelyn kuten jäysteenpoiston helpottamiseksi. Tämän lisäksi tavoitteisiin lukeutui myös selvitys siitä, kuinka paljon JTA Connection voi hyötyä ohjelmistosta ja millä tavalla.

Tuloksena saatiin käsitys molempien käytössä olleiden Robotmasterin versioiden toiminnasta, sekä riittävä käsitys Mastercamin toiminnallisuuksista. Mastercamilla luodut työstöradat eivät olleet täydellisiä, mutta riittäviä johtopäätösten tekemiseksi. Tulosten pohjalta kummassakaan asiakastapauksessa ei päädytty valitsemaan Robotmasteria. Ensimmäisen asiakastapauksen kohdalla Robotmasteriin investointi ei ole kannattavaa, sillä käyttöliittymän kehittäminen on kustannustehokkaampi ratkaisu. Toisen asiakastapauksen kohdalla kappale on geometrialtaan liian monimutkainen, jotta ohjelman voisi tehdä Robotmaster V7:llä. Robotmaster V6.6 puolestaan edellyttää CAM-ohjelmointiosaamista, jota JTA Connectionilla ei ole.

Yhteenvetona tuloksista voi todeta, että JTA Connectionin kaltaisen järjestelmäintegraattorin ei kannata investoida tämän kaltaisiin offline-ohjelmointityökaluihin. Vahva osaaminen robottiohjelmoinnista ja CAM-ohjelmointiosaamisen puute ei ole ideaalinen yhdistelmä Robotmasterin kanssa. Tämän lisäksi JTA Connection toimii ensisijaisesti KUKA ja Fanuc integraattorina, joista molemmilla toimittajilla on omat offline-ohjelmointityökalut.

Isojen tuotantolaitosten näkökulmasta ohjelmisto voi kuitenkin osoittautua hyödylliseksi, kuten kappaleessa 7 on todettu. Useiden eri robottimerkkien käyttö samalla ohjelmistolla ja robottiohjelmien luominen ilman yrityksen ulkopuolista toimijaa ovat merkittäviä etuja. Robotmasterin edut tarjoavat joustavuutta yrityksen tuotannonsuunnitteluun ja mahdollistavat useiden eri robottimerkkien käytön ilman useita eri ohjelmistolisenssejä.

Työssä onnistuttiin luomaan ajettavia työstöratoja molemmilla Robotmasterin versioilla, joista saatiin kokonaisvaltainen käsitys ohjelmien soveltuvuuksista.

Myös KUKAn käytössä Robotmasterin kanssa onnistuttiin hyvin. Mastercamin toiminnan syvällinen ymmärtäminen jäi työssä vähälle, mistä aiheutui osittain myös ongelmia. Useat työstöt ovat suoritettu 5-akselisina työstöinä, jonka täydellisesti hallitseminen CAM-ohjelmassa edellyttäisi reilusti yli kahden kuukauden perehtymistä.

Tässä tutkimuksessa esitettyjä Robotmasterin ominaisuuksia on käytetty onnistuneesti koeajoissa. Tämän takia tuloksia voidaan hyödyntää esimerkiksi Robotmasteriin tutustumisessa tai käyttöönoton tukena, etenkin jos käytössä on KUKAn robotti. Tutkimuksen pohjalta on myös mahdollista arvioida ohjelmiston soveltuvuutta yrityksen liiketoimintaan. Tuloksia voi myös hyödyntää vastaavissa tutkimuksissa.

Tutkimuksen jatkokehitystä varten on suositeltavaa hankkia vahva CAM-ohjelmointiosaaminen, jota ei tällä hetkellä ole JTA Connectionilla. Testiajoja olisi myös hyvä suorittaa enemmän kuin kaksi lisäymmärryksen saamiseksi. Testiajot voisivat koostua esimerkiksi kolmesta kappaleesta, jotka ovat ohjelman haastavuutta ajatellen erilaisia. Kaikki testiajot voisi myös suorittaa loppuun asti oikeilla työkaluilla, jotta työnlaadun pääsisi todentamaan.

Tutkimuksessa esitettyjä tuloksia tukee Robotmasterin omat kotisivut. Sivuilla muun muassa mainitaan, että Robotmasterin kehittäjillä on vahva osaaminen CAD/CAM-ohjelmistoista. Tämän lisäksi kappaleessa 4.6 ja Robotmasterin kotisivuilla on yhden pitävästi todettu, että Robotmasterilla luotujen ohjelmien käyttöönotto edellyttää hyvin vähän ohjelmointiosaamista. Robotmasteria myös kuvaillaan työkaluksi, jonka avulla robottiohjelmia pystyy luomaan, vaikka ei olisi robotiikan asiantuntija. (Robotmaster n.d.)

Robotmasterin kotisivut tukevat siis väitteitä siitä, että ohjelmisto ei ole ideaalinen järjestetelmäintegraattorille, jolla on osaamista robotiikasta, mutta ei CAM-ohjelmoinnista. Järjestelmäintegraattoreiden asiakkaiksi lukeutuvat tuotantolaitokset puolestaan voivat Robotmasterin kotisivujen ja tutkimustulosten mukaan hyötyä ohjelmistosta, sillä ohjelmiston käyttö ei edellytä robotiikan osaamista juuri lainkaan.

**LÄHTEET**

Iisac Maw. 2019. The What Why and How of Industrial Robot Simulation Software for Offline Programming. Luettu 2.3.2020 https://www.engineering.com/AdvancedManufacturing/ArticleID/18288/The-What-Why-and-How-of-Industrial-Robot-Simulation-Software-for-Offline-Programming-OLP.aspx

ISG Stuttgart. 2020. Mechanical Backlash. Luettu 15.3.2020. https://www.isg-stuttgart.de/kernel-html5/en-GB/308643979.html

KUKA GmbH. 2015. KUKA System Software 8.3. Operating and Programming Instructions for System Integrators. Augsburg: KUKA Roboter GmbH. http://www.wtech.com.tw/public/download/manual/kuka/krc4/KUKA%20KSS-8.3-Programming-Manual-for-SI.pdf

KUKA CNC 2.1. 2013. KUKA CNC 2.1. For KUKA System Software 8.3. Augsburg: KUKA Roboter GmbH. http://supportwop.com/IntegrationRobot/content/5-Applicatifs_metiers_KST/CNC/KST_CNC_21_en

KUKA. n.d. KUKA CNC. Luettu 15.3.2020 https://www.kuka.com/en-us/products/robotics-systems/software/application-software/kuka_cnc

L.Tirloni, I.Fassi, G.Legnani. 2012. Robotics state of the art and future trends. Robot in Industrial Applications: State of the art and current trends. New York: Nova Science Publisher Inc.

Mastercam. n.d. Mastercam solutions. Luettu 9.3.2020. https://www.mastercam.com/solutions/

Mühe, H., & Angerer, A., Hoffmann, A., Reif, W. 2010. On reverse-engineering the KUKA Robot Language. https://www.researchgate.net/publication/46587313_On_reverse-engineering_the_KUKA_Robot_Language

Petschnigg, C., Breitenhuber, G., Breiling, B., Dieber, B., & Brandstötter, M. 2018. Online simulation for flexible robotic manufacturing. In 2018 7th International Conference on Industrial Technology and Management (ICITM), IEEE, 88-92. https://www.researchgate.net/publication/323535498_Online_simulation_for_flexible_robotic_manufacturing

Phoenixrobotic. 2016. Budgetary Industrial Robotmaster Pricing. Luettu 16.3.2020. http://www.phoenixrobotic.com/wp-content/uploads/2016/09/Robotmaster-Prices-2016.pdf

Robotmaster. n.d. Success stories. Luettu 15.3.2020. https://www.robotmaster.com/en/success-stories

Robotmaster V7. 2018. Robotmaster V7 offline robot programming launch. Luettu 9.3.2020. https://www.robotmaster.com/en/newsroom/robotmaster-v7-offline-robot-programming-launch)

Robotmaster. n.d. Why robotmaster? Luettu 9.3.2020. https://www.robotmaster.com/en/why-robotmaster

# LIITTEET

## Liite 1. Ohjelmiston kustannukset (Phoenixrobotic 2016)

Robotmaster
CAD/CAM FOR ROBOTS

**Budgetary Industrial Robotmaster Pricing – 2016**

A Robotmaster v6.5 station includes:

- Suitable 64 bit Mastercam Mill or Mill 3D system
- Robotmaster parameter c-hook
- Robot simulation module
- Robot code generator (post processor)
- Robot definition (setup of kinematic model and simulation setup of one robot).
- Tool definition (setup of one tool)

Robotmaster

- For standard 6 axis robot
- Activated for one brand of robot
- One Robot and Tool definition included
- 1 year of maintenance included

**Robotmaster Pro System for 6 axis Drilling/Trimming/Contouring £16,995**

- Mastercam *Mill* functionality (including nesting).
- 3D Solid/Surface/Wireframe CAD
- 2, 2.5D toolpaths for contouring, pocketing and drilling
- 5 axis Toolkit for auto multi surface edge toolpath calculation
- 3D toolpaths with limited multi-surface roughing and finishing
- 5 axis toolpaths for contouring and drilling (not 5 axis surface machining)
- 1 x day onsite installation*, 5 x days training at 4D offices. **.
- 12 months support and software updates.

**Robotmaster Pro System for 6 axis Drilling/Trimming & Multi-surface Machining £21,995**

- Mastercam *Mill 3D* functionality (including nesting)
- 3D Solid/Surface/Wireframe CAD
- 2, 2.5D toolpaths for contouring, pocketing and drilling
- 5 axis Toolkit for auto multi surface edge toolpath calculation
- 3D toolpaths with advanced multi-surface roughing and finishing
- MultiAxis 5 axis toolpaths for advanced multiaxis, multi-surface machining
- 1 x day onsite installation*, 7 x days training at 4D offices. **
- 12 months support and software updates.

**Options:**

- Add Robotmaster external axes (rail or rotary) support - £995 per axis (Note: simultaneous rail/rotary programming only available on RM Pro System).
- Additional Robot brand activation - £995 per brand
- Additional Robot definition setup - £500 per setup
- Additional Tool definition - £250 per setup

Robotmaster will need a suitable 64 bit with minimum Windows 10 PC for installation. PC specification on application.

**Typical Payment Terms:**

a) 100% on Order

*** Mastercam system will include - training and installation

All prices exclude VAT and are subject to our normal terms and conditions and payment terms. E & OE. This budgetary price list is *strictly confidential* and not for distribution to any other 3rd parties without prior written consent from a Director at 4D Engineering Ltd.
A detailed quotation is available on request. * Note: this is for UK mainland installation. Off mainland will be subject to any additional travel expenses and travel time costs by prior agreement. ** Training is for up to 2 people on same course.