Opinnäytetyö (AMK)

Kone- ja tuotantotekniikka

Koneautomaatiotekniikka

2017

Mervi Salmi

SIMULOINTI JA OFFLINE-OHJELMOINTI ROBOTEXPERT-OHJELMISTOLLA

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Kone- ja tuotantotekniikka | Koneautomaatio

Tammikuu 2017 | Sivumäärä 29 + 64

Ohjaaja: Sakari Koivunen

Mervi Salmi

SIMULOINTI JA OFFLINE-OHJELMOINTI ROBOTEXPERT-OHJELMISTOLLA

Opinnäytetyön tavoitteena oli tutustua Siemensin RobotExpert-ohjelmiston käyttöön. RobotExpert on robottisimulointiin ja offline-ohjelmointiin tarkoitettu ohjelmisto, jota voidaan käyttää useiden eri teollisuusrobottivalmistajien robottien kanssa. Lisäksi tavoitteena oli selvittää, soveltuuko ohjelmisto käytettäväksi osana Turun ammattikorkeakoulun koneautomaatio-opetusta.

Ohjelmiston käyttöä opeteltiin Siemensin tarjoamien tutoriaalivideoiden ja ohjelmiston oman ohjeen avulla. Ohjelmistolla luotiin simulointimalli Turun ammattikorkeakoulun käytössä olevasta särmäyssolusta. Mallin avulla harjoiteltiin offline-ohjelmointia ja testattiin ohjelman lataamista oikealle robotille sekä robotilla luodun ohjelman siirtämistä ohjelmistoon.

Lopputulokseksi saatiin käyttöohje ohjelmiston peruskäyttöä varten sekä ohjeistus ohjelman siirtämisestä oikealta robotilta ohjelmistoon ja päinvastoin. Lisäksi työ sisältää kuvauksen offline-ohjelmoinnin eri vaiheista sekä selvityksen ohjelmiston soveltuvuudesta Turun ammattikorkeakoulun koneautomaatio-opetukseen. Ohjelmiston katsottiin olevan hyödyllinen työkalu simuloinnin ja offline-ohjelmoinnin opetteluun.

ASIASANAT:

teollisuusrobotti, offline-ohjelmointi, etäohjelmointi, simulointi

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Mechanical and Production Engineering | Machine Automation

January 2017 | Total number of pages 29 + 64

Instructor: Sakari Koivunen

Mervi Salmi

SIMULATION AND OFFLINE PROGRAMMING WITH ROBOTEXPERT SOFTWARE

The purpose of this thesis was to study RobotExpert software from Siemens. RobotExpert is a robot simulation and offline programming software that supports robots from different industrial robot vendors. Furthermore, the aim was to study whether RobotExpert is a suitable software to be used as a part of machine automation education of Turku University of Applied Sciences.

Tutorial videos from Siemens and the help of the software were used to learn how to use the software. A simulation model of a robotic press brake cell used at Turku University of Applied Sciences was made with the software. Offline programming was practiced with the model as well as downloading and uploading robotic programs from and to the real robot.

As a result, an instruction manual for basic use of the software was made containing instructions on how to download and upload robotic programs. Also, the thesis includes a description of the phases of offline programming and the suitability of the software as a part of machine automation education of Turku University of Applied Sciences is discussed. RobotExpert was found to be a practical tool for learning simulation and offline programming.

KEYWORDS:

industrial robot, offline programming, simulation

SISÄLTÖ

LIITTEET

KUVAT

KÄYTETYT LYHENTEET

JBI	Motoman Robot Job File, Motoman-robottien käyttämä tiedostomuoto datan siirtämiseen robotin ja simulointiohjelmien välillä (OpenTheFile 2016).
OLP	Off-Line Programming, offline- eli etäohjelmointi (Wikipedia 2017).
PLC	Programmable Logic Controller, ohjelmoitava logiikka, käytetään automaatioprosessien ohjauksessa (Wikipedia 2015).
PLM	Product Lifecycle Management, tuotteen elinkaaren hallinta (Wikipedia 2014).
RCS	Robot Controller Simulation module, robottikohtainen ohjelmiston lisäosa, jolla robottimallin liikkeet saadaan vastaamaan tarkemmin oikean robotin liikkeitä (Siemens PLM Software 2017a).
TCP	Tool Center Point, työkalupiste, robotin työkalukoordinaatiston origo (ABB 2017).
XML	Extensible Markup Language, XML-kieli on rakenteellinen kuvauskieli, jota käytetään tiedonvälitykseen järjestelmien välillä sekä dokumenttien tallentamiseen (Wikipedia 2016).

1 JOHDANTO

Tämän opinnäytetyön tavoitteena on tutustua Siemensin robottisimulointiin ja offline-ohjelmointiin suunniteltuun RobotExpert-ohjelmistoon. Työ tehtiin Turun ammattikorkeakoululle ja sen tarkoituksena oli selvittää, soveltuuko ohjelmisto käytettäväksi osana koneautomaatiotekniikan opetusta. Selvityksen kohteeksi valittiin Siemensin RobotExpert-ohjelmisto, koska Turun ammattikorkeakoulu tekee yhteistyötä Siemens PLM Softwaren palveluja Suomessa toimittavan IDEAL PLM:n kanssa.

Tällä hetkellä opiskelijat tutustuvat simulointiin ja offline-ohjelmointiin pääasiassa ABB:n RobotStudio-ohjelmiston avulla. Turun ammattikorkeakoululla ei ole käytössä ABB:n robotteja, joten ohjelmistolla luotuja ohjelmia ei ole päästy testaamaan oikeilla roboteilla. Siemensin RobotExpert-ohjelmisto tukee monien eri valmistajien kuten Fanucin ja Motomanin robotteja, joita ammattikorkeakoululla on käytössä. RobotExpert-ohjelmiston avulla opiskelijat voisivat käytännönläheisemmin tutustua simulointiin ja offline-ohjelmointiin, koska ohjelmistolla luoduilla virtuaalisilla robottisoluilla olisi vastine oikeassa maailmassa. Ohjelmia voitaisiin ladata oikealle robotille ja toisaalta siirtää robotilla luotuja ohjelmia tarkasteltavaksi ja muokattavaksi ohjelmistoon.

Tarkkaa tilastotietoa etä- eli offline-ohjelmoinnin käytöstä suomalaisissa yrityksissä ei tätä työtä tehtäessä ollut saatavilla, mutta muun muassa Turun ammattikorkeakoulussa vuonna 2014 tehty opinnäytetyö *Robottien etäohjelmoinnin kypsyystaso* viittaa siihen, että offline-ohjelmointia ei käytetä laajalti Suomessa. Syiksi mainitaan osaavan työvoiman puute sekä ohjelmistojen ja käyttökoulutuksen kalleus (Rainio 2014, 28–29). Yli kymmenen vuotta sitten Tekniikka&Talous julkaisi verkkosivuillaan artikkelin *Satakunta kaivaa robotit nurkista tehokäyttöön*, jossa niin ikään mainitaan offline-ohjelmoinnin olevan useissa yrityksissä merkittävä kompastuskivi, koska ohjelmistoja ei osata käyttää (Kuusinen 2003). Lisäksi Liikenne- ja viestintäministeriön teettämässä selvityksessä *Robotiikan taustaselvityksiä* mainitaan ylipäätään automaatio- ja robotiikkaosaamisen puutteen jarruttavan automaatioalan kehitystä Suomessa (Liikenne- ja viestintäministeriö 2016, 59).

Robotiikkaosaamisen lisääminen on siis perusteltua ja automaatio-opetuksen kehittämiselle on tarvetta. Kun valmistuva automaatioinsinööri hallitsee esimerkiksi offline-ohjelmoinnin, ei hänet palkkaavan yrityksen tarvitse panostaa kalliisiin kursseihin. Tämä ei tietysti muuta sitä, että käytettävät ohjelmistot saattavat olla hintavia, mutta ohjelmistojen

yleistyttyä hinnatkin todennäköisesti laskevat. Lisäksi tehokkaasti käytettynä ohjelmistot lisäävät merkittävästi robottien käyttösuhdetta ja maksavat itsensä näin takaisin (Delfoi 2017).

Tässä työssä käydään lyhyesti läpi robottisimuloinnin ja -ohjelmoinnin teoriaa, esitellään RobotExpert-ohjelmisto ja kerrotaan offline-ohjelmoinnin vaiheet yleisesti sekä miten tämän työn tekeminen eteni. Lopuksi pohditaan ohjelmiston hyviä ja huonoja puolia automaatio-opetuksen kannalta. Ohjelmiston käyttöä varten on kirjoitettu ohje, joka löytyy liitteestä 1.

2 ROBOTTISIMULOINTI

Robottisimuloinnissa, siinä missä muunlaisessakin simuloinnissa, todellisesta systeemistä luodaan malli, jolla suoritetaan kokeita, jotta voitaisiin ymmärtää systeemin käyttäytymistä (Shannon 1998, 7). Robottisimuloinnissa systeemillä tarkoitetaan robottia, sen ympäristöä ja oheislaitteita. Näistä tehdään 3D-mallit, jotka viedään robottisimulointiohjelmistoon, jonka avulla robottijärjestelmän toimintaa voidaan testata.

Simulointia käytetään teollisuudessa, koska se helpottaa robottijärjestelmien suunnittelua. Simuloinnilla voidaan muun muassa testata erilaisia layout-vaihtoehtoja, arvioida työaikaa, tehdä törmäystarkasteluja sekä tarkistaa robotin ulottuvuusalueet. Lisäksi voidaan suunnitella ja testata erilaisia työkaluja ja oheislaitteita. Simuloinnin avulla mahdolliset ongelmat voidaan havaita ennen kalliita investointeja. (Kuivanen 1999, 96.)

Simuloinnin käyttöä automaatio-opetuksessa voidaan perustella sillä, että valmistuva koneautomaatioinsinööri pystyy paremmin vastaamaan työelämän tarpeisiin. Toinen simuloinnin tuoma hyöty opetuksessa on sen edullisuus todelliseen robottijärjestelmään verrattuna. Simulointimalleja voidaan luoda eri tarkoituksiin ja harjoitella ohjelmointia näiden avulla ilman, että koulun tarvitsisi rakentaa kymmentä erilaista robottisolua opetuskäyttöön. Kokemuksesta voidaan kuitenkin todeta, että simulointimallin kanssa työskentely ei täysin vastaa oikean robotin kanssa työskentelyä. Jos haluaa oppia käyttämään oikeaa robottia, vaatii se muutakin kuin simulointiympäristössä työskentelyä. Esimerkiksi kiertyvänivelisen robotin käsivarren liikuttelu tietokoneen näytöllä hiiren avulla poikkeaa melko paljon oikean robotin liikuttamisesta käsiohjaimella.

3 ROBOTIN OHJELMOINTI

Robotin tärkeimpiin ominaisuuksiin kuuluu sen joustavuus ja monipuolisuus, joka saadaan aikaiseksi erilaisilla tarttujilla ja työkaluilla sekä ammattitaitoisella ohjelmoinnilla (Alander & Niemi 1987, 19). Nykyisillä globaaleilla markkinoilla joustava automatisoitu tuotanto on houkutteleva vaihtoehto pienille ja keskisuurille yrityksille. Tuotevalikoima monipuolistuu uusien innovaatioiden myötä ja tuotteiden elinkaaret lyhenevät. Teollisuusrobotit tarjoavat parhaan ratkaisun sekä tuottavuudelle että joustavuudelle. Valitettavasti ohjelmoinnin monimutkaisuus, suuritöisyys ja kustannukset ovat suurimpia esteitä teollisuusrobottien käytölle PK-yrityksissä. (Pan ym. 2011, 87–88.)

Robottien ohjelmointimenetelmät voidaan jakaa online- sekä offline-ohjelmointiin. Ne poikkeavat toisistaan siten, että online-ohjelmoinnissa robotti on osa ohjelmointijärjestelmää, eikä se voi toimia tuotannossa ohjelmoinnin aikana. Offline-ohjelmointi tapahtuu erillisellä ohjelmointijärjestelmällä tuotannon ulkopuolella, jolloin robotti voi olla tuottavassa työssä ohjelmoinnin ajan. (Alander & Niemi 1987, 19.) Molempia menetelmiä on viime vuosina pyritty yksinkertaistamaan, jotta kynnys robottien käyttöönotolle erityisesti PK-yrityksissä ei olisi niin korkea (Pan ym. 2011, 88). Tämän työn pääpaino on offline-ohjelmoinnissa, erityisesti mallipohjaisessa etäohjelmoinnissa, mutta vertailun vuoksi käydään lyhyesti läpi myös online-ohjelmointimenetelmät.

3.1 Online-ohjelmointi

Online-ohjelmointi voidaan jakaa johdattamalla ohjelmointiin ja opettamalla ohjelmointiin. Molemmissa menetelmissä ohjelma tehdään suoraan robotilla, jolloin robotti ei voi olla tuottavassa työssä.

Johdattamalla ohjelmoinnissa robottia liikutellaan käsin haluttujen pisteiden kautta tai haluttua liikerataa pitkin. Nivelten asennot ja muut komennot, esimerkiksi työkalun toiminta, tallennetaan muistiin. Johdattamismenetelmä on helppo ohjelmointitapa ja se oli aiemmin hyvin suosittu teollisuudessa. (Alander & Niemi 1987, 20.) Edelleenkin sitä käytetään esimerkiksi maalausroboteissa ja muissa yksinkertaisissa sovelluksissa, joissa liikeradalta ei vaadita suurta tarkkuutta (Keinänen ym. 2007, 262).

Epätarkkojen liikeratojen lisäksi johdattamalla ohjelmointia rajoittaa toistorakenteen puuttuminen, jolloin esimerkiksi ohjelmoitaessa kappaleiden noutoa paletilta, täytyy robotti johdattaa jokaisen kappaleen luo erikseen. Ohjelmointivirheiden korjaaminen on myös vaikeaa ja virheen sattuessa joudutaan usein ohjelmoimaan koko tehtävä uudelleen. (Alander & Niemi 1987, 20.) Näiden rajoitusten vuoksi johdattamalla ohjelmointi voi viedä tarpeettoman paljon aikaa.

Johdattamalla ohjelmoinnin rajoituksia on pyritty poistamaan opetusmenetelmällä (Alander & Niemi 1987, 20). Opettamalla ohjelmoitaessa robottia liikutellaan haluttuihin pisteisiin erillisellä käsiohjaimella ja pisteet tallennetaan ohjaimen muistiin. Tallennetuille pisteille määritetään, millä tavalla ja millä nopeudella liike pisteeseen tapahtuu. Liiketapa voi olla esimerkiksi lineaarinen tai liike voi tapahtua ympyräkaarta pitkin. Opettamalla ohjelmointi sopii yksinkertaisiin sovelluksiin, kuten paletointi- ja pakkaussovelluksiin, joissa robotti liikkuu pisteestä pisteeseen. (Keinänen ym. 2007, 262.)

Vaikka opettamalla ohjelmointi on yksinkertainen ja laajasti käytetty menetelmä, silläkin on rajoituksensa. Robotin liikuttaminen käsiohjaimella ei välttämättä ole intuitiivista erilaisten koordinaatistojen takia. Ohjelmointi voi viedä paljon aikaa, jos työstettävä kappale on monimutkainen ja opetettavia pisteitä on paljon. (Pan ym. 2011, 88.)

3.2 Offline-ohjelmointi

Offline- eli etäohjelmointi voidaan jakaa tekstipohjaiseen ja mallipohjaiseen ohjelmointitapaan. Molemmissa tavoissa ohjelma tehdään erillisellä ohjelmistolla tietokoneella, jolloin robotti voi olla tuottavassa työssä ohjelmoinnin ajan. Robotti pysäytetään vain ohjelman siirtämisen ja toiminnan tarkastamisen ajaksi.

Tekstipohjaisessa etäohjelmoinnissa ohjelma luodaan ulkopuolisella tietokoneella kirjoittamalla tarvittavat käskyt robottikohtaisella ohjelmointikielellä (Malm 2008, 97). Huonoa tässä ohjelmointitavassa on se, että usein liikepisteet joudutaan tarkistamaan opettamalla ne robotille online-tilassa, jolloin robotti ei voi olla tuottavassa työssä. CAD- ja 3D-mallinnusohjelmistojen kehityksen myötä syntyi tekstipohjaista etäohjelmointia tehokkaampi mallipohjainen etäohjelmointimenetelmä. (Nof 1999, 756.)

Mallipohjaisessa etäohjelmoinnissa erillisellä etäohjelmointiohjelmistolla robottisolusta luodaan simulointimalli ja robotin liikeratojen luomisessa voidaan hyödyntää työstettävän

kappaleen 3D-mallin geometriaa. Esimerkiksi tässä työssä käytetty RobotExpert on mallipohjainen etäohjelmointiohjelmisto.

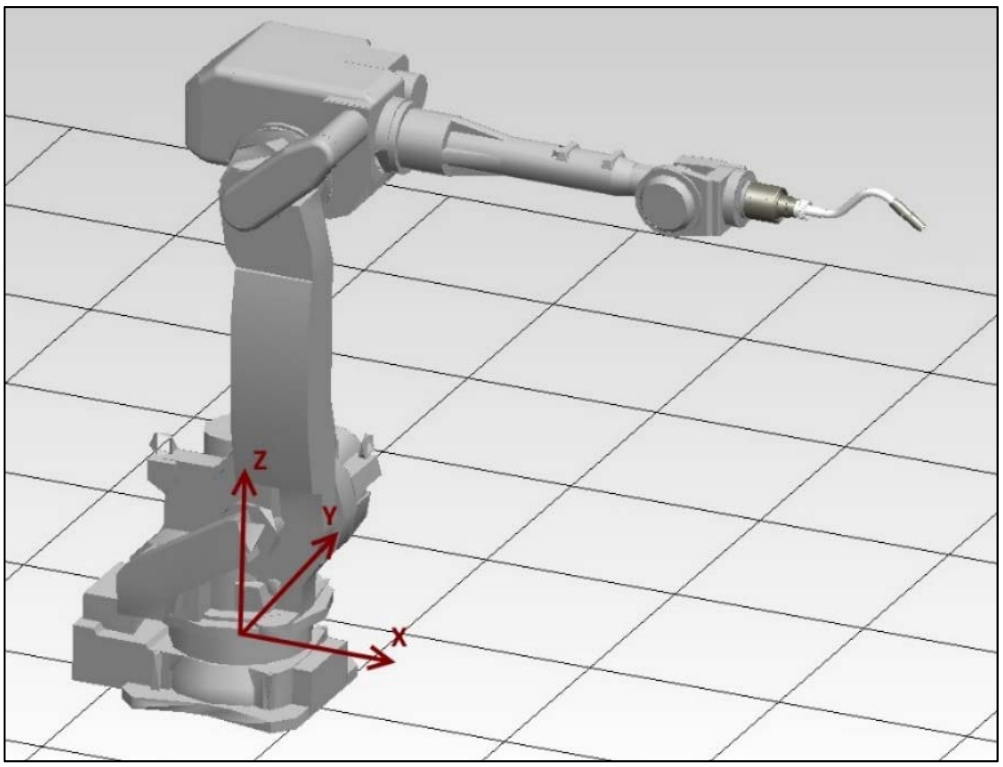
Mallipohjaista ohjelmointijärjestelmää voidaan usein käyttää muuhunkin kuin robotin ohjelmoimiseen, ja siksi sen voidaan katsoa sijoittuvan tuotekehityksen ja valmistuksen väliin (Kuivanen 1999, 81). RobotExpert-ohjelmistoa voidaan offline-ohjelmoinnin lisäksi käyttää esimerkiksi robottisolujen layoutin suunnitteluun ja törmäystarkasteluun. Mallipohjaisilla ohjelmointiohjelmistoilla voidaan myös testata tuotteen valmistettavuutta simuloimalla, kun halutaan tietää, sopivatko olemassa olevat työkalut, kiinnittimet ja muut oheislaitteet uuden tuotteen valmistukseen (Kuivanen 1999, 82).

Mallipohjainen ohjelmointiohjelmisto pystyy yhdistämään robotin työkalukoordinaatiston tuotteen geometriamuotoihin, kuten sivuihin, tasoihin ja kulmapisteisiin. Muototietoa hyödyntämällä voidaan ohjelmointia helpottaa, sillä ohjelmisto generoi paikoituspisteitä näiden muotojen perusteella, eikä kaikkia liikepisteitä tarvitse opettaa yksitellen. (Kuivanen 1999, 84). Mallipohjainen ohjelmointi on siis samankaltaista kuin opettamalla ohjelmointi, eli virtuaalista robottia liikutetaan haluttuihin pisteisiin ja pisteet tallennetaan. Mallipohjainen ohjelmointi on kuitenkin usein nopeampaa muototietojen hyödyntämisen vuoksi. Esimerkiksi liitteessä 1 on esitetty RobotExpertillä tehty yksinkertainen hitsaussovellus, jossa hitsauspistoolin liikerata on generoitu hitsattavan kappaleen pohjan ja sivun geometrian perusteella. Lisäksi mallipohjaisessa ohjelmoinnissa pisteet voidaan tallentaa suhteessa työstettävään kappaleeseen, jolloin robottisolun layoutin muuttuessa ohjelmaa ei tarvitse kirjoittaa uudelleen, koska koordinaatistomuutokset huomioidaan automaattisesti (Kuivanen 1999, 84).

Mallipohjainen etäohjelmointi sopii erityisesti pienien sarjojen tai yksittäiskappaleiden tuotantoon, sillä uusien ohjelmien tekeminen ei vie robotin työaikaa. Lisäksi se sopii sellaisiin valmistusprosesseihin, jotka edellyttävät suuria määriä paikoituspisteitä. Pisteiden opetus on nopeampaa, etenkin jos voidaan hyödyntää työstettävän kappaleen muototietoa. Mallipohjainen etäohjelmointi sopii myös sellaisiin erityistapauksiin, joissa robottia ei ole turvallista ohjelmoida opettamalla, esimerkiksi ampumatarviketeollisuus ja ydinvoimalaitokset. (Kuivanen 1999, 82.) Automaatio-opetuksessa mallipohjaisen etäohjelmoinnin käytöstä on hyötyä esimerkiksi silloin, kun opetukseen tarkoitetut robotit ovat varattuja. Tällöin opiskelijat voivat harjoitella ohjelmointia virtuaalisella robotilla.

3.3 Koordinaatistot

Robottien liikuttamiseen ja ohjelmointiin käytetään erilaisia koordinaatistoja. Peruskoordinaatisto on kiinteä koordinaatisto, jonka origo sijaitsee robotin jalustassa, kuten kuvassa 1.

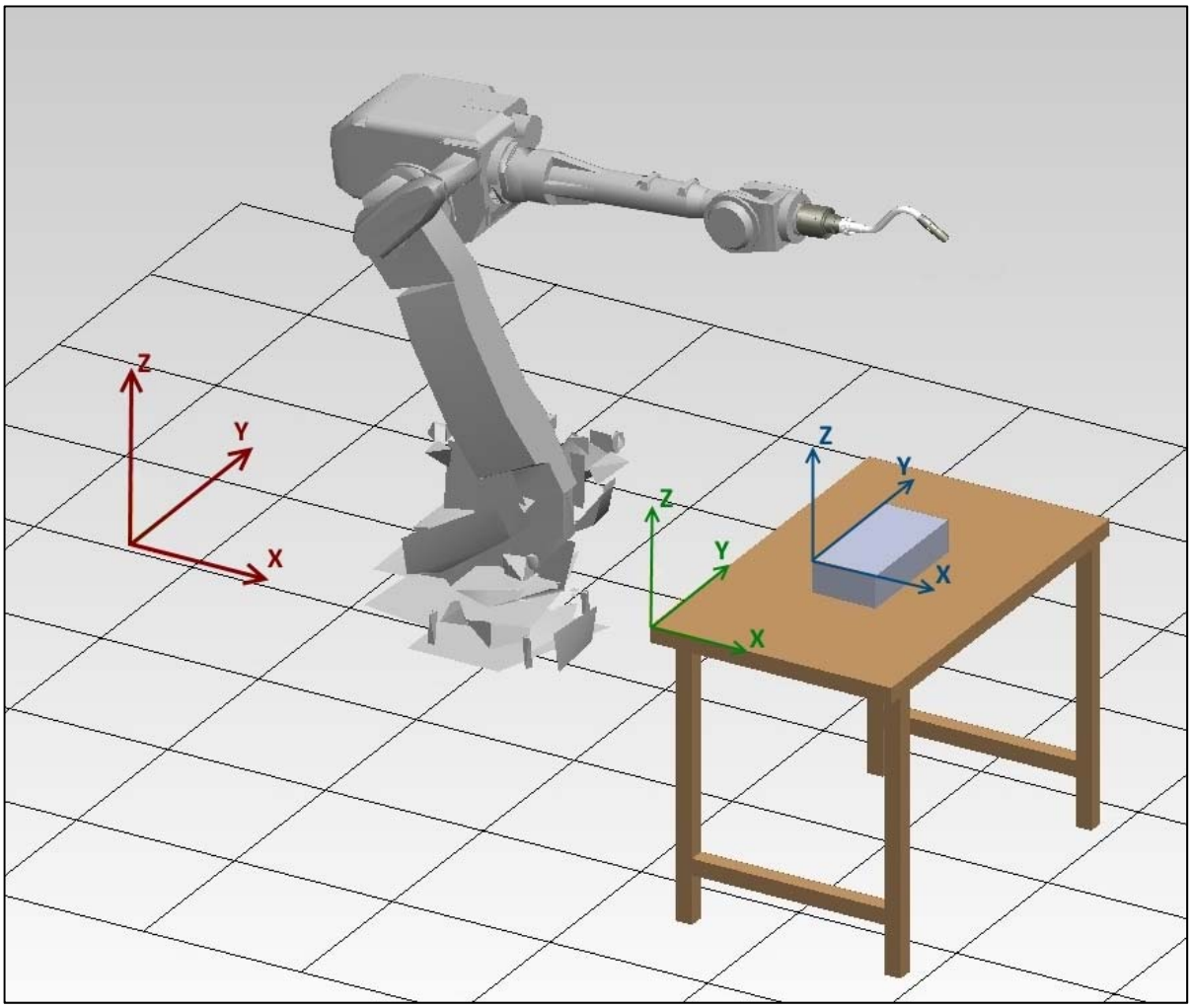


Kuva 1. Robotin peruskoordinaatisto.

Tavallisesti peruskoordinaatiston Z-akseli on robotin ensimmäisen vapausasteen akselin suuntainen, X-akseli osoittaa ensimmäisen nivelen työalueen keskikohtaan ja XY-taso on lattiatason suuntainen (Kuivanen 1999, 21). Peruskoordinaatisto sopii robotin liikuttamiseen asennosta toiseen, mutta robotin ohjelmointia varten on usein parempi käyttää muita koordinaatistoja, kuten työkappalekoordinaatistoa (ABB 2017).

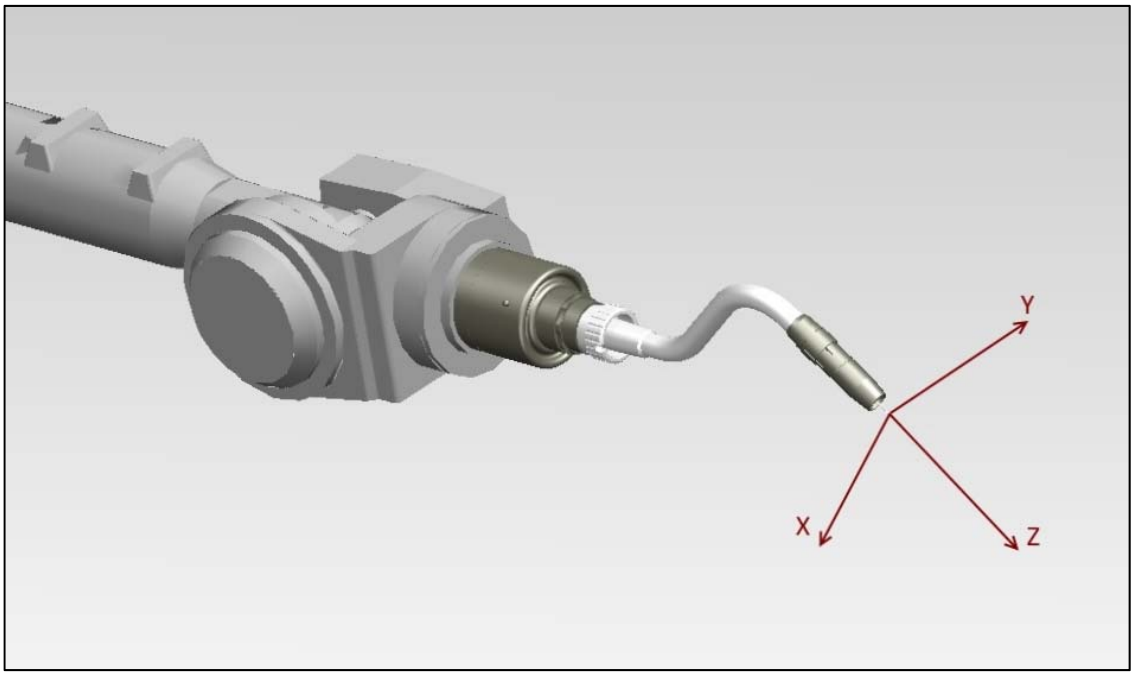
Maailmakoordinaatisto on robotin ulkopuolinen kiinteä koordinaatisto, joka määrittelee robottisolun. Kaikki muut koordinaatistot liittyvät maailmakoordinaatistoon suoraan tai epäsuorasti. Peruskoordinaatisto on usein maailmakoordinaatiston kanssa samansuuntainen ja sopii niin ikään robotin liikuttamiseen asennosta toiseen. Maailmakoordinaatisto

on hyödyllinen silloin, kun käsitellään useampia robotteja, tai kun robotin asema määräytyy ulkoisen akselin perusteella. (ABB 2017.) Maailmakoordinaatisto on esitetty kuvassa 2 punaisella.



Kuva 2. Maailmakoordinaatisto, käyttäjän koordinaatisto ja työkappalekoordinaatisto.

Työkalukoordinaatisto määrittelee robotin työkalun asennon ja se liikkuu robotin mukana. Työkalukoordinaatiston origoa kutsutaan työkalupisteeksi (TCP), ja se määritetään haluttuun kohtaan työkalua esimerkiksi kaarihitsauksessa vapaalangan päähän, kuten kuvassa 3. Työkalupiste on se piste, jonka robotti liikuttaa ohjelmoituun pisteeseen ohjelmaa suoritettaessa. Robotin liikuttaminen työkalukoordinaatiston mukaisesti on hyödyllistä silloin, kun työkalun asentoa ei haluta muuttaa liikkeen aikana, esimerkiksi sahanterää liikutettaessa siten, ettei terä pääse taipumaan. (ABB 2017.)



Kuva 3. Hitsauspistoolin työkalukoordinaatisto.

Työkappalekoordinaatisto liittyy nimensä mukaisesti työstettävään kappaleeseen ja sopii hyvin robotin ohjelmointiin. Työkappalekoordinaatisto määrittelee työstettävän kappaleen sijainnin suhteessa maailmakoordinaatistoon tai mihin tahansa muuhun koordinaatistoon. Työkappalekoordinaatisto on erityisen kätevä silloin, kun työstettävän kappaleen sijaintia muutetaan. Jos liikepisteet ja -radat on ohjelmoitu käyttäen työkappalekoordinaatistoa, ei tarvitse muuttaa kuin työkappalekoordinaatiston paikkaa ja ohjelmoitujen liikeratojen paikat päivittyvät automaattisesti. Työkappalekoordinaatisto mahdollistaa myös sellaisten kappaleiden työstämisen, jotka liikkuvat esimerkiksi liukuhihnalla, koska ohjelmoidut liikeradat liikkuvat kappaleen mukana. (ABB 2017.) Työkappalekoordinaatisto on esitetty kuvassa 2 sinisellä.

Käyttäjän koordinaatistot helpottavat ohjelmointia kuten työkappalekoordinaatistot. Käyttäjän koordinaatisto on kiinteä koordinaatisto, joka sopii sellaisten laitteiden ja kalusteiden määrittämiseen, joihin on kiinnitetty työstettäviä kappaleita tai muita koordinaatistoja, kuten työkappalekoordinaatistoja. (ABB 2017.) Esimerkki käyttäjän koordinaatistosta on esitetty kuvassa 2 vihreällä.

4 ROBOTEXPERT

RobotExpert on simulointi- ja offline-ohjelmointiohjelmisto, joka on osa Siemens PLM Softwaren digitaalisen valmistuksen ratkaisuihin kuuluvaa Tecnomatix®-tuoteperhettä (IDEAL PLM 2017). RobotExpert on riisuttu versio saman tuoteperheen Process Simulate -ohjelmistosta (Fabryka Robotów 2012).

Siemens PLM Software mainostaa RobotExpert-ohjelmiston olevan helposti käyttöönotettava suunnitteluun, simulointiin, optimointiin ja offline-ohjelmointiin sopiva ohjelmisto, joka tukee teollisuudelle ominaisia sovelluksia, kuten kappaleiden siirtoa, kaarihitsausta, kiillotusta, liimausta jne. RobotExpert tukee monien eri valmistajien robotteja, esimerkiksi ABB:n, Fanucin, Kukan ja Motomanin robotteja. (Siemens PLM Software 2017b.)

RobotExpert mahdollistaa robotin liikesekvenssien simuloinnin ja ohjelmien lataamisen oikealle robotille. Ohjelmisto toimii useimpien teollisuusrobottien kanssa ja muuttaa luodun ohjelman käytössä olevan ohjaimen vaatimusten mukaiseksi. Luotuihin liikeratoihin voidaan lisätä ohjainkohtaista tietoa, kuten liike- ja prosessimääreitä. Lisäksi RobotExpert-ohjelmistoon voidaan siirtää oikealla robotilla luotuja ohjelmia, jolloin niitä voidaan muokata ja optimoida sekä ladata uudelleen robotille. (Siemens PLM Software 2017b.)

RobotExpert sisältää joitain yleiskäyttöisiä OLP-käskyjä (offline-ohjelmointikäskyjä) ja lisäksi robottikohtaisten ohjainten mukana tulee robotille ominaisia käskyjä. Tämän lisäksi käyttäjä voi luoda itse OLP-käskyjä ja ylläpitää omia komento- ja makrokirjastoja, jolloin kerran luotuja käskyjä voidaan tarvittaessa käyttää uudelleen ilman ylimääräistä työtä (Siemens PLM Software 2017b). Omien OLP-käskyjen luomista varten täytyy tuntea XML:n (Extensible Markup Language) käyttöä. Käyttäjä pystyy määrittämään, miten omat OLP-käskyt simuloidaan RobotExpertillä. (Siemens PLM Software 2017c.)

RobotExpertiä voidaan käyttää robottisolujen suunnittelussa. Layoutin suunnittelua varten ohjelmistossa on Smart Place -työkalu, jolla robotin ja muiden komponenttien sijainti solussa voidaan optimoida ja tarkistaa, että robotti yltää tarvittaviin pisteisiin. Suunnittelua auttaa myös törmäystarkasteluun tarkoitettu työkalu, jolla voidaan simuloinnin aikana havaita mahdolliset törmäykset. Tällä työkalulla voidaan myös analysoida, onko tapahtunut kappaleen tunkeutuminen toiseen kappaleeseen, kappaleiden kosketus vai pelkästään läheltä piti -tilanne. (Siemen PLM Software 2017b.)

RobotExpertillä pystyy luomaan kinemaattisia 3D-malleja, kuten pyörityspöytiä tai muita työkaluja, joissa on liikkuvia osia. Kappaleita pystytään mallintamaan RobotExpertillä, mutta myös muilla mallinnusohjelmilla luotuja malleja voidaan tuoda ohjelmistoon. (Siemens PLM Software 2017b.) Liitteestä 1 löytyy lisätietoa mallien tuomisesta ohjelmaan ja muista ohjelmiston ominaisuuksista.

RobotExpert sisältää oletusohjaimen, jota voidaan käyttää kaikkien robottien kanssa. Joillain robottivalmistajilla on tarjota robottikohtaisia RCS-moduuleja (Robot Controller Simulation) osana robottikohtaista ohjainta. (Siemens PLM Software 2017a.) Oletusohjainta käytettäessä robotin liikepisteiden sijainnit ovat samat kuin oikeaa ohjainta käytettäessä, mutta robotti saattaa liikkua näiden pisteiden välillä eri tavalla kuin oikeasti (Konopa 2013, 33). Jos RobotExpertillä käytetään robottikohtaisen ohjaimen kanssa RCS-moduulia, simulointimallin liikkeet saadaan vastaamaan tarkasti oikean robotin liikkeitä (Siemens PLM Software 2017a).

Siemens käyttää robottisimulointiohjelmistoissaan aikaperusteista simulointia (time-based simulation) ja tapahtumaperusteista simulointia (event-based simulation). Molemmat simulointitavat ovat käytössä Process Simulate -ohjelmistossa. RobotExpert pohjautuu Process Simulate -ohjelmistoon, mutta tapahtumaperusteinen simulointi ei ole RobotExpertillä mahdollista (Fabryka Robotów 2012). Tapahtumaperusteinen simulointi voi edetä usealla eri tavalla, sillä erilaiset signaalit ohjaavat simulaatiota. Tapahtumaperusteisessa simuloinnissa voidaan käyttää joko oikeaa tai virtuaalista PLC:tä (ohjelmoitavaa logiikkaa) signaalien käsittelyyn. Aikaperusteista simulointia rajoittaa eri operaatioiden kestoaika ja simulointi voi johtaa ainoastaan yhteen määrättyyn lopputulokseen. Aikaperusteisen simuloinnin logiikka perustuu Gantt-kaavioon, jonka avulla määritetään, mikä operaatio alkaa, kun edellinen päättyy ja mitkä operaatiot ovat käynnissä samanaikaisesti. (Konopa 2013, 38–46.)

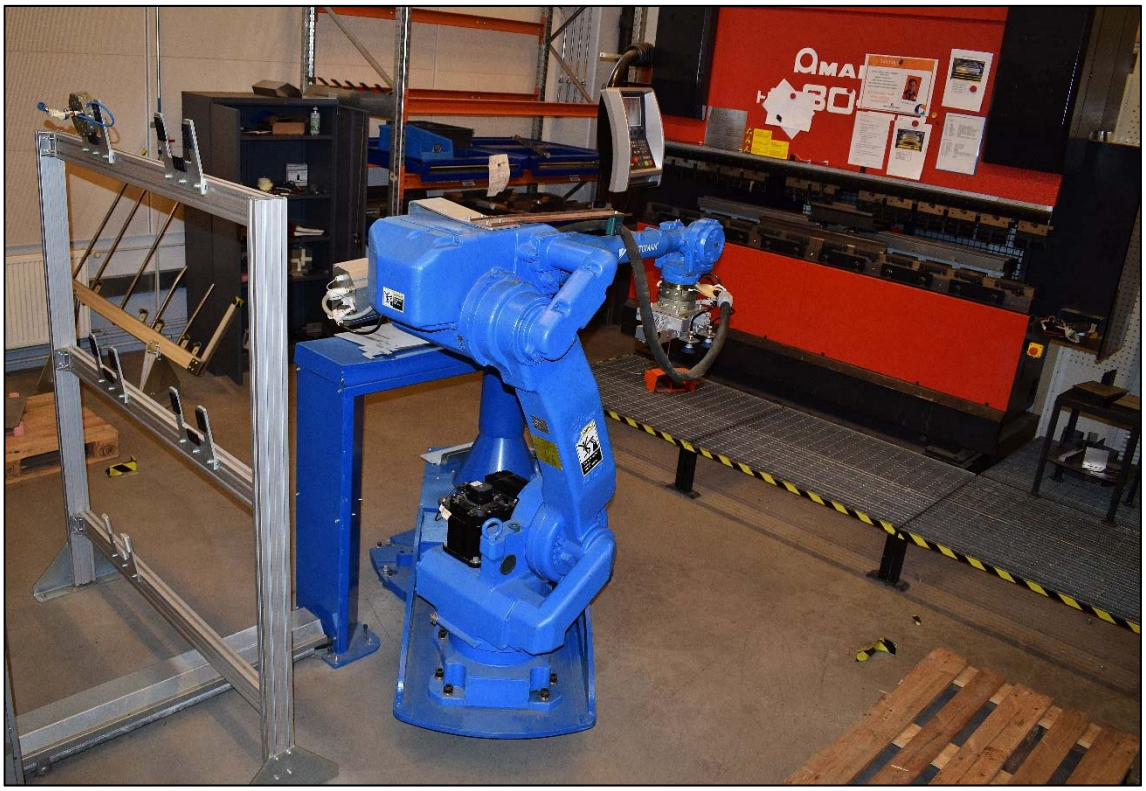
5 OFFLINE-OHJELMOINNIN VAIHEET JA TYÖN SUORITUS

Yksinkertaisimmillaan offline-ohjelmoinnin vaiheet voidaan jakaa robottisolun komponenttien mallinnukseen ja sijoittamiseen simulointisoluun, liikepisteiden generointiin ja OLP-käskyjen kirjoittamiseen, ohjelman toiminnan tarkistamiseen simuloimalla ja valmiin ohjelman lataamiseen robotille. Offline-ohjelmointiohjelmistolla luodut robottiohjelmat voidaan ottaa käyttöön tuotannossa, jos robottisolun simulointimalli vastaa tarkasti tuotantosolua. Tämä edellyttää usein simulointimallin kalibrointia. (Kuivanen 1999, 85–86.)

5.1 Simulointimallin luominen

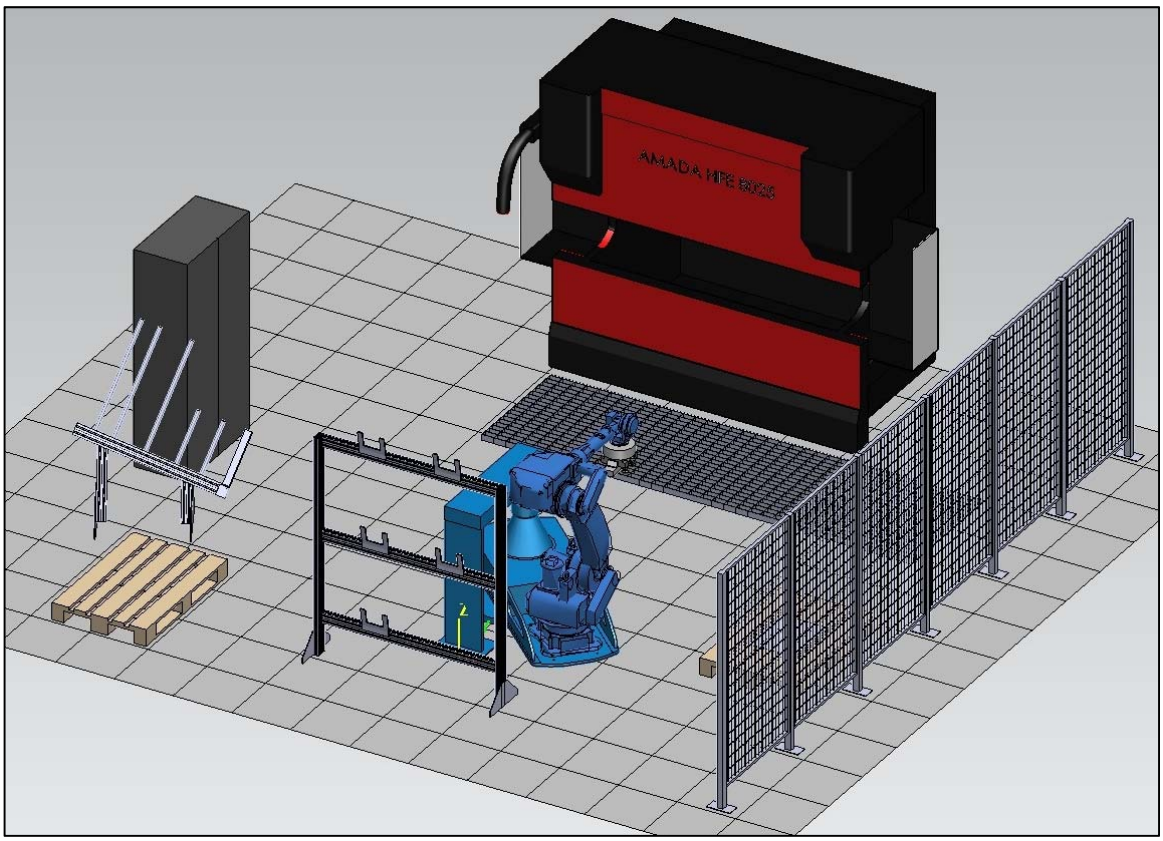
Ensimmäinen vaihe voi olla jo olemassa olevan solun mallintaminen tai uuden solun suunnittelu. Kummassakin tapauksessa solun koneet, laitteet ja muut komponentit mallinnetaan. Mallinnuksessa kannattaa ottaa huomioon solun toiminnan kannalta tärkeät tekijät. Mallista on turha tehdä liian tarkkaa, jos yksinkertaisempikin malli toimii.

Tässä työssä mallinnettiin Turun ammattikorkeakoulun käytössä oleva särmäyssolu. Solussa on Motoman UP50N -robotti, jossa on työkaluna imukuppitarttuja, ja joka on kiinnitetty Motoman VST-600 SX -pyöritysjalkaan. Solussa on myös työkalunvaihtoasema, Amada HFE 80-25 -särmäyspuristin, kaksi EUR-lavaa, paikoitusteline, kaappi ja hyllykkö. Lisäksi särmäyspuristimen edessä on ritilätaso. Solu on esitetty kuvassa 4.



Kuva 4. Särmäyssolu.

Robotin malli saatiin ohjelmiston toimittajalta, pyöritysjalan malli Motomanin Internet-sivustolta ja särmäyspuristimen sekä imukuppitarttujan mallit Turun ammattikorkeakoululta. Työkalunvaihtoasema, EUR-lava, paikoitusteline, kaappi, ritilätaso ja turva-aita mallinnettiin itse SolidWorks-ohjelmistolla. Näiden solun osien katsottiin olevan tärkeimmät tämän työn kannalta, sillä niihin robotti saattaa törmätä siirtäessään ohutlevyä lavalta toiselle. Kaikkia solussa olevia komponentteja, kuten hyllykköä ja osaa turva-aidoista, ei siis lisätty soluun. Solun eri komponenttien sijainnit mitattiin rullamitalla ja sijoitettiin RobotExpert-ohjelmistoon näiden mittojen perusteella. Luotu simulointimalli on esitetty kuvassa 5.



Kuva 5. RobotExpertillä luotu särmäyssolun simulointimalli.

5.2 Kalibrointi

Offline-ohjelmoidut robottiohjelmat voidaan ottaa käyttöön oikealla robotilla, jos robottisolun simulointimalli vastaa tarpeeksi tarkasti oikeaa solua. Tämä edellyttää usein simulointimallin kalibrointia, jossa oikea robottisolu mitataan ja mittauspisteet syötetään ohjelmistoon. Ohjelmiston kalibrointityökalulla mittaustulokset analysoidaan ja mahdolliset poikkeamat kompensoidaan. Pienissä robottisoluissa riittää yleensä työkalupisteen paikan ja työkappaleen aseman kalibrointi. Suurissa soluissa joudutaan kalibroimaan lisäksi robotin peruskoordinaatisto, työkalu sekä mahdolliset jigit, pyörityspöydät ja servoradat. (Kuivanen 1999, 86–87.)

Poikkeamat voivat johtua esimerkiksi työkappaleiden ja muiden solun komponenttien sijainnin ja asennon eroavaisuuksista todellisen ja virtuaalisen robottisolun välillä. Robot-

tiin liittyviä poikkeamia ovat muun muassa robotin peruskoordinaatiston asema- ja asentovirheet suhteessa robotin alustaan. Nämä voivat johtua esimerkiksi välyksistä robotin rakenteessa. (Bergström 2011, 8.)

Kalibrointi voidaan suorittaa manuaalisesti käyttämällä robottia mittalaitteena. Robotin työkalulaippaan kiinnitetään kartiomainen työkalu tai piikki, jonka avulla robottisolusta määritetään riittävä määrä pisteitä. Manuaalisen kalibroinnin hyviä puolia on sen yksinkertaisuus ja edullisuus, sillä erillisiä mittalaitteita ei tarvita. Huono puoli on menetelmän epätarkkuus. Kalibroinnin tulos saattaa vaihdella riippuen siitä, kuka kalibroinnin suorittaa. Monissa tapauksissa manuaalinen kalibrointi on kuitenkin riittävän tarkka. Kalibrointi voidaan suorittaa myös käyttämällä robotin sijaan erillistä ulkoista mittalaitetta. Erilaisia vaihtoehtoja on paljon ja mittausmenetelmät voivat perustua laseriin, erilaisiin antureihin ja lähestymiskytkimiin, kameroihin ja niin edelleen. Erillisillä mittalaitteilla päästään manuaalista kalibrointia tarkempiin tuloksiin, mutta huonona puolena on mittalaitteiden investoinneista johtuvat ylimääräiset kustannukset. (Bergström 2011, 19–26.)

RobotExpertillä on mahdollista käyttää robotilta saatuja mittaustietoja simulointimallin kalibrointiin. Kalibrointityökalu vertaa oikealla ja virtuaalisella robotilla luotuja pisteitä ja säätää niiden sijaintia sekä asentoa siten, että pisteiden keskimääräinen etäisyys on mahdollisimman pieni. (Siemens PLM Software 2017a). Varsinaista kalibrointia ei tässä työssä luodulle simulointimallille tehty, koska testiohjelmat eivät vaatineet suurta tarkkuutta. RobotExpertin kalibrointityökalun toimintaa kuitenkin kokeiltiin ja ohje työkalun käyttöön löytyy liitteestä 1.

5.3 Liikepisteiden luominen, OLP-käskyjen lisääminen ja simulointi

Offline-ohjelmointia testattiin luomalla ohjelmistolla yksinkertainen ohjelma, jonka tarkoituksena on hakea lavalla olevasta pinosta ohutlevy ja viedä se toiselle lavalle. Ohjelmassa käytetään alipainevahtia eli jokaista hakupistettä ei tarvitse opettaa erikseen, vaan hakupisteeksi määritetään levypinon pohja ja liike pysähtyy, kun alipainevahti aktivoituu, eli kun imukuppitarttuja on tarttunut levyyn. Robotti palaa kotiasemaan ja ohjelma päättyy, kun alipainevahti ei aktivoidu, eli kun pinossa ei enää ole levyjä.

Ohjelmistolla luotiin operaatio, johon lisättiin tarvittavat liikepisteet ja OLP-käskyt Yaskawan INFORM-ohjelmointikielellä. Simuloinnissa ei otettu OLP-logiikkaa huomioon, joten pelkästään robotin liikkeet simuloitiin. Simuloinnin aikana tehtiin törmäystarkastelu, jonka

tuloksena liikepisteitä muokattiin, jotta robotti ei törmäisi paikoitustelineeseen ja särmäyspuristimeen. Luotu operaatio on esitetty kuvassa 6.

Paths & Locations	X	Y	Z	RX	RY	RZ	Motion...	Joint Sp...	Speed	OLP Commands
Lavaus								0.00	0.00	
via	1892.71	-708.58	1694.50	-41.67	-90.00	0.00	Joint	100.00	0.00	
via1	2113.66	1194.97	361.85	0.03	1.00	0.33	Joint	100.00	0.00	*LABEL1
via2	2113.66	1194.97	361.85	0.03	1.00	0.33	Joint	100.00	0.00	CALL JOB:IMUT-ON
via3	2116.44	1195.08	202.88	0.03	-0.00	0.33	Linear	0.00	40.00	JUMP *LABEL3 IF IN#(1)=OFF
via4	2116.42	1194.86	568.87	0.03	-0.00	0.33	Linear	0.00	40.00	
via5	-2517.23	-486.06	527.69	0.03	0.02	-179.67	Joint	100.00	0.00	
via6	-2517.15	-486.20	257.69	0.03	0.02	-179.67	Linear	0.00	40.00	CALL JOB:IMUT-OFF
via7	-2517.24	-486.06	536.69	0.03	0.02	-179.67	Linear	0.00	40.00	JUMP *LABEL1 *LABEL3 CALL JOB:IMUT-OFF
via8	1892.69	-708.60	1711.00	-41.67	-90.00	0.00	Joint	100.00	0.00	CALL JOB:KOTI-AS

Kuva 6. Operaatio lavausohjelmaa varten.

Kuvassa nähdään liikepisteiden koordinaatit, liiketyyppi (nivel- tai lineaariliike), nopeus sekä kunkin liikepisteen jälkeen toteutettavat käskyt.

5.4 Ohjelman testaaminen robotilla

Operaatiosta tehtiin Motoman-robotille sopiva ohjelma eli JBI-tiedosto. Luotu ohjelma on esitetty kuvassa 7. RobotExpert-ohjelmisto ei tunnistanut UNTIL-käskyä, joten UNTIL IN#(1)=ON -kohta lisättiin ohjelmaan muokkaamalla tiedostoa Notepad-tekstieditorilla.

```
/JOB
//NAME Lavaus
//POS
///NPOS 9,9,0,0,0,0
///TOOL 1
///POSTYPE PULSE
///PULSE
C00000=0,0,0,501,0,150471
C00001=20149,36966,-66253,-386,-28270,-4968
C00002=20149,36966,-66253,-386,-28270,-4968
C00003=20209,52077,-65152,-72,-20251,-5105
C00004=20194,19668,-63428,-42,-37410,-5117
C00005=-456,98613,83486,-33,-57336,-12419
C00006=-449,111722,81439,-36,-49845,-12418
C00007=-457,98285,83794,-33,-57629,-12419
C00008=0,0,0,501,0,-234
///TOOL 0
BC00000=178476
BC00001=124318
BC00002=124318
BC00003=124318
BC00004=124318
BC00005=-162721
BC00006=-162721
BC00007=-162721
BC00008=178476
//INST
///DATE 2017/01/11 11:47
///COMM
///ATTR SC,RW
///GROUP1 RB1,BS1
NOP
MOVJ C00000 BC00000 VJ=100.00 PL=0
MOVJ C00001 BC00001 VJ=100.00 PL=0
*LABEL1
MOVJ C00002 BC00002 VJ=100.00 PL=0
CALL JOB:IMUT-ON
MOVL C00003 BC00003 V=40.00 PL=0 UNTIL IN#(1)=ON
JUMP *LABEL2 IF IN#(1)=OFF
MOVL C00004 BC00004 V=40.00 PL=0
MOVJ C00005 BC00005 VJ=100.00 PL=0
MOVL C00006 BC00006 V=40.00 PL=0
CALL JOB:IMUT-OFF
MOVL C00007 BC00007 V=40.00 PL=0
JUMP *LABEL1
*LABEL2
CALL JOB:IMUT-OFF
MOVJ C00008 BC00008 VJ=100.00 PL=0
CALL JOB:KOTI-AS
END
```

Kuva 7. Lavausohjelma.

Ohjelma vietiin robotille muistikortilla ja ajettiin vaihe vaiheelta käsiajolla, pienellä nopeudella, jotta mahdolliset virheet eivät johtaisi törmäykseen. Virheitä ei ollut ja ohjelma toimi oikein. Ensin robotti liikkui lavan yläpuolelle ja CALL JOB -käskyllä kutsuttiin imukuppitarttujat päälle laittava ohjelma, jonka jälkeen robotti lähti laskeutumaan kohti levypinon pohjaa, kunnes se tarttui ensimmäiseen levyyn eli tulo 1 aktivoitui [IN#(1)=ON]. Tämän jälkeen robotti liikkui toisen lavan päälle, laskeutui lähemmäs lavaa ja CALL JOB -käskyllä kutsuttiin imukuppitarttujat pois päältä laittava ohjelma. Robotti nousi kauemmas lavan pinnasta ja JUMP-käskyllä hypättiin takaisin ohjelman alkuun kohtaan LABEL1. Kun levyjä ei enää ollut pinossa eli tulo 1 ei aktivoitunut [IN#(1)=OFF], ennen kuin robotti saavutti levypinon pohjan, hypättiin JUMP-käskyllä kohtaan LABEL2. Imukuppitarttuja laitettiin pois päältä, robotti nousi kauemmas lavasta ja kutsuttiin ohjelma, joka ajoi robotin kotiasemaan.

6 ROBOTEXPERTIN KÄYTTÖ KONEAUTOMAATIO-OPETUKSESSA

Robottisimulointia ja offline-ohjelmointia opetellaan Turun ammattikorkeakoulussa pääasiassa ABB:n RobotStudio-ohjelmiston avulla. Koska ammattikorkeakoululla ei ole käytössä ABB:n robotteja, RobotStudiolla luotuja ohjelmia ei ole päästy siirtämään oikealle robotille. Pääasiassa tästä syystä RobotExpert-ohjelmisto kannattaa ottaa osaksi koneautomaatio-opetusta, koska se tukee useiden eri valmistajien robotteja. RobotExpertillä luotuja ohjelmia pystytään siirtämään muun muassa ammattikorkeakoululla käytössä oleville Fanucin ja Motomanin roboteille.

Yhtenä huonona puolena opiskelijoiden kannalta voidaan pitää sitä, että RobotExpert-ohjelmisto ei sisällä valmista robotti- tai komponenttikirjastoa, vaan kaikki 3D-mallit pitää tuoda ohjelmistoon omista tiedostoista. Joidenkin robottivalmistajien, kuten ABB:n, sivuilta löytyy valmiita robottien malleja, mutta muussa tapauksessa malleja täytyy pyytää robottien tai RobotExpertin toimittajilta. Tätä työtä varten on saatu kahden Turun ammattikorkeakoulun käytössä olevan robotin mallit ja nämä voidaan antaa opiskelijoiden käyttöön. Muissa tapauksissa malleja täytyy pyytää edellä mainituilta toimittajilta. Suurin osa koneautomaatiotekniikan opiskelijoista on todennäköisesti käynyt 3D-mallinnuskurssin ennen robotiikan kursseja, joten sinänsä työkalujen ja muiden solun komponenttien mallintaminen ja tuominen RobotExpertiin ei ole ongelma. Mallintamiseen kuluu toki hieman aikaa, eikä ohjelmiston käyttöä pääse kokeilemaan heti.

RobotExpert-ohjelmistosta ei löydy paljon viitteitä Internetistä, kuten käyttäjien kokemuksia, esimerkkejä tai muuta ohjelmiston käyttöä tukevaa materiaalia. Lisäksi esimerkiksi robottikohtaisten ohjainten mukana tulleet ohjeet olivat Process Simulate -ohjelmistoa varten. Myös ohjelmiston omassa ohjeessa viitattiin välillä Process Simulateen RobotExpertin sijaan. RobotExpert on riisuttu versio Process Simulate -ohjelmistosta, joten on vaikea sanoa, ovatko kaikki ohjeissa mainitut ominaisuudet käytössä RobotExpertissä vai ainoastaan Process Simulatessa. Ohjelmiston oma englanninkielinen ohje on kuitenkin melko kattava ja Siemens PLM Community -foorumille kirjauduttuaan opiskelijat voivat kysyä neuvoa ohjelmiston muilta käyttäjiltä. Lisäksi opiskelijat voivat käyttää tässä opinnäytetyössä tehtyä suomenkielistä ohjetta ohjelmiston peruskäytön opetteluun.

Koska RobotExpert on yleiskäyttöinen, monien eri valmistajien robotteja tukeva ohjelmisto, on ymmärrettävää, että sen toiminnot ovat myös yleiskäyttöisiä. Esimerkiksi Motomanin robottien kanssa olisi todennäköisesti järkevämpää käyttää Yaskawan MotoSim-ohjelmistoa, joka on suunniteltu juuri Motomanin robotteja varten. Tällöin opiskelijoille tulisi realistisempi kuva Motoman-robottien ohjelmoimisesta. RobotExpertin etu on siinä, että opiskelijat voivat saman ohjelmiston avulla työskennellä eri robottimerkkien kanssa. Koska opiskeluaika on rajallinen, on käytännössä hyvin vaikeaa ehtiä opetella käyttämään monia eri ohjelmistoja. Toisaalta kun hallitsee yhden ohjelmiston, on yleensä helpompi oppia käyttämään myös muita samaan tarkoitukseen suunniteltuja ohjelmistoja.

ABB:n RobotStudioon verrattuna RobotExpertissä on yksi suurelta tuntuva puutos. RobotExpertin simulaatio perustuu aikaan: kun yksi operaatio päättyy, toinen alkaa. Simulaatio voi johtaa vain yhteen ennalta määrättyyn lopputulokseen. Simulointia ei siis voida ohjata signaaleilla eikä älykkäiden komponenttien tai antureiden luominen ole mahdollista. Tämä ei välttämättä ole opiskelijoiden kannalta suuri ongelma, mutta aiempi kokemus ABB:n RobotStudiosta asetti tietynlaisia odotuksia RobotExpertiä kohtaan. RobotStudioon verrattuna RobotExpert tuntui ajoittain liian yksinkertaiselta ohjelmistolta, jolla ei pystytä tekemään yhtä monimutkaisia simulaatioita kuin RobotStudiolla.

Koska ABB:n RobotStudio on Turun ammattikorkeakoululle tuttu ohjelmisto, on luultavasti järkevää jatkaa sen käyttöä ja ottaa RobotExpert vaiheittain osaksi opetusta esimerkiksi siten, että aluksi vain pienempi ryhmä perehtyy ohjelmiston käyttöön. RobotExpert sopii paremmin sellaisille opiskelijoille, jotka ovat jo käyttäneet robottia ja osaavat jonkin verran ohjelmointia. Ohjelmistosta ei esimerkiksi löydy kaikkia Motomanin robotille ominaisia käskyjä, vaan ne täytyy itse kirjoittaa. Tämä edellyttää tietysti sitä, että opiskelija tuntee nämä käskyt ja tietää niiden toiminnan.

RobotExpert-ohjelmistolla opiskelijat voivat suunnitella robottisoluja ja tarkistaa, että robotti ulottuu määrättyihin liikepisteisiin sekä varmistaa, ettei robotti törmää työkierron aikana solun muihin komponentteihin. Opiskelijat voivat myös harjoitella ammattikorkeakoulun käytössä olevien robottisolujen mallinnusta. Opiskelijoiden kannalta tärkein RobotExpertin ominaisuus on mahdollisuus ladata ohjelmia oikealle robotille. Tällä tavoin voidaan tarkistaa, että ohjelma toimii oikein ja robottisolun mallista on tehty tarpeeksi tarkka. Mallin tarkkuutta voi arvioida myös siirtämällä robotilla tehtyjä ohjelmia RobotExpertiin. Jos liikepisteet ovat ohjelmistossa samassa kohtaa kuin oikealla robotilla, malli on tarpeeksi tarkka. Robotilta siirrettyjä ohjelmia voidaan myös muokata ohjelmistossa

ja ladata uudelleen robotille. Tätä ominaisuutta voidaan hyödyntää muun muassa silloin, kun oikeilla robottisoluilla on ruuhkaa. Turun ammattikorkeakoulun käytössä olevia robotteja käyttävät muutkin kuin ammattikorkeakoulun opiskelijat, joten RobotExpertillä ohjelmointi onnistuu silloinkin, kun robotit ovat varattuja. Opiskelijat voivat aloittaa ohjelmoinnin robotilla ja jatkaa RobotExpertillä tai päinvastoin. Työaikaa robotilla voidaan lyhentää, jos opiskelijat tekevät suurimman osan ohjelmoinnista RobotExpertillä. Näin useampi opiskelija ehtii päivän aikana työskentelemään oikealla robotilla.

RobotExpertin käytön opettelu vaatii jonkin verran aikaa, joten olisi hyvä, että opiskelijat voisivat käyttää sitä myös kotona. ABB:n RobotStudio-ohjelmiston opiskelijat pystyvät lataamaan ilmaiseksi omille tietokoneilleen. Siemens PLM Softwaren sivuilta pystyy lataamaan RobotExpertin koekäyttöön 30 päiväksi. Ilmainen kokeiluversio ei kuitenkaan sisällä OLP-rajapintoja, joten robottikohtaisten ohjainten käyttö ei ole mahdollista. Olisi siis hyvä, että ohjelmiston lisenssi sallisi etäkäytön, jotta opiskelijat voisivat työskennellä ohjelmiston parissa myös ammattikorkeakoulun ulkopuolella.

7 PÄÄTELMÄT

Työn tavoitteena oli tutustua RobotExpert-ohjelmiston käyttöön ja selvittää, sopiiko ohjelmisto simuloinnin ja offline-ohjelmoinnin opetukseen Turun ammattikorkeakoulussa. Työn tuloksena saatiin käyttöohje ohjelmiston peruskäyttöä varten sekä ohjeistus ohjelman lataamiseen ohjelmistosta oikealle robotille ja robotilla tehdyn ohjelman siirtämiseen ohjelmistoon. Ohjelmistossa on hyvät ja huonot puolensa verrattuna opetuksessa tähän asti käytettyyn ABB:n RobotStudio-ohjelmistoon. Tällä hetkellä merkittävin hyöty RobotStudioon nähden on mahdollisuus ladata RobotExpertillä tehtyjä ohjelmia Turun ammattikorkeakoulun käytössä oleville roboteille ja siirtää roboteilla tehtyjä ohjelmia ohjelmistoon. Jos ammattikorkeakoululla olisi ABB:n robotteja, tilanne voisi olla toinen, mutta toistaiseksi RobotExpertiä kannattaa hyödyntää koneautomaatio-opetuksessa ainakin jossain laajuudessa.

Ohjelmiston käytön opetteluun ja käyttöohjeen kirjoittamiseen meni tätä työtä tehtäessä paljon aikaa. Tästä syystä oikealla robotilla työskentely jäi turhan vähälle. Lisäksi työn loppuvaiheilla särmäyssolu oli usein varattu muuhun käyttöön, joten robotille pääsy oli vaikeaa. Tämän työn aikana testattiin lähinnä robotin liikkeiden simulointia ja ohjelmointia. Olisi ollut hyvä ehtiä kokeilla monimutkaisempien ohjelmien tekemistä. Ajan ja allekirjoittaneen XML-tietotaidon puutteen vuoksi omien OLP-käskyjen luominen jäi myös kokeilematta.

Kaikkiin RobotExpert-ohjelmiston ominaisuuksiin ei siis ehditty perehtyä ja aiheesta pystyisi tekemään vielä toisen jos kolmannenkin opinnäytetyön. Jatkossa kannattaa ainakin tutustua Turun ammattikorkeakoulun käytössä olevan robottihitsausaseman Motoman-robotin sekä kappaleenkäsittelyyn tarkoitetun Fanuc-robotin simulointiin ja offline-ohjelmointiin RobotExpertillä. Särmäyssolun malli tehtiin melko suurpiirteisesti lähinnä törmäystarkastelua varten. Tähän työhön tarkkuus oli riittävä, mutta mitoitukseen ja mallinnukseen olisi voinut käyttää enemmän aikaa. Solua ei myöskään kalibroitu. Jatkossa solusta voisi tehdä tarkemman mallin, jolla voitaisiin luoda särmäysohjelmia ja suorittaa esimerkiksi työkalun vaihtoja tai muita suurempaa tarkkuutta vaativia töitä. Lisäksi simulointimallien kalibrointiin voisi perehtyä tarkemmin.

RobotExpertin omia mallinnustyökaluja ei käytetty tässä työssä, vaan kappaleet mallinnettiin SolidWorks-ohjelmistolla. Yksinkertaisten kappaleiden, kuten laatikoiden, mallin-

tamiseen RobotExpert voisi sopia, mutta monimutkaisempien mallien tekemiseen ohjelmiston työkalut ovat luultavasti liian yksinkertaiset. Opiskelijat voisivat kuitenkin perehtyä mallinnustyökalujen käyttöön ja selvittää, onko niiden käyttö joissain tilanteissa järkevämpää kuin ulkopuolisella mallinnusohjelmalla mallintaminen.

Tässä työssä ei käytetty RCS-moduulia eikä niiden käyttö jatkossakaan ole välttämätöntä. Opiskelijat voisivat kuitenkin tutkia, kuinka tarkkoja töitä RobotExpertillä pystytään tekemään ilman RCS-moduulia. Ideaalitilanteessa ammattikorkeakoululle hankittaisiin kalibrointiin tarkoitettu mittalaite. Tämä ei välttämättä ole taloudellisesti kannattavaa, joten opiskelijat voisivat selvittää, kuinka tarkkoja simulointimalleja pystytään luomaan ilman kalibrointia. Toisaalta voitaisiin myös kehittää omia mittausmenetelmiä kalibrointia varten.

Simuloinnin ja offline-ohjelmoinnin tarkkuuden lisäksi olisi hyvä perehtyä omien OLP-käskyjen määrittämiseen. Lisäksi olisi syytä kokeilla laajemmin erilaisia robotin käskyjä ja varmistaa niiden toiminta. Jos vaikuttaa siltä, että RobotExpert ei ole tarpeeksi monipuolinen ammattikorkeakoulun tarpeisiin, voitaisiin harkita myös Siemensin Process Simulate -ohjelmiston käyttöönottoa.

LÄHTEET

ABB 2017. What is a coordinate system? Viitattu 4.1.2017 http://developercenter.robotstudio.com/BlobProxy/manuals/IRC5FlexPendantOpManual/doc210.html.

Alander, J. & Niemi, K. 1987. Robottien ohjaus ja ohjelmointi. Espoo: Otakustantamo.

Bergström, G. 2011. Method for calibration of off-line generated robot program. Master of Science Thesis. Department of Automatic Control. Göteborg: Chalmers University of Technology. Viitattu 23.1.2017 http://publications.lib.chalmers.se/records/fulltext/153281.pdf.

Delfoi 2017. Robotiikkaa piensarjoille. Viitattu 2.1.2017 http://www.delfoi.com/web/solutions/robotiikka/fi_FI/piensarjoille/.

Fabryka Robótow 2012. Siemens RobotExpert – successor of Robcad? Viitattu 4.1.2017 http://fabryka-robotow.pl/2012/08/siemens-robotexpert-successor-of-robcad/.

IDEAL PLM 2017. RobotExpert. Viitattu 4.1.2017 http://www.ideal.fi/fi/tuotteet/manufacturing-process-management/tecnomatix/robotexpert/.

Keinänen, T.; Kärkkäinen, P.; Lähetkangas, M. & Sumujärvi, M. 2007. Automaatiojärjestelmien logiikat ja ohjaustekniikat. 1. painos. Helsinki: WSOY Oppimateriaalit Oy.

Konopa, M. 2013. Simulation of Production Processes. Diploma Thesis. Faculty of Electrical Engineering. Praha: Czech Technical University in Prague. Viitattu 12.1.2017 https://support.dce.felk.cvut.cz/mediawiki/images/f/fa/Dp_2013_konopa_miroslav.pdf.

Kuivanen, R. (toim.) 1999. Robotiikka. Helsinki: Talentum Oyj/MetalliTekniikka.

Kuusinen, J. 2003. Satakunta kaivaa robotit nurkista tehokäyttöön. Tekniikka&Talous. Viitattu 2.1.2017 http://www.tekniikkatalous.fi/tekniikka/metalli/2003-04-24/Satakunta-kaivaa-robotit-nurkista-tehok%C3%A4ytt%C3%B6%C3%B6n-3288501.html.

Liikenne- ja viestintäministeriö. 2016. Robotiikan taustaselvityksiä. Liikenne- ja viestintäministeriön julkaisuja 2/2016. Viitattu 2.1.2017 http://urn.fi/URN:ISBN:978-952-243-470-8.

Malm, T. (toim.) 2008. Vuorovaikutteisen robotiikan turvallisuus. Helsinki: Suomen Robotiikkayhdistys ry.

Nof, S. (toim.) 1999. Handbook of Industrial Robotics. 2. painos. New York: John Wiley & Sons, Inc.

OpenTheFile 2016. Viitattu 12.1.2017 http://www.openthefile.net/extension/jbi.

Pan, Z.; Polden, J.; Larkin, N.; Van Duin, S. & Norrish, J. 2011. Recent progress on programming methods for industrial robots. Robotics and Computer-Integrated Manufacturing. Vol. 28, No 2/2012, 87–94.

Rainio, H. 2014. Robottien etäohjelmoinnin kypsyystaso. Opinnäytetyö. Kone- ja tuotantotekniikan koulutusohjelma. Turku: Turun ammattikorkeakoulu. Viitattu 2.1.2017 https://www.theseus.fi/handle/10024/77610.

Shannon, R. 1998. Introduction to the art and science of simulation. Proceedings of the 1998 Winter Simulation Conference. Viitattu 26.10.2016 http://cecs.wright.edu/~fciarall/ISE195/Readings/ShannonSimulationART.pdf.

Siemens PLM Software 2017a. RobotExpert Reference Manual. Viitattu 5.1.2017 https://docs.plm.automation.siemens.com/tdoc/tecnomatix/13.0.2/RBX/#uid:index_xid1015930.

Siemens PLM Software 2017b. RobotExpert – Plug-n-Play software solution for robotics simulation and programming. Viitattu 5.1.2017 www.plm.automation.siemens.com > Solutions by Product Line > Tecnomatix > Manufacturing Simulation & Validation > Robotics Simulation & Programming > RobotExpert > Product Information.

Siemens PLM Software 2017c. Robotics Customized UI Manual. Viitattu 12.1.2017 https://docs.plm.automation.siemens.com/tdoc/tecnomatix/13.0.2/RBX#uid:index_xid1092354:xid1077734.

Wikipedia 2014. Tuotteen elinkaaren hallinta. Viitattu 12.1.2017 https://fi.wikipedia.org/wiki/Tuotteen_elinkaaren_hallinta.

Wikipedia 2015. Ohjelmoitava logiikka. Viitattu 16.1.2017 https://fi.wikipedia.org/wiki/Ohjelmoitava_logiikka.

Wikipedia 2017. Off-line programming (robotics). Viitattu 12.1.2017 https://en.wikipedia.org/wiki/Off-line_programming_(robotics).

RobotExpert-ohjelmiston käyttöohje

Sisältö

1. Yleistä ohjelmistosta

Ohjetta tehtäessä on käytetty ohjelmiston versiota 13.0.2. Muilla versioilla ohjelmiston ulkonäkö tai toiminnot saattavat hieman poiketa tästä ohjeesta.

1.1 Tiedostomuodot

RobotExpert-ohjelmisto ei sisällä valmiita robottien ja muiden komponenttien 3D-malleja, vaan mallit tuodaan ohjelmaan käyttäjän omista tiedostoista. Joidenkin robottivalmistajien sivuilta löytyy valmiita malleja ladattavaksi. Ohjelmisto käyttää COJT-tiedostomuotoa ja tässä muodossa olevat tiedostot voidaan tuoda suoraan ohjelmaan. Seuraavissa tiedostomuodoissa olevat tiedostot voidaan ohjelmiston avulla muuttaa COJT-muotoon:

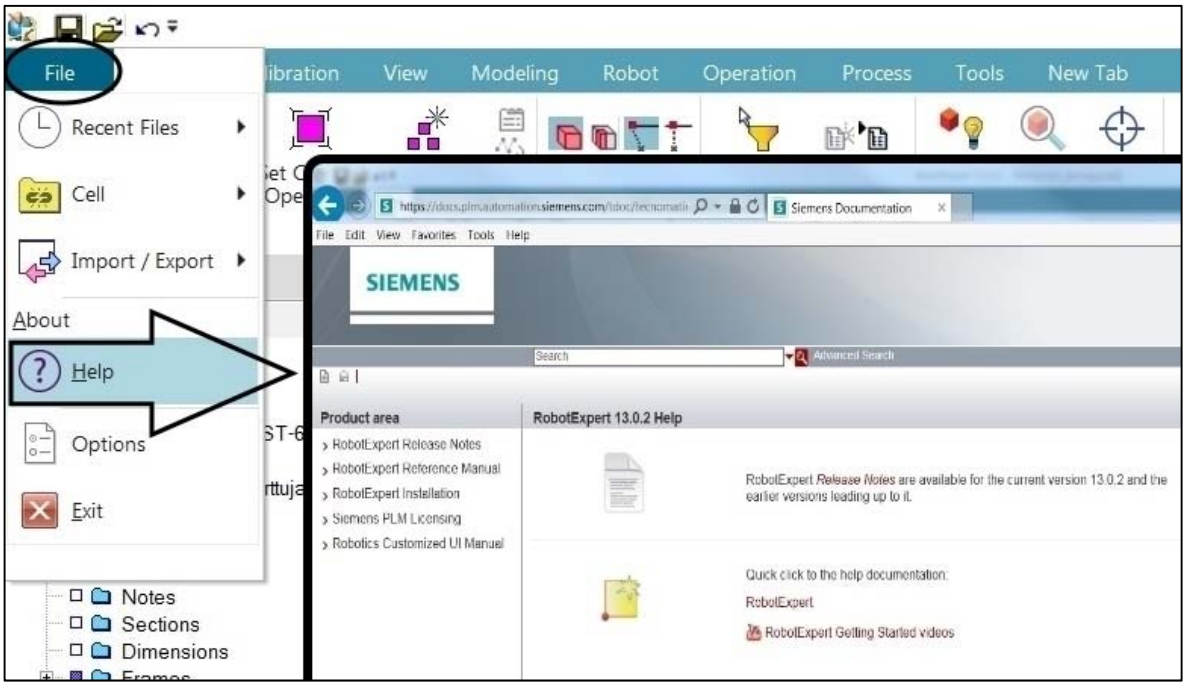
- NX (.prt)
- JT (.jt)
- Solid Edge (.par; .psm; .asm)
- SolidWorks (.sldprt; .sldasm)
- Pro/ENGINEER (.prt; .xpr;.asm)
- CATIA V4/V5 (.model; .CATPart; .CATProduct)
- Parasolid Ascii (.x_t; .xmt_txt)
- STEP (.stp; .step)
- IGES (.igs; .iges)
- Tecnomatix Components (.co; .cojt)

1.2 Tutoriaalit, Siemens PLM Community ja Help

Kun ohjelmisto käynnistetään, avautuu kuvan 1 mukainen ikkuna, josta löytyy tutoriaaleja ohjelmiston käytöstä sekä linkki Siemens PLM Community -foorumille, josta löytyy lisätietoa, ja johon kirjauduttuaan voi esittää kysymyksiä muille ohjelmiston käyttäjille.

Kuva 1. Tutoriaalit ja Siemen PLM Community.

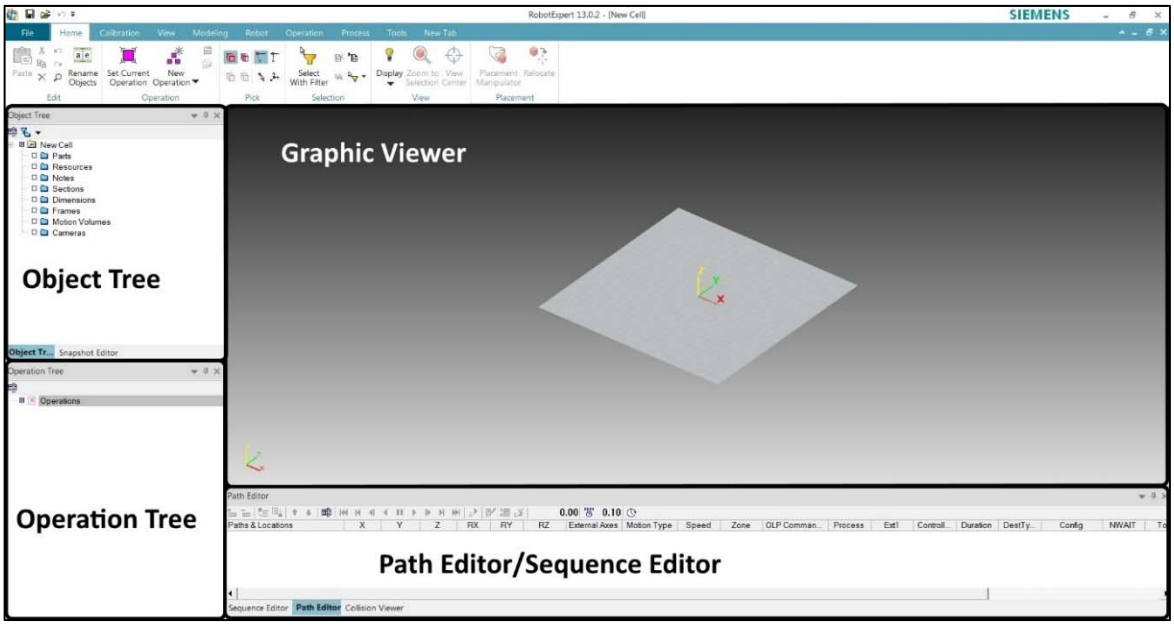
Tarkempaa tietoa ohjelmiston eri toiminnoista löytyy valitsemalla **File → Help**, jolloin selaimeen avautuu kuvan 2 mukainen ohje. Tästä ohjeesta löytyy myös linkit tutoriaalivideoihin ja Siemens PLM Community -foorumille.



Kuva 2. Ohjelmiston käyttöohje.

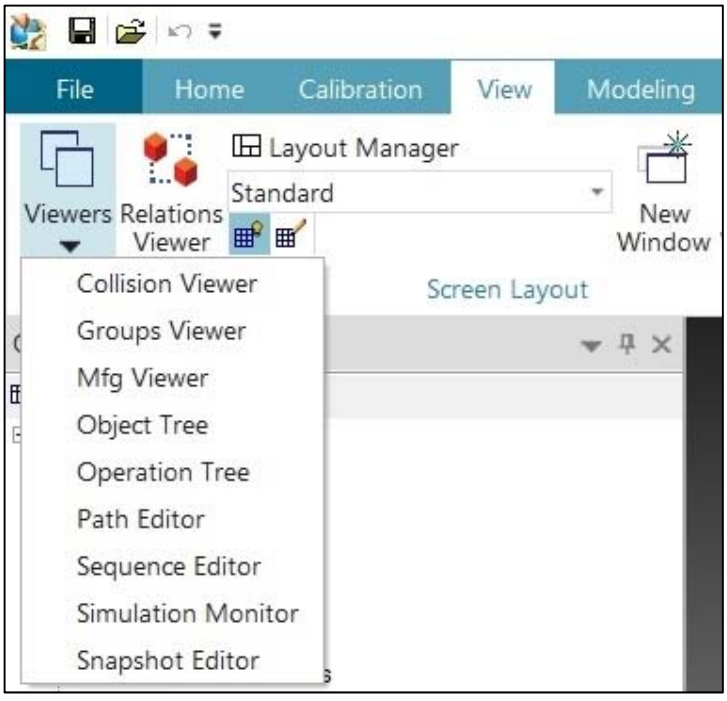
1.3 Viewers

Ohjelmistossa on erilaisia ”**Viewer**eitä”. Peruskäytössä tärkeimpinä mainittakoon kuvassa 3 esitetyt **Graphic Viewer**, **Object Tree** (objektipuu), **Operation Tree** (operaatiopuu), **Path Editor** ja **Sequence Editor**. **Graphic Viewer** näyttää kolmiulotteisena ohjelmistoon tuodut komponentit. Objektipuussa näkyy solun eri komponentit ja operaatiopuussa puolestaan luodut operaatiot. **Path** ja **Sequence Editor**ien avulla pystytään muokkaamaan luotuja operaatioita.



Kuva 3. Graphic Viewer, Object Tree, Operation Tree, Path Editor ja Sequence Editor.

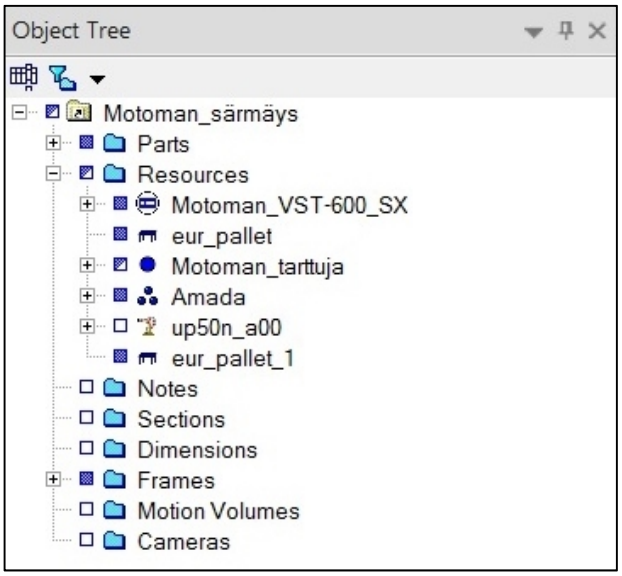
Viewereitä voidaan sulkea ja ottaa tarvittaessa uudelleen käyttöön avaamalla **View**-välilehdeltä **Screen Layout** -kohdasta **Viewers**-pudotusvalikko kuten kuvassa 4.



Kuva 4. Viewereiden lisääminen.

Objekti- ja operaatiopuussa objektien (robottien, muiden komponenttien, koordinaatistojen, liikepisteiden jne.) edessä näkyy kuvan 5 mukainen pieni neliö, joka kertoo objektin näkyvyydestä **Graphic Viewer**issä. Jos neliö on sininen, objekti on

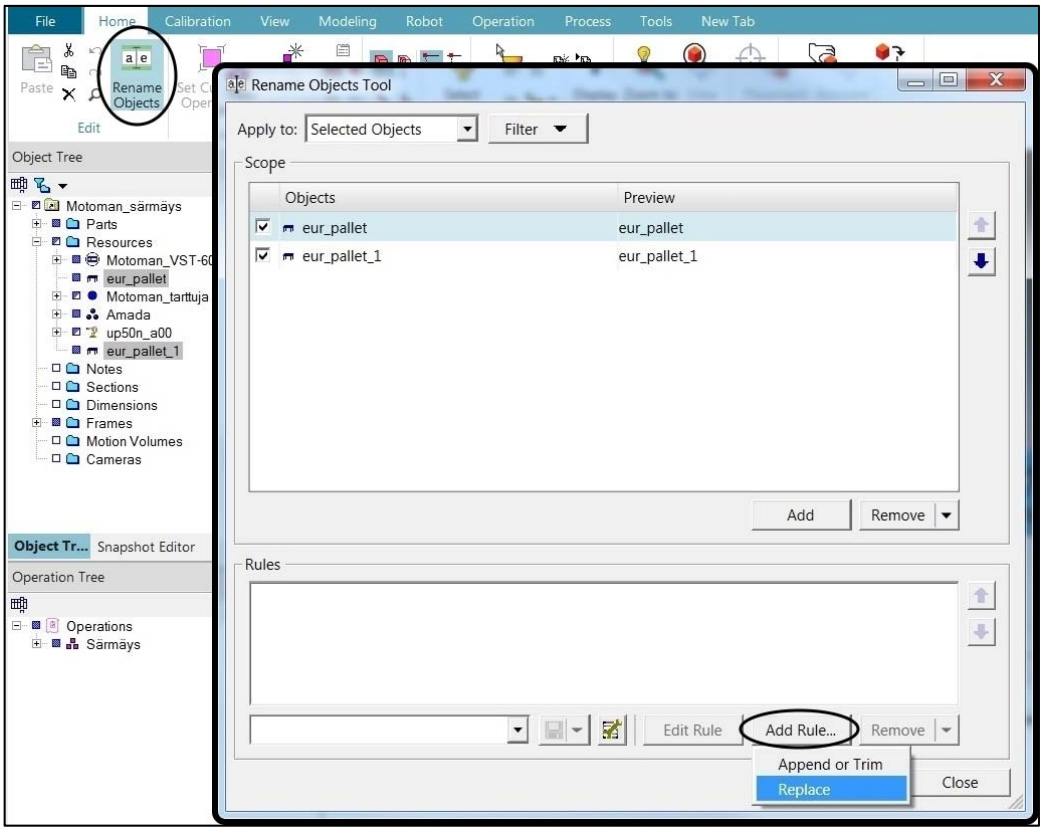
näkyvillä, ja jos neliö on valkoinen, objekti on piilotettu. Jos neliö on puoliksi sininen, osa objektista on näkyvillä. Objektin saa piiloon ja takaisin näkyviin klikkaamalla neliötä.



Kuva 5. Objektien näkyminen ja piilottaminen.

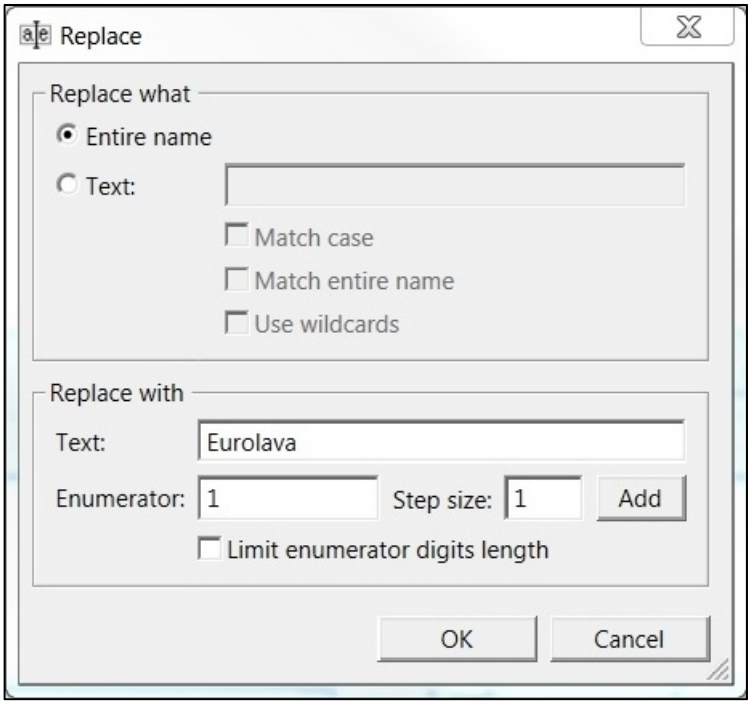
1.4 Objektien uudelleennimeäminen

Jos komponentin, operaation, liikepisteen jne. haluaa nimetä uudelleen, se onnistuu tuplaklikkaamalla nimeä tai klikkaamalla ja painamalla F2. Jos haluaa nopeasti nimetä uudelleen useampia objekteja, se onnistuu valitsemalla kyseiset objektit aktiivisiksi klikkaamalla niitä Ctrl pohjassa ja valitsemalla **Home**-välilehdeltä **Edit**-kohdasta **Rename Objects**, jolloin avautuu kuvan 6 mukainen ikkuna.



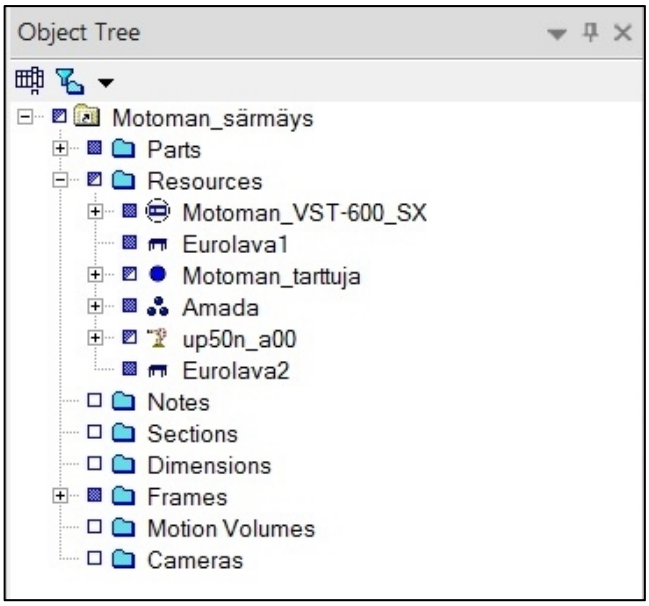
Kuva 6. Objektien uudelleennimeäminen.

Valitaan **Add Rule…→ Replace**, jolloin avautuu kuvan 7 mukainen ikkuna.



Kuva 7. Vanhojen nimien korvaaminen ja numerointi.

Replace what -kohtaan valitaan **Entire name**, jos halutaan muuttaa koko nimi tai **Text**, jos halutaan muuttaa osa nimestä. **Replace with** -kohtaan kirjoitetaan uusi nimi. Jos kappaleet halutaan numeroida, **Enumerator**-kohtaan määritetään, mistä luvusta lähdetään liikkeelle ja **Step size** -kohtaan, kuinka suurella hyppäyksellä numerointi etenee → **Add** → **OK** → **Apply** → **Close**, jolloin objektit saavat kuvan 8 mukaiset nimet.

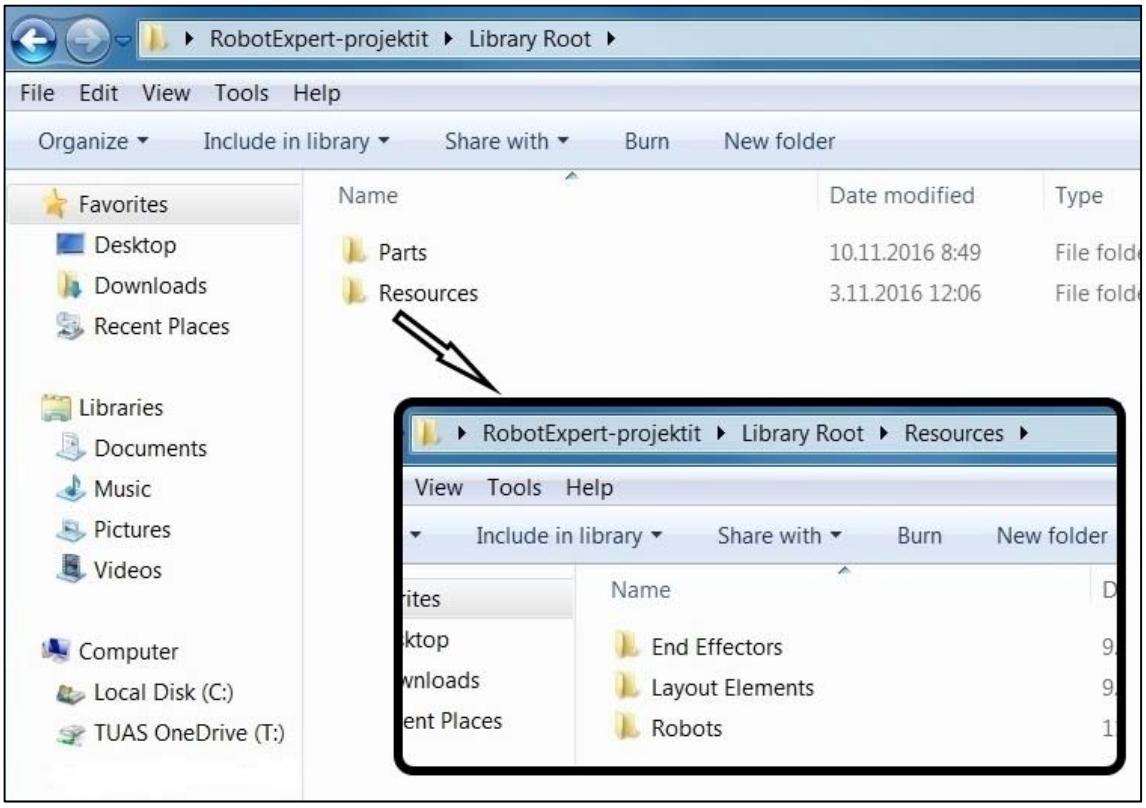


Kuva 8. Uudelleennimeämisen tulos.

2. Library Root -hakemiston luominen

Ennen ohjelmiston käytön aloittamista tulee tietokoneelle luoda Library Root -hakemisto, johon ohjelmiston käyttämät 3D-mallit (robotit, työkalut, pyörityspöydät, työstettävät kappaleet jne.) tallennetaan sen jälkeen, kun ne on muutettu COJT-muotoon. Ohjelmiston asetuksiin määritetään Library Root -hakemistopolku, jonka avulla tarvittavat tiedostot löytyvät.

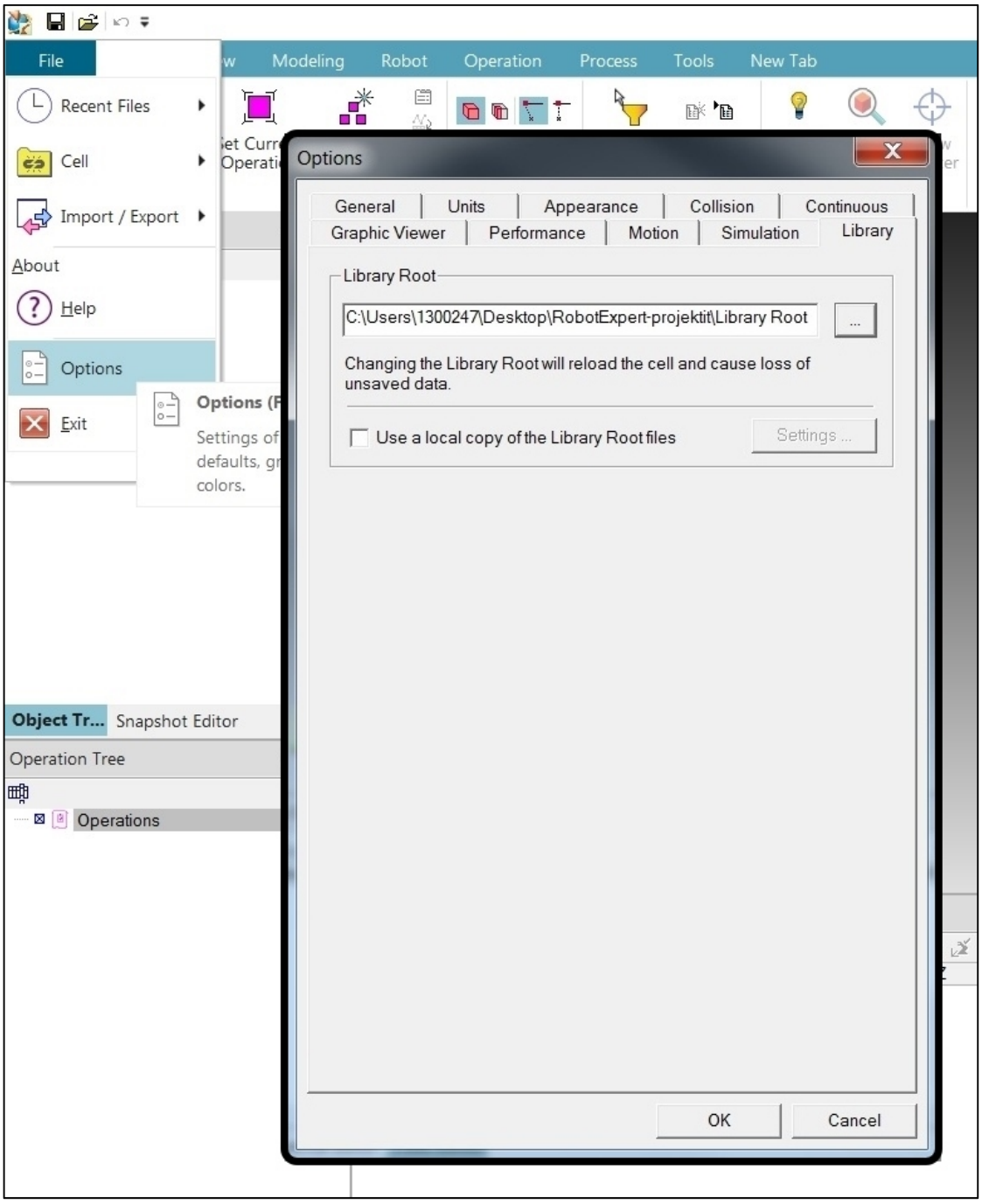
Käytännössä siis tietokoneelle luodaan kansio sellaiseen paikkaan, johon käyttäjällä on luku- ja kirjoitusoikeus. Kansiolle kannattaa selvyyden vuoksi antaa nimi Library Root. Kansion sisään voi luoda lisää kansioita oman mielensä mukaan, mutta ohjelmiston tutoriaalit suosittelevat kuvan 9 mukaisesti Resource- ja Parts-nimisten kansioiden luomista. Resource-kansion sisään voi edelleen luoda End Effectors-, Layout Elements- ja Robots-kansiot, joihin nimiensä mukaisesti tallennetaan työkaluja, layout-komponentteja ja robotteja. Parts-kansioon tallennetaan työstettävät kappaleet.



Kuva 9. Library Root -hakemisto.

Kansioiden nimillä ei ole väliä eikä periaatteessa Library Root -kansioon tarvitse luoda muita kansioita, mutta looginen kansiorakenne todennäköisesti helpottaa käyttäjää 3D-mallien löytämisessä, varsinkin jos malleja on paljon. Tärkeintä on, että kaikki mallit löytyvät saman kansion alta ja tämän kansion polku on ohjelmistossa määritetty Library Root -poluksi.

Ohjelmiston käynnistyttyä määritetään Library Root -polku valitsemalla **File → Options**, jolloin avautuu kuvan 10 mukainen ikkuna. **Library**-välilehdelle kohtaan **Library Root** lisätään polku, jossa Library Root -kansio sijaitsee **→ OK.**

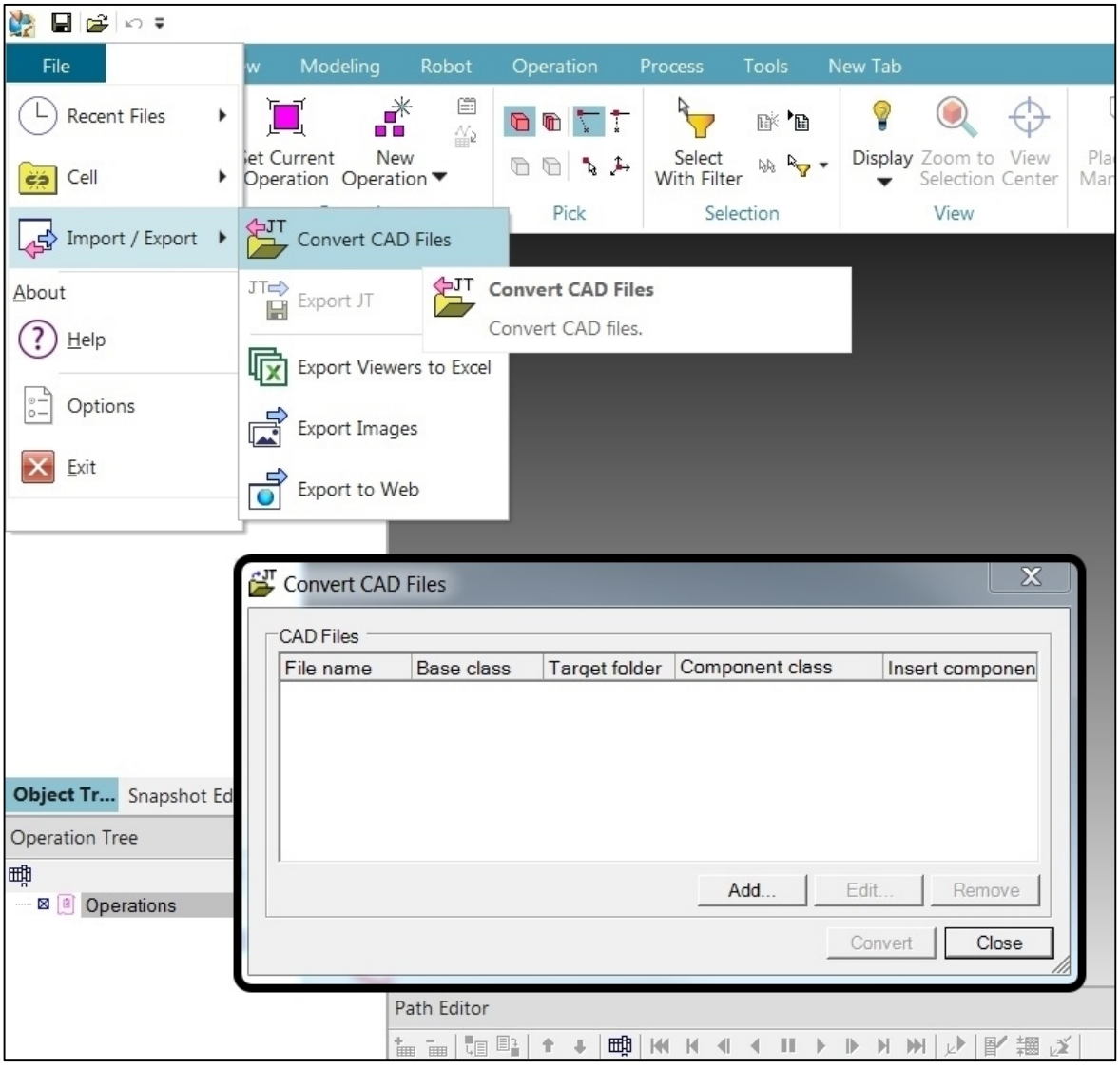


Kuva 10. Library Root -polun määritys.

3. Komponenttien muuttaminen COJT-muotoon ja tuominen ohjelmaan

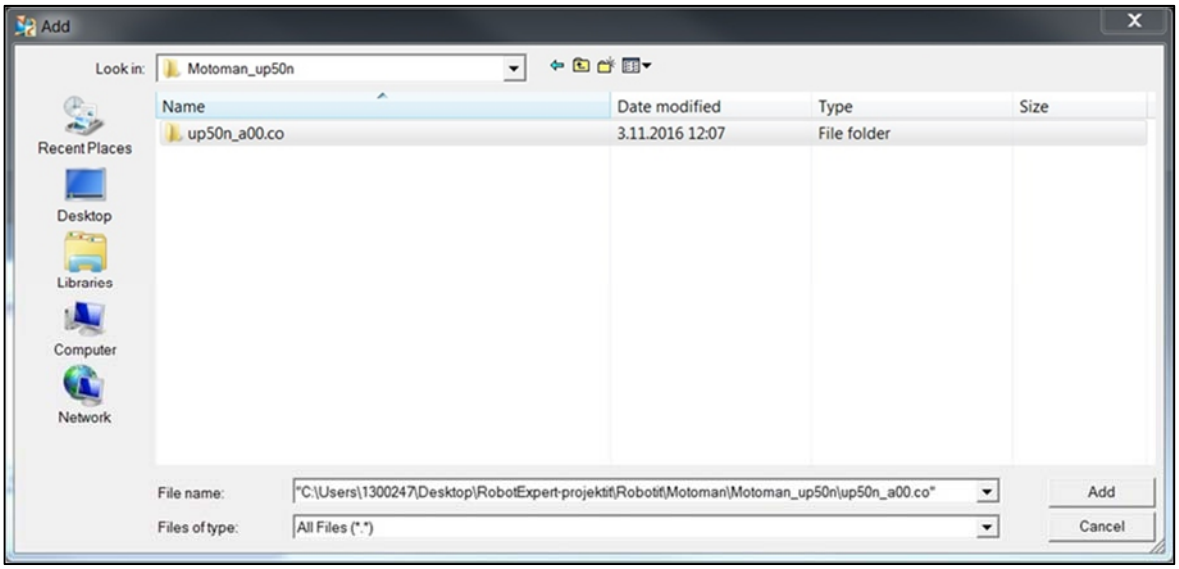
3.1 Robotti

Valitaan **File → Import / Export → Convert CAD Files** jolloin avautuu kuvan 11 mukainen ikkuna.



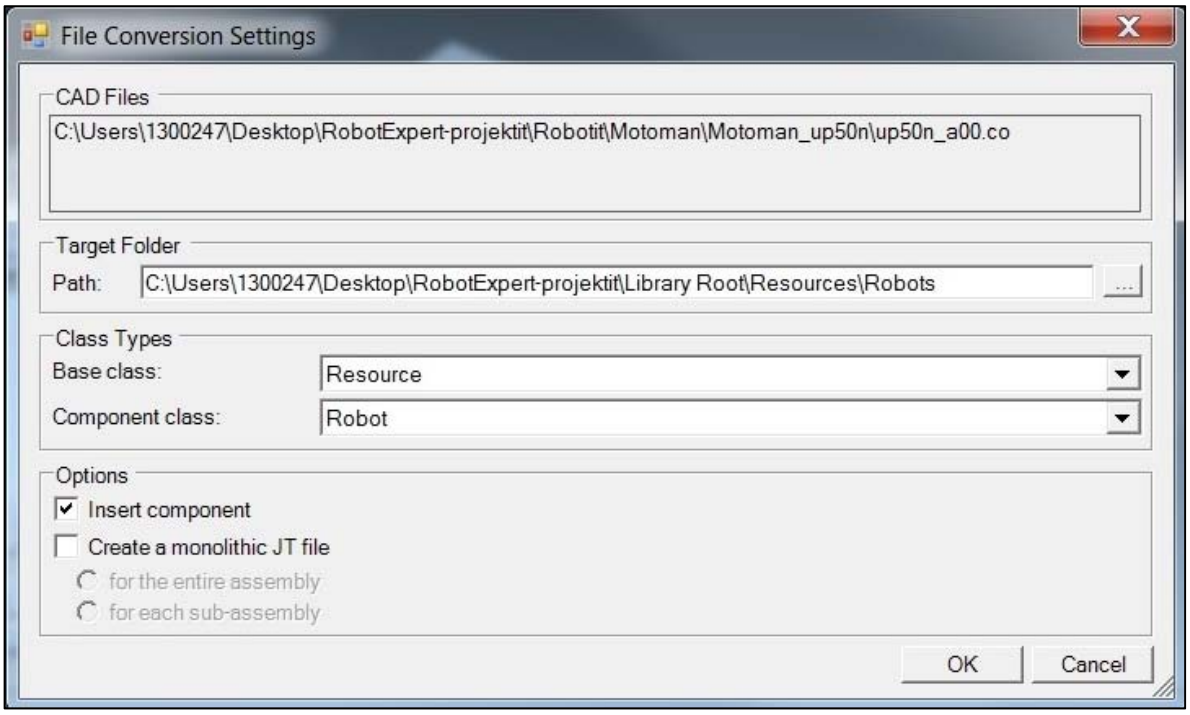
Kuva 11. Convert CAD Files -ikkuna.

Valitaan **Add…**, jolloin avautuu kuvan 12 mukainen ikkuna ja etsitään tietokoneelta robotti, joka halutaan muuttaa → **Add**.



Kuva 12. Robotin hakeminen tiedostoista.

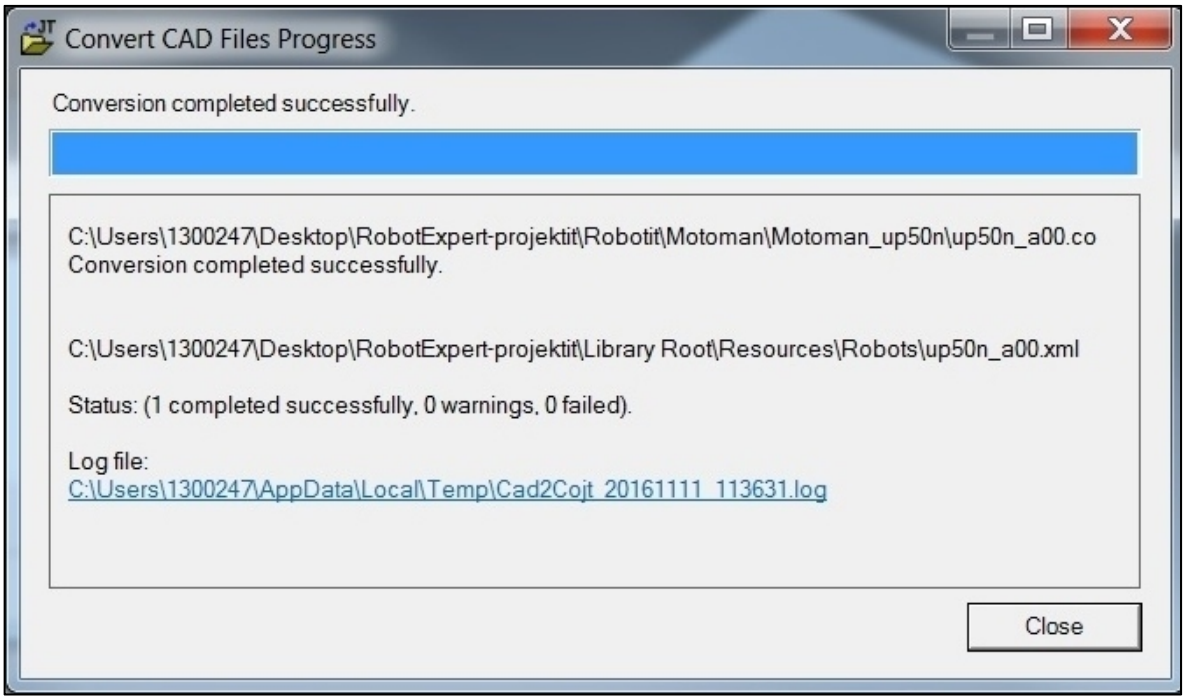
Tämän jälkeen avautuu kuvan 13 mukainen ikkuna, johon määritetään kohdekansio (**Target Folder**), johon muutettu tiedosto tallennetaan. Tämän kansion tulee olla Library Root -kansion alla.



Kuva 13. File Conversion Settings -ikkuna.

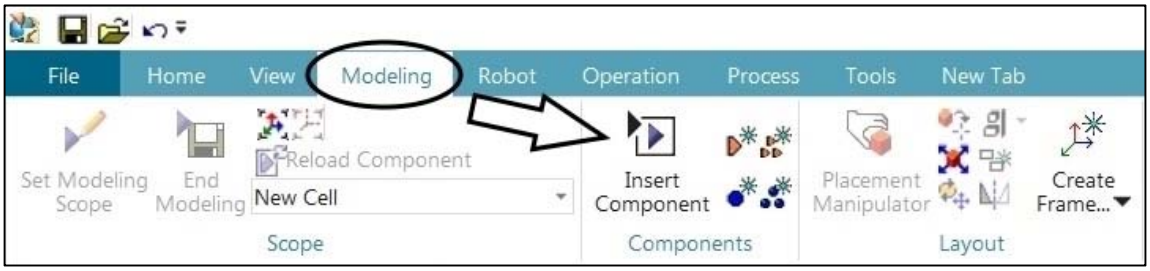
Class Types -kohtaan määritetään **Base class → Resource** ja **Component class → Robot**. **Options**-kohtaan valitaan **Insert component**, jos robotti halutaan heti lisätä ohjelmaan **→ OK** ja **Convert CAD Files** -ikkunasta **→ Convert**.

Kun tiedosto on muutettu kuten kuvassa 14, kannattaa tarkistaa, että **Status**-kohdassa on **0 warnings** ja **0 failed**. Jos muuttaminen ei onnistu, **Log file** kertoo epäonnistumisen syyn.



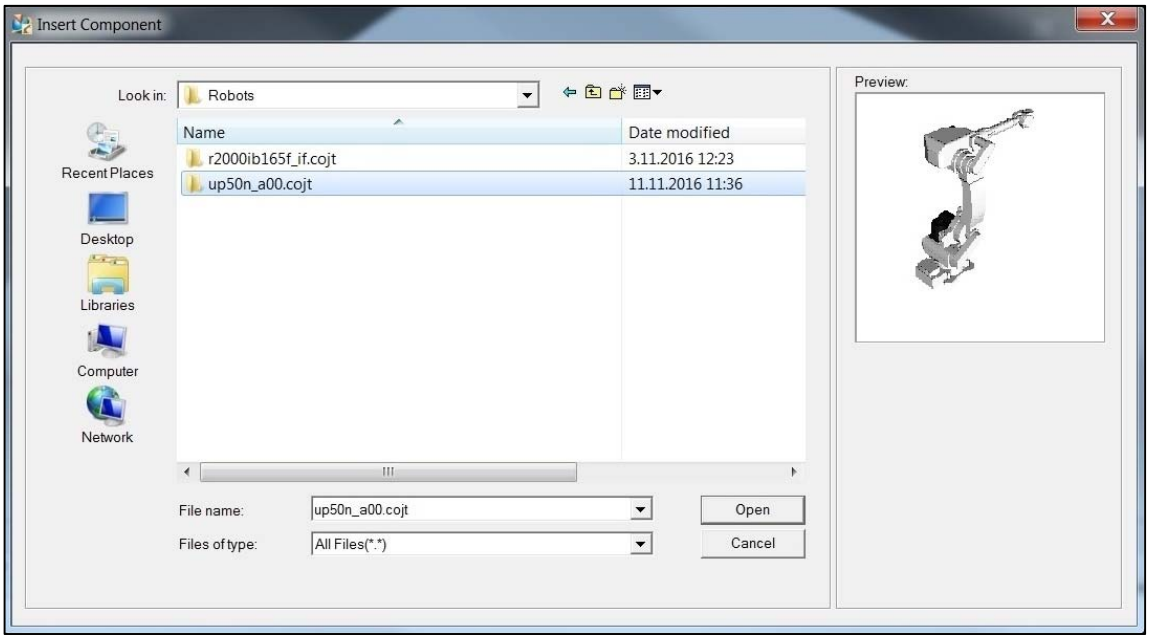
Kuva 14. Tiedostomuutoksen eteneminen.

Jos muuttaminen on onnistunut, robotin pitäisi ilmestyä **Graphic Viewer**iin. Jos robottia ei näy, vaikka **Status** on ollut kunnossa, voi tämä johtua siitä, että **Insert component** -kohta ei ole ollut valittuna **File Conversion Settings** -ikkunassa. Tällöin robotti on kyllä muutettu oikeaan tiedostomuotoon ja tallennettu kohdekansioon, mutta sitä ei ole tuotu ohjelmaan. Robotin voi tuoda myöhemmin ohjelmaan valitsemalla kuvan 15 mukaisesti **Modeling**-välilehdeltä **Insert Component**.



Kuva 15. Robotin tuominen ohjelmaan Insert Component -toiminnon avulla.

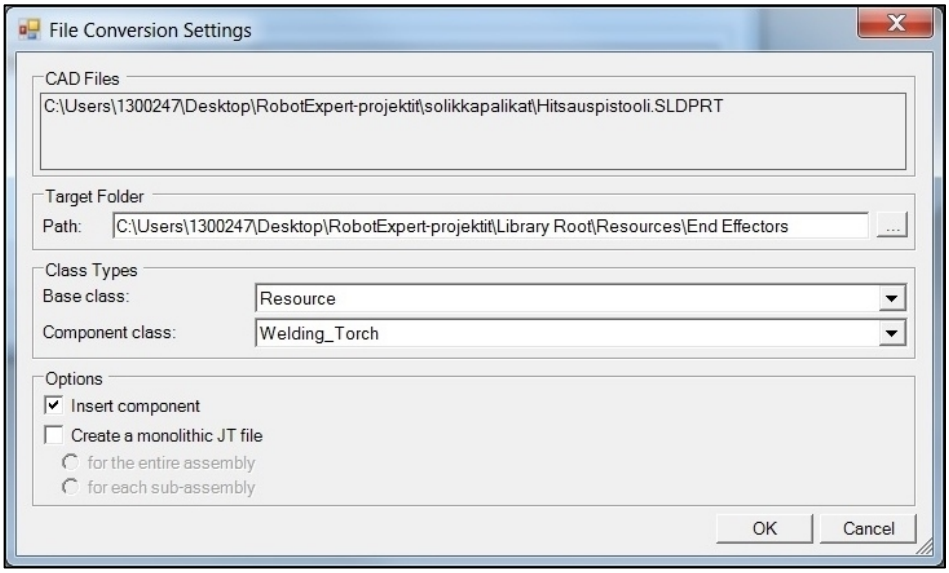
Ohjelmaan avautuu kuvan 16 mukainen **Insert Component** -ikkuna, jonka avulla muutettu tiedosto voidaan lisätä kohdekansiosta.



Kuva 16. Robotin lisääminen kohdekansiosta.

3.2 Muut komponentit

Muut komponentit (työkalut, pyörityspöydät jne.) muutetaan oikeaan muotoon ja tuodaan ohjelmaan samoin kuin robotit. **Class Types** -kohtaan valitaan **Base Class** sen mukaan, onko tuotava kappale Resource vai Part ja **Component class** -valikosta sopiva vaihtoehto esimerkiksi Welding_Torch kuten kuvassa 17.

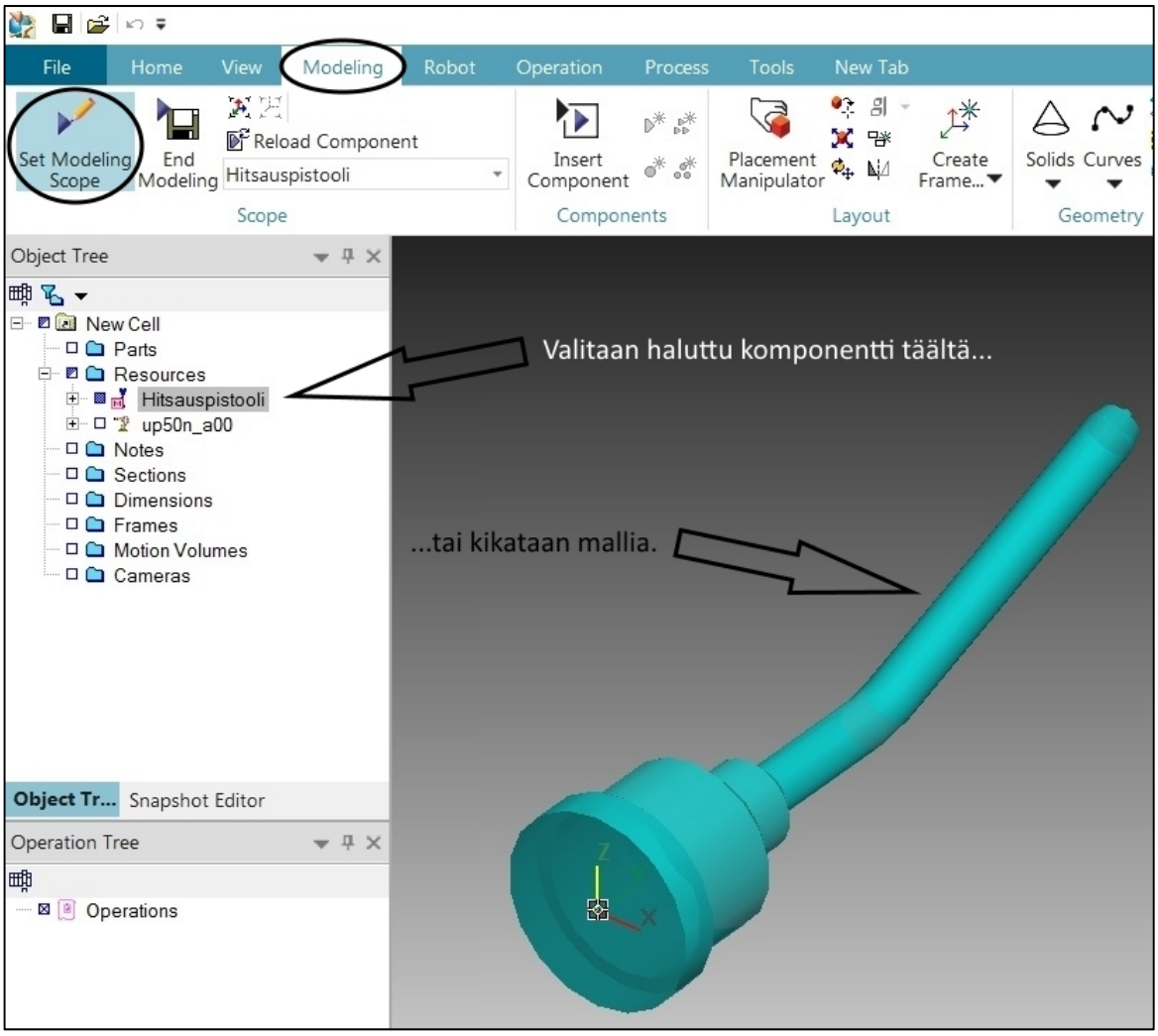


Kuva 17. Muiden komponenttien muuttaminen COJT-muotoon.

Jos komponentti on jo COJT-muodossa, sen voi tuoda ohjelmaan **Insert Component** -toiminnolla kuten kuvissa 15 ja 16.

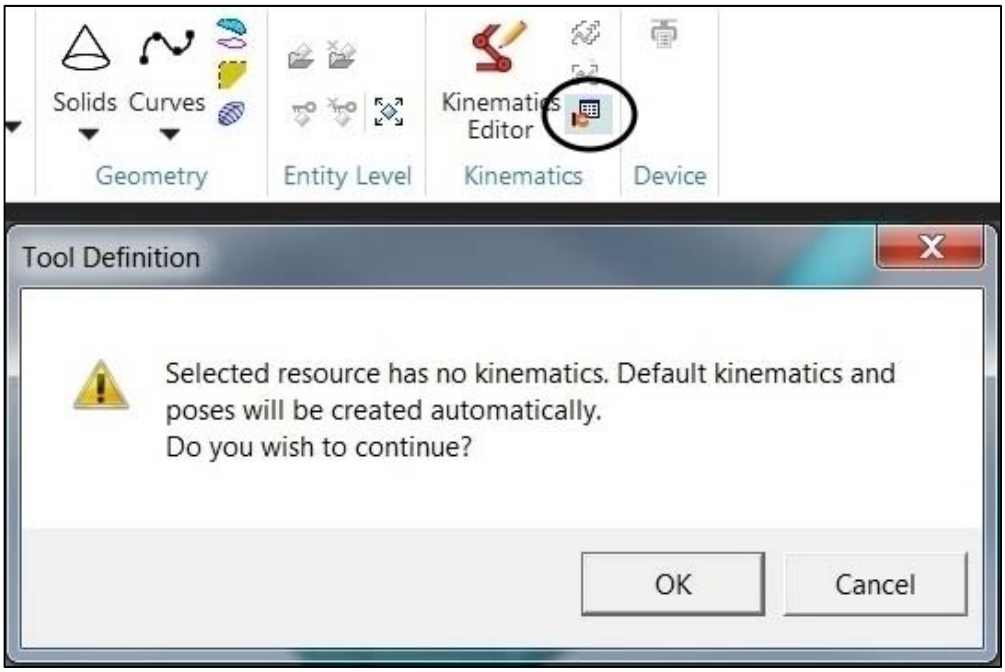
4. Työkalun luominen ja kiinnittäminen robottiin

Kun komponentti on tuotu oikeassa muodossa ohjelmaan edellisten kohtien mukaan, siitä voidaan tehdä työkalu. Kun komponenttien kinematiikkaa halutaan muokata tai niistä halutaan tehdä työkaluja, valitaan kyseinen komponentti joko objektipuusta tai klikataan mallia ja mennään **Modeling**-välilehdelle, josta valitaan **Set Modeling Scope**. Tällöin komponentin kohdalle objektipuuhun ilmestyy pieni punainen M-kirjain ja muokkaaminen on mahdollista. Komponentin ottaminen muokattavaksi on esitetty kuvassa 18.



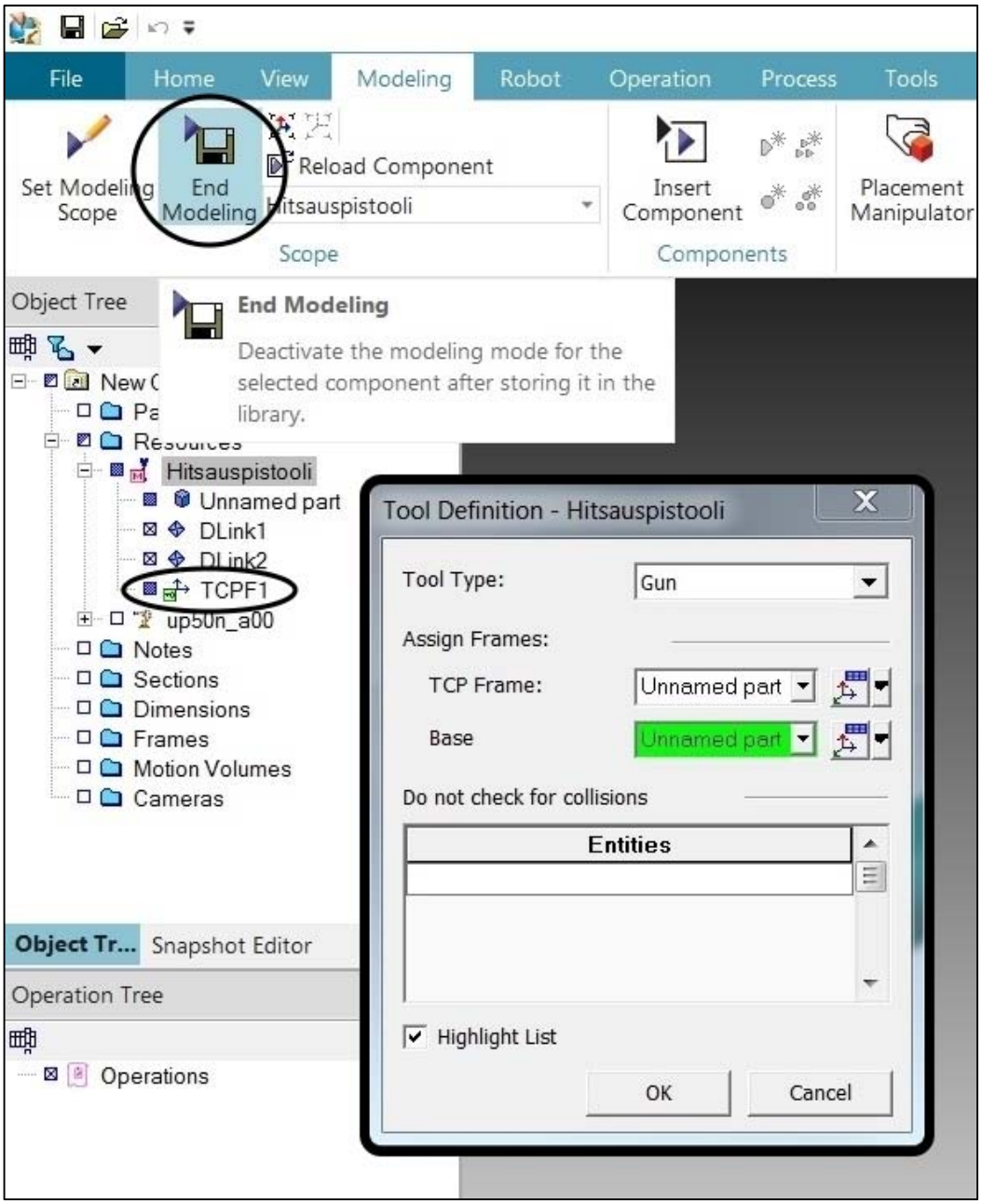
Kuva 18. Komponentin ottaminen muokattavaksi.

Valitaan **Modeling**-välilehdeltä **Kinematics**-kohdasta **Tool Definition**. Jos valitulla komponentilla ei ole kinematiikkaa, avautuu kuvan 19 mukainen ikkuna, jolloin komponentille luodaan oletuskinematiikka → **OK**.



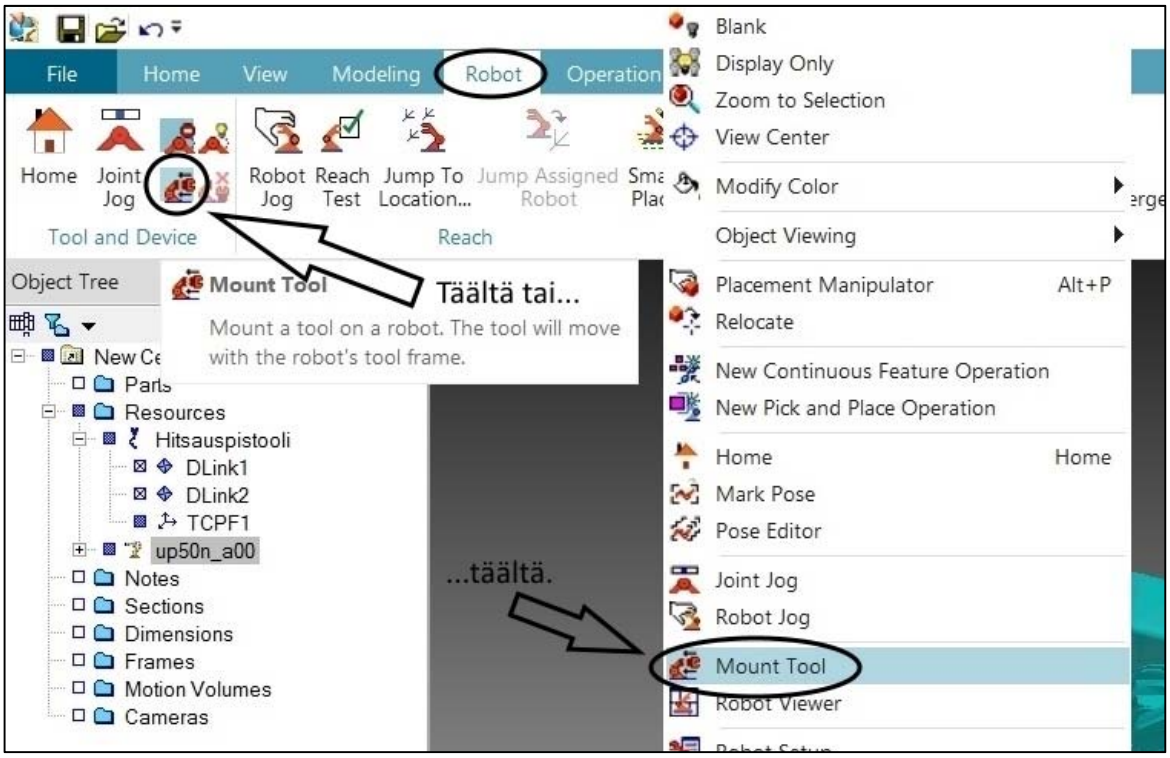
Kuva 19. Komponentin määrittäminen työkaluksi.

Tämän jälkeen avautuu kuvan 20 mukainen **Tool Definition** -ikkuna, johon valitaan **Tool Type → Gun** (tai mikä työkalu onkin kyseessä). **TCP Frame** -kohtaan määritetään työkalupisteen paikka klikkaamalla haluttua kohtaa työkalussa. Työkalukoordinaatiston asentoa ja paikkaa voidaan myöhemmin muokata. **Base**-kohtaan valitaan, mistä työkalu kiinnitetään robottiin → **OK**. **Object Tree** -valikkoon kyseisen työkalun alle ilmestyy TCPF1 työkalukoordinaatiston merkiksi. Kun muokkaus on valmis, valitaan **End Modeling**, jolloin muutokset tallentuvat.



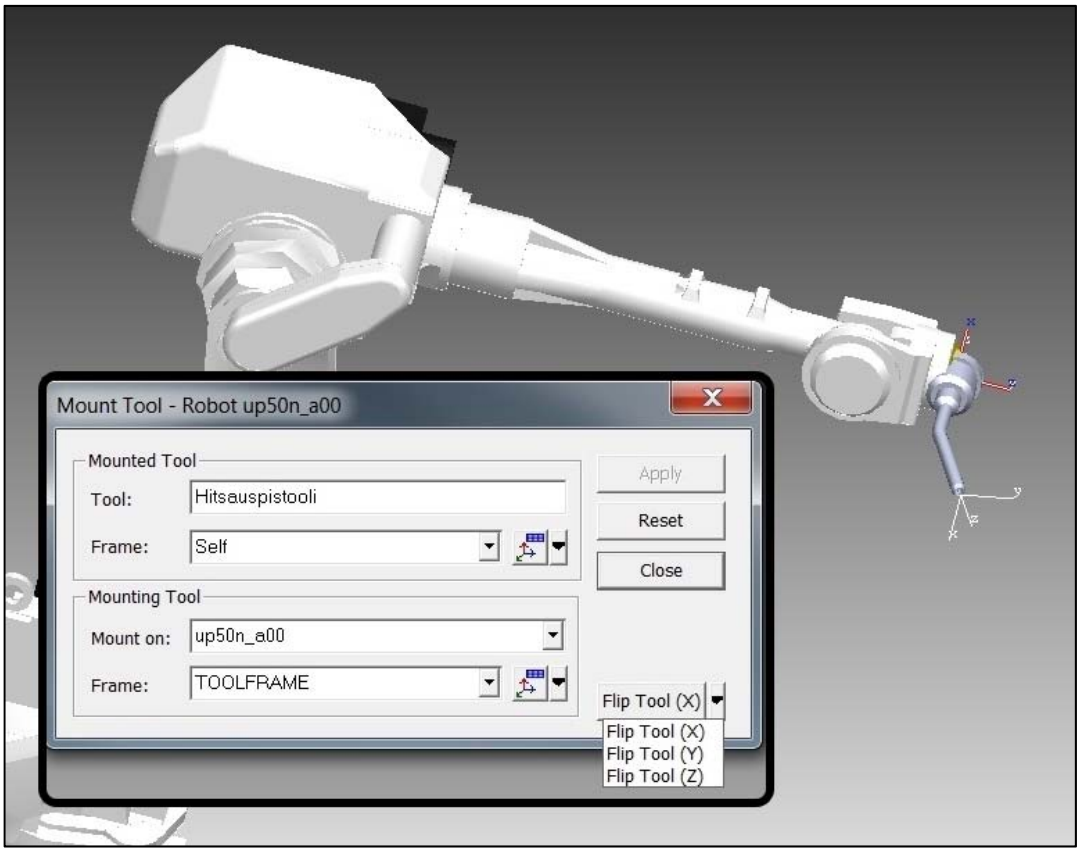
Kuva 20. Työkalutyypin ja työkalupisteen määritys.

Työkalu kiinnitetään robottiin joko klikkaamalla robottia hiiren oikealla ja valitsemalla **Mount Tool** tai valitsemalla sama valikko **Robot**-välilehdeltä **Tool and Device** -kohdasta, kuten kuvassa 21 on esitetty.



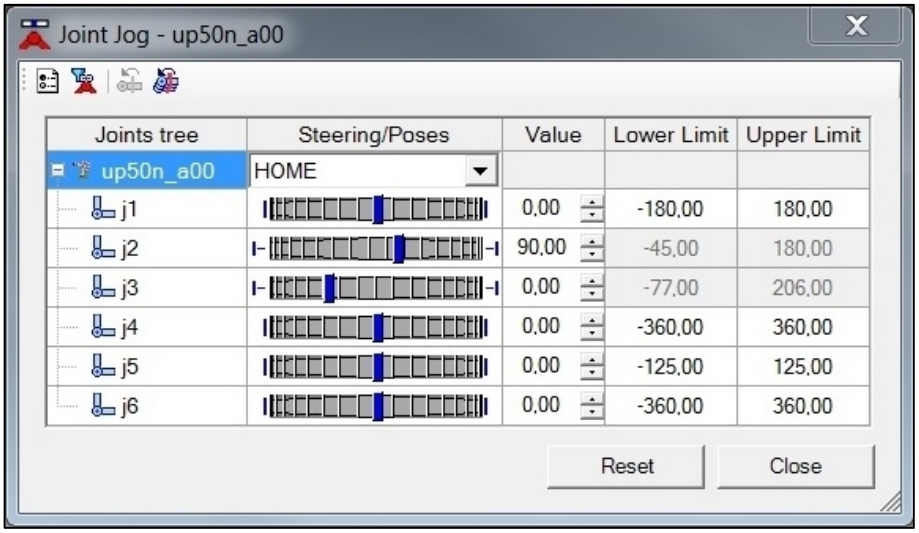
Kuva 21. Mount Tool -valikko.

Mount Tool -ikkunaan määritetään **Tool**-kohtaan työkalu, joka halutaan kiinnittää ja **Frame**-kohtaan, mistä työkalu halutaan kiinnittää. **Mounting on** -kohdassa näkyy robotti, johon työkalu kiinnitetään ja **Frame**-kohdassa, mihin kohtaan robottia työkalu kiinnitetään → **Apply**. Jos työkalu kiinnittyy robottiin väärin päin, sitä voidaan kiertää **Flip Tool** -kohdan avulla. Muutokset hyväksytään painamalla rastia. Työkalun kiinnittäminen on esitetty kuvassa 22.



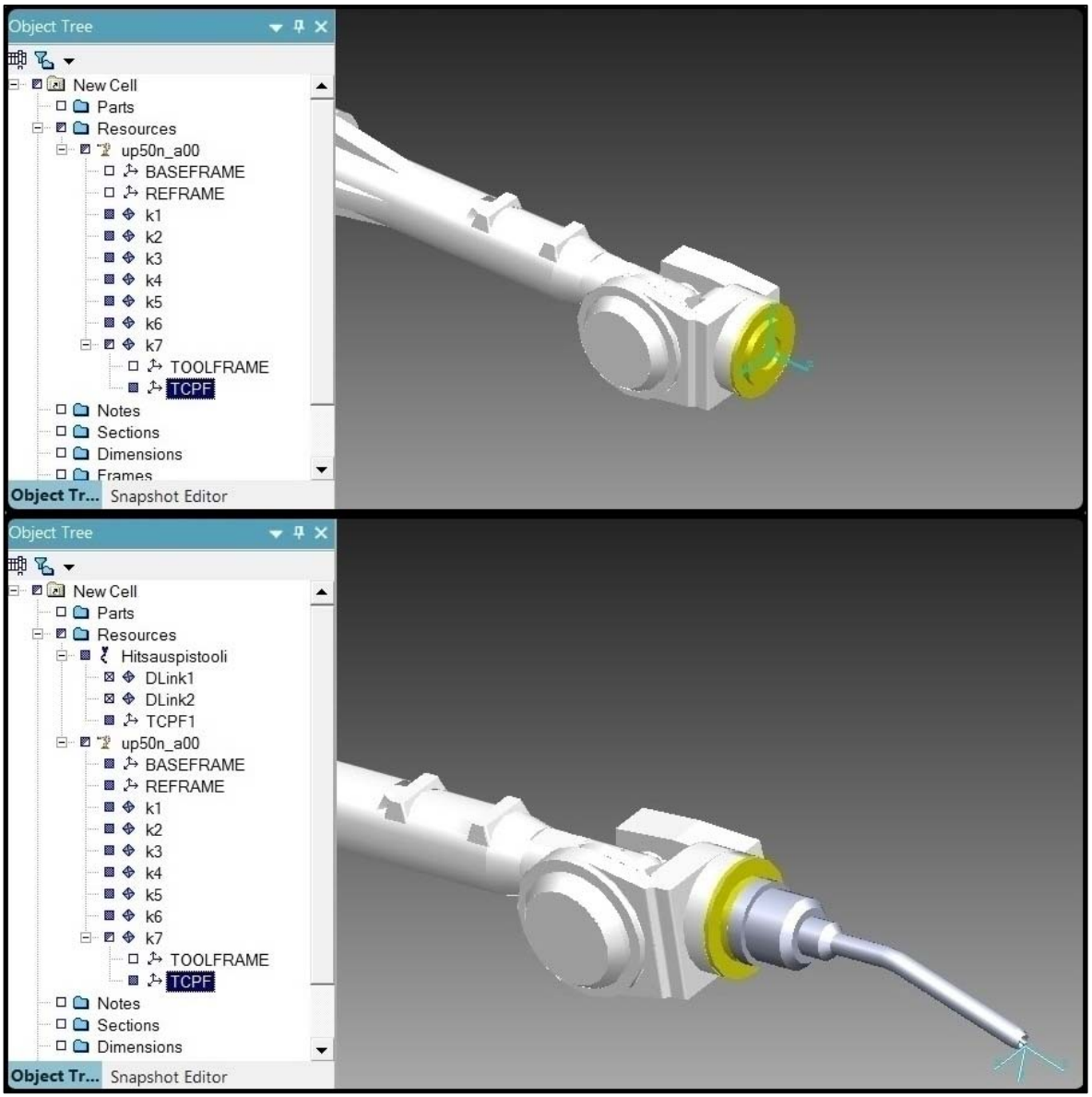
Kuva 22. Työkalun kiinnittäminen robottiin.

Työkalun kiinnittyminen robottiin voidaan varmistaa klikkaamalla robottia hiiren oikealla ja valitsemalla **Joint Jog** tai **Robot Jog**, jolloin robottia pystytään liikuttamaan. Jos työkalu on kiinnittynyt oikein, se kulkee robotin mukana. Valitsemalla **Joint Jog** avautuu kuvan 23 mukainen ikkuna, josta liukuvalitsimilla, nuolilla tai syöttämällä arvoja, pystytään muuttamaan nivelten asentoja.



Kuva 23. Robotin nivelten liikuttaminen.

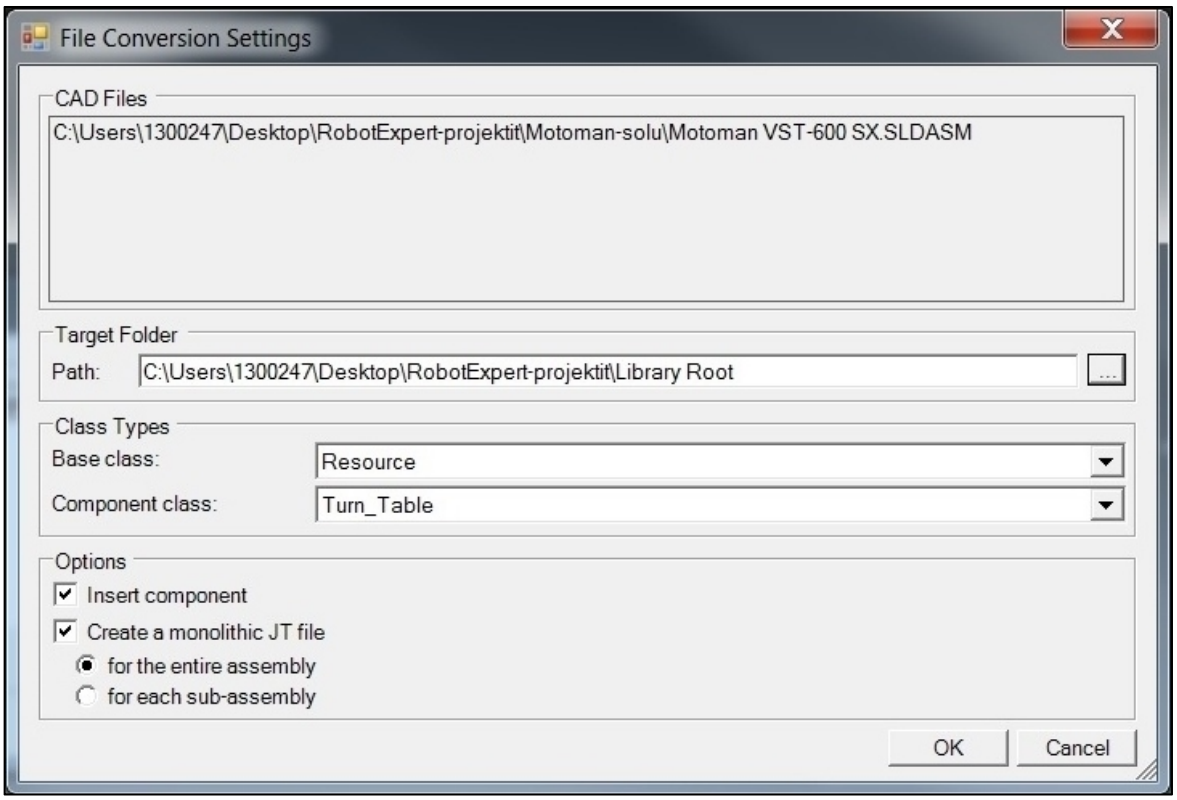
Robotin työkalupisteen pitäisi muuttua automaattisesti samaksi kuin työkalulle määritetty työkalupiste. Tämän voi tarkistaa objektipuusta etsimällä robotin TCPF:n. Ennen työkalun kiinnittämistä TCPF oli kiinni robotin päässä, nyt sen pitäisi olla työkalun päässä kuten kuvassa 24.



Kuva 24. Työkalupisteen ja -koordinaatiston siirtyminen robotista työkaluun.

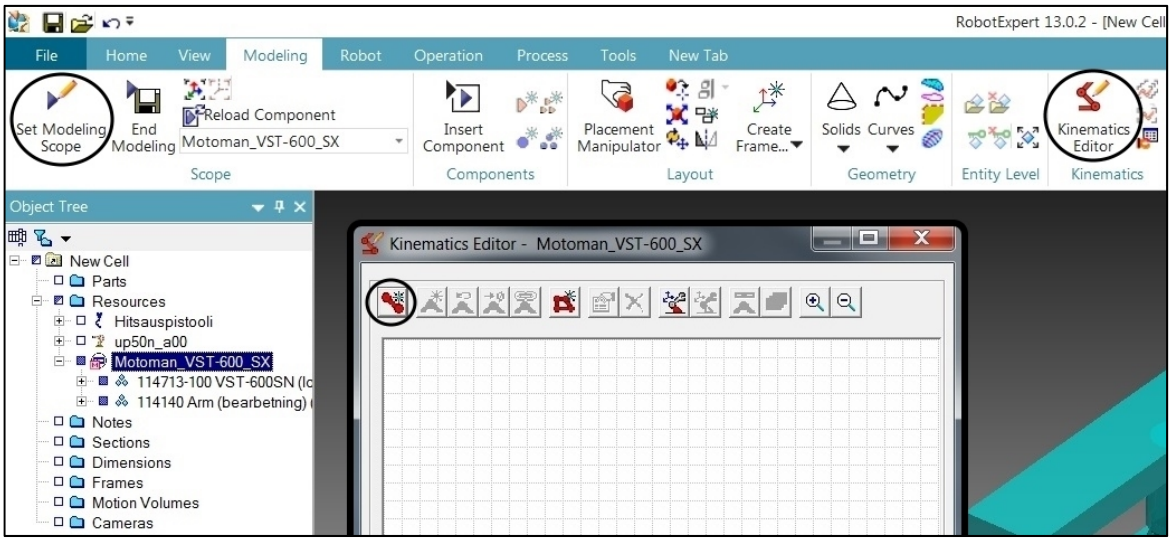
5. Kinematiikan luominen (esimerkkinä robotin pyöritysjalka)

Pyöritysjalan, -pöydän tai vastaavan malli tuodaan ohjelmaan kuten edellä, mutta **File Conversion Settings** -ikkunan **Options**-kohtaan valitaan **Create monolithic JT file for the entire assembly**, kuten kuvassa 25. Jos tätä ei valita, kokoonpanon eri osien välille ei pystytä luomaan kinematiikkaa.



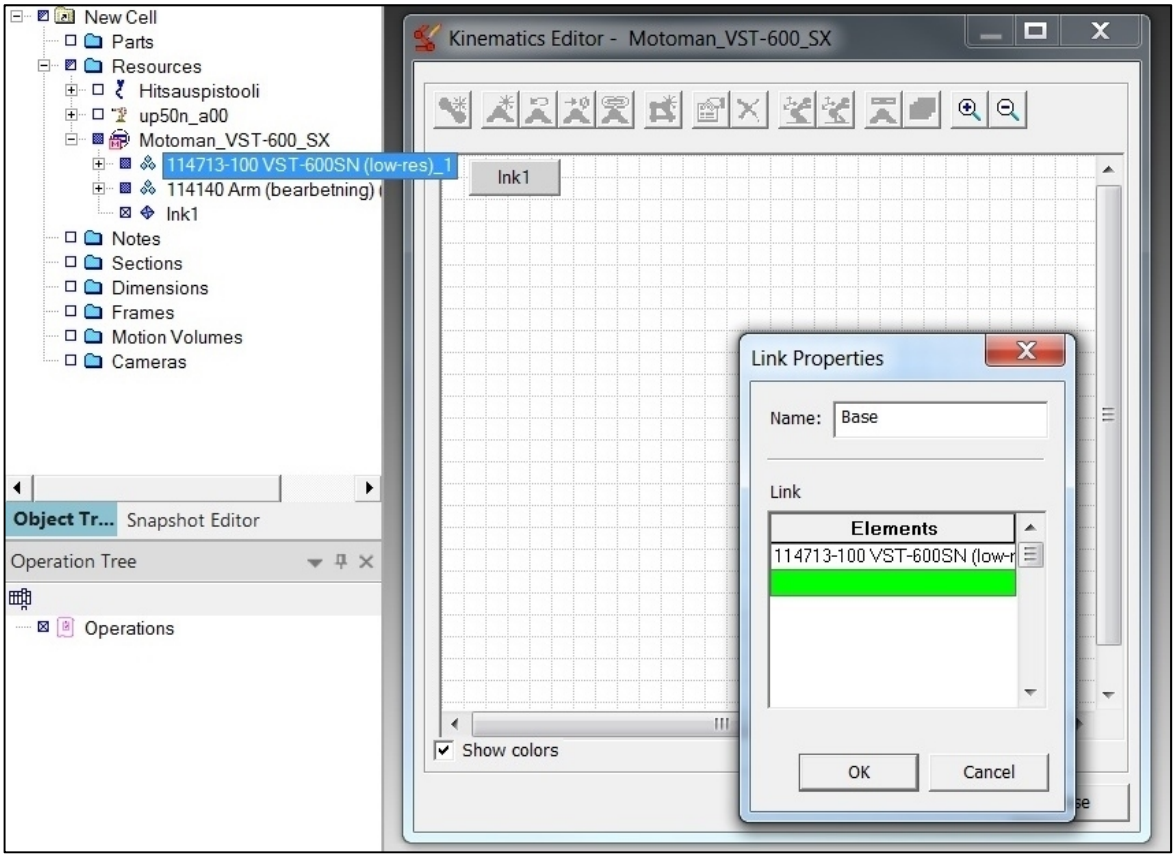
Kuva 25. Pyöritysjalan tuominen ohjelmaan.

Jotta pyöritysjalan osat saataisiin liikkumaan oikein toisiinsa nähden, otetaan jalka muokattavaksi (**Set Modeling Scope**) ja valitaan **Kinematics Editor**, jolloin avautuu kuvan 26 mukainen ikkuna, josta valitaan **Create Link**, jolloin avautuu kuvan 27 mukainen **Link Properties** -ikkuna.



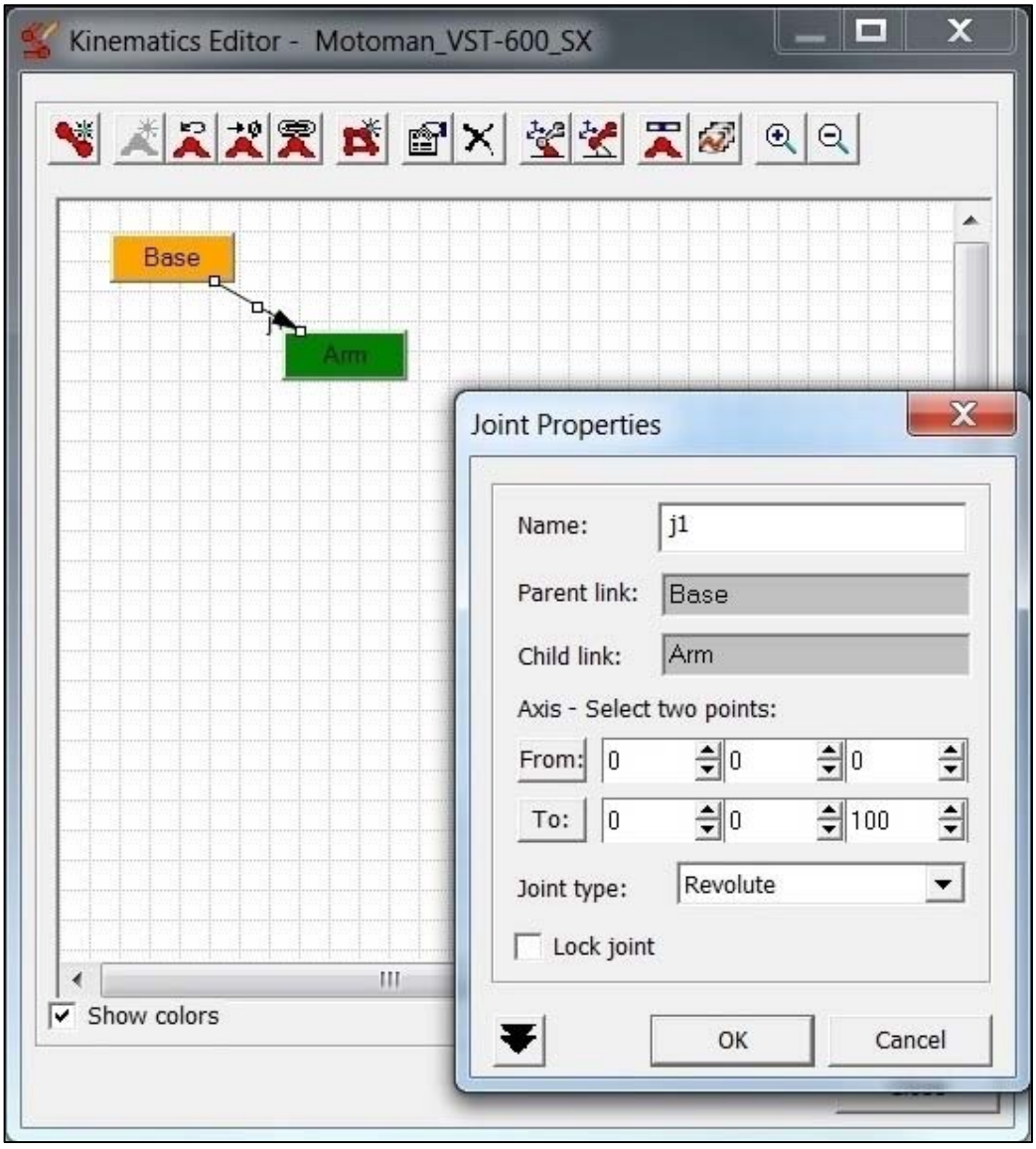
Kuva 26. Kinematics Editor.

Link Properties -ikkunassa annetaan ensimmäiselle linkille nimi ja valitaan **Elements**-kohtaan haluttu osa pyöritysjalasta, joko objektipuusta tai klikkaamalla kyseistä osaa mallista → **OK**.



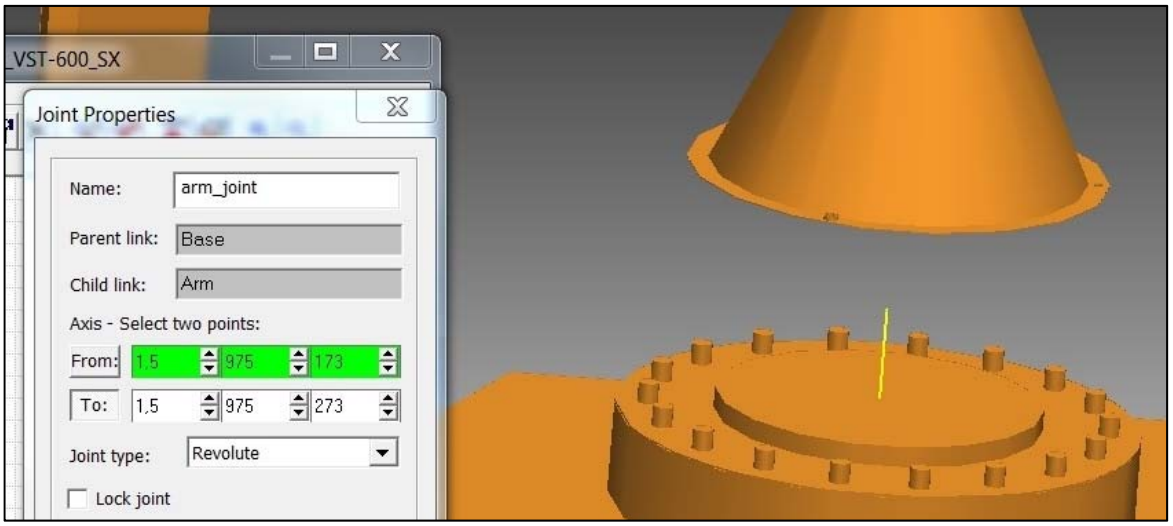
Kuva 27. Link Properties -ikkuna.

Lisätään toinen linkki samalla tavalla. Näiden linkkien väliin luodaan nivel vetämällä toisesta laatikosta nuoli toiseen, jolloin avautuu kuvan 28 mukainen **Joint Properties** -ikkuna, johon määritetään nivelen nimi sekä **Parent** ja **Child** linkit. Parent link on se nivelen osa, joka pysyy paikallaan ja Child link on nivelen liikkuva osa. **Joint type** -kohtaan määritetään niveltyypiksi joko pyörivä (Revolute) tai lineaarinen (Prismatic).



Kuva 28. Nivelen ominaisuuksien määrittäminen.

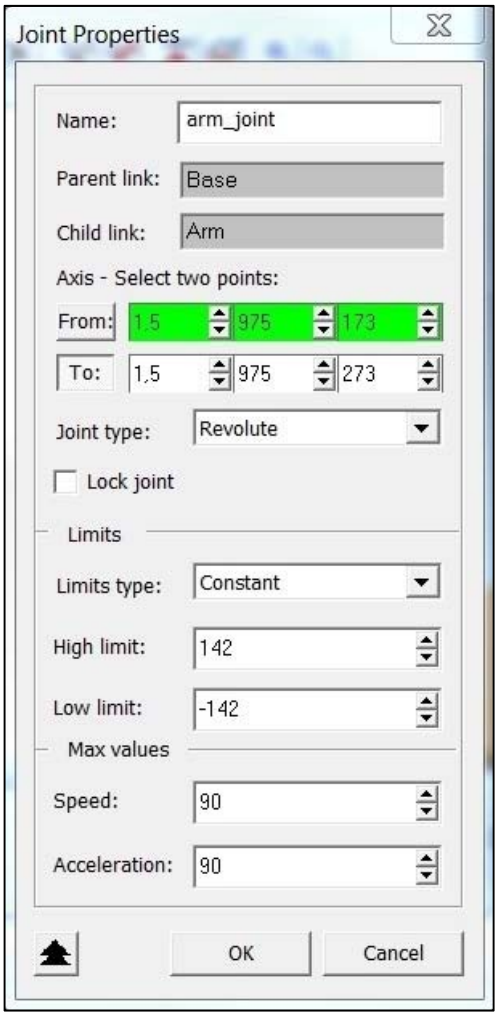
Axis-kohtaan määritetään, minkä akselin ympäri tai mitä akselia pitkin liike tapahtuu. Akselista määritetään alku- ja loppupisteen koordinaatit. Helpoiten tämä käy klikkaamalla ensin **From**-kohta aktiiviseksi ja sitten klikkaamalla haluttua alkupistettä mallista. **To**-kohtaan kirjoitetaan samat arvot, mutta lisätään esimerkiksi 100 mm halutun akselin suuntaan. Määritetty akseli näkyy kuvassa 29 keltaisena viivana.



Kuva 29. Akselin määrittäminen nivelelle.

Nivelen liikkeelle voidaan asettaa rajoituksia. Tässä esimerkiksi jalka pyörii ±142 astetta. Lisäksi voidaan tarvittaessa määrittää maksiminopeus ja -kiihtyvyys. Laajennetaan **Joint Properties** -ikkunaa nuolesta kuvan 30 mukaisesti. Valitaan **Limits type** -kohtaan **Constant** ja annetaan liikkeelle ylä- ja alaraja asteina tai millimetreinä riippuen niveltyypistä.

Kun halutut määritykset on tehty → **OK** → **Close** ja **Modeling**-välilehdeltä **End Modeling**.

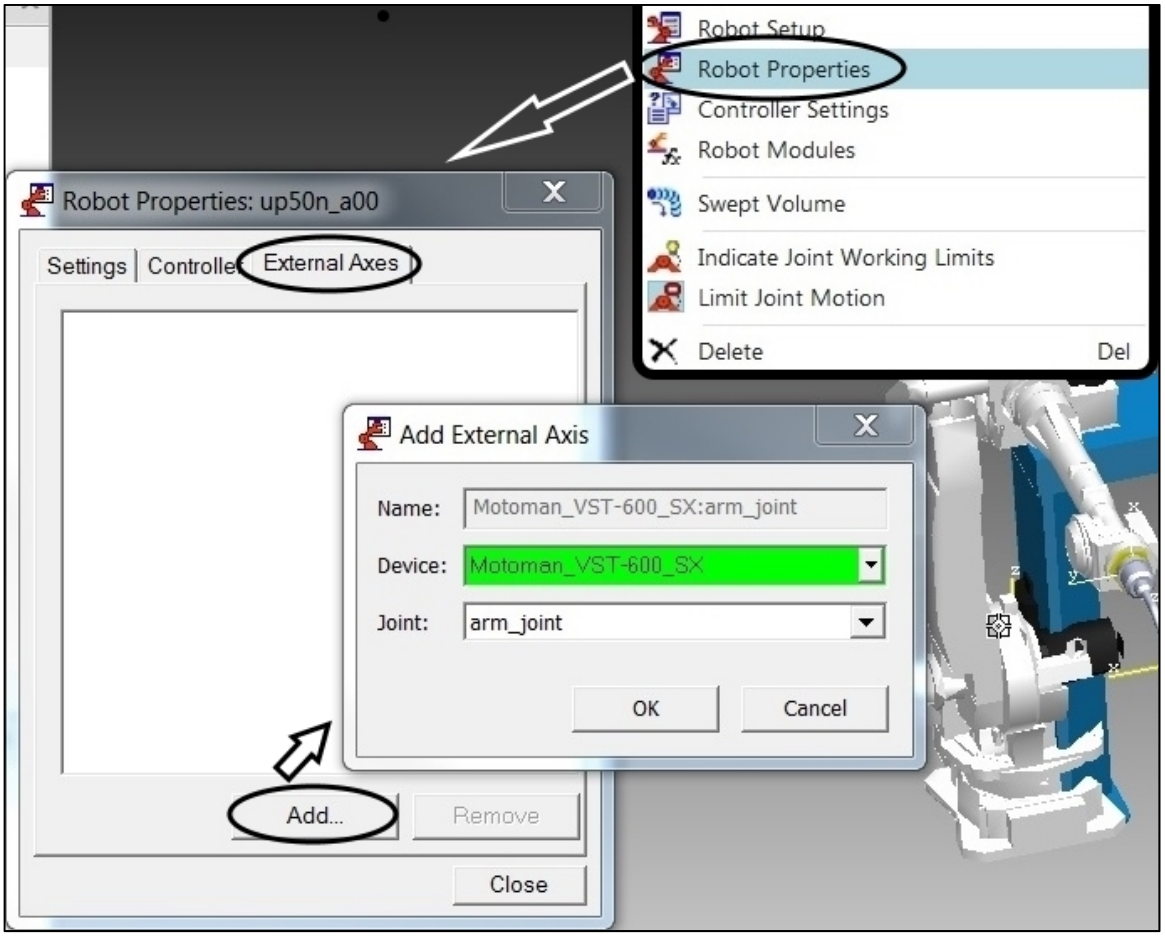


Kuva 30. Nivelen liikkeiden rajoittaminen.

Nivelen toimintaa voidaan testata klikkaamalla pyöritysjalkaa hiiren oikealla ja valitsemalla **Joint Jog**.

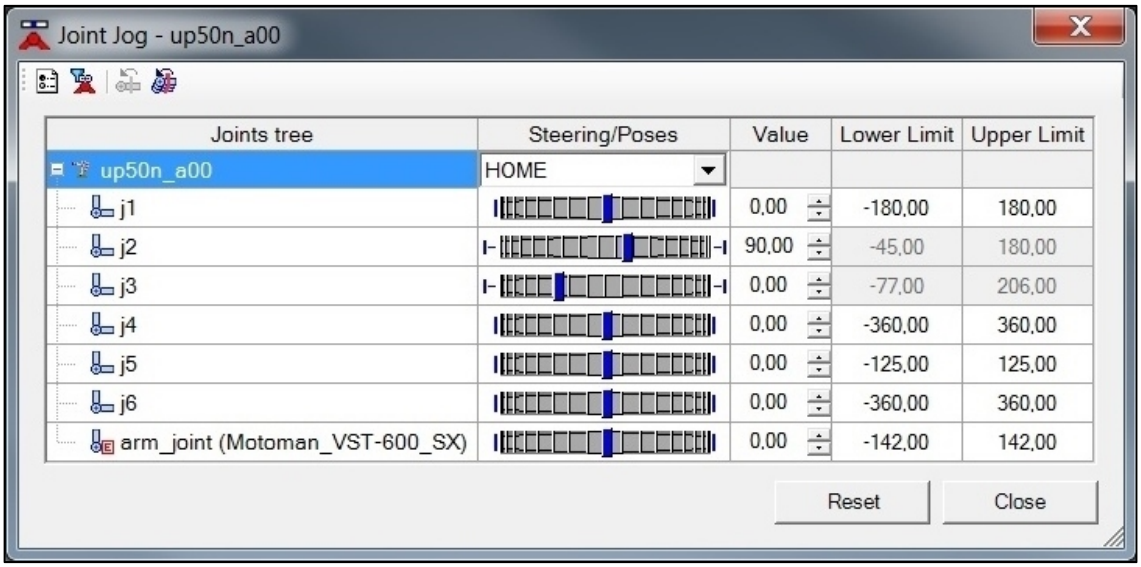
6. Ulkoisen akselin luominen robotille (ja robotin kiinnittäminen pyöritysjalkaan)

Jotta pyöritysjalan tai -pöydän liikkeet saadaan synkronoitua robotin liikkeiden kanssa, täytyy robotille määrittää ulkoinen akseli. Klikataan robottia hiiren oikealla ja valitaan **Robot Properties** ja **External Axes** -välilehti → **Add**. Valitaan **Device**-kohtaan pyöritysjalka ja **Joint**-kohtaan oikea nivel (tässä tapauksessa ainoa nivel, joten valikoituu automaattisesti) → **OK** → **Close**. Ulkoisen akselin luominen on esitetty kuvassa 31.



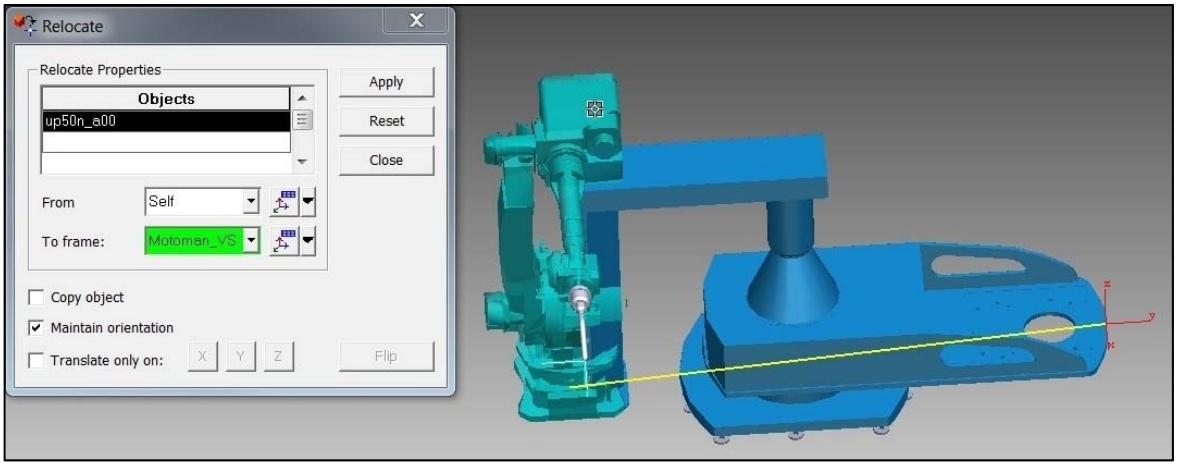
Kuva 31. Ulkoisen akselin luominen robotille.

Nyt pyöritysjalan akseli on osa robotin akselistoa. Tämän voi tarkistaa klikkaamalla robottia hiiren oikealla ja valitsemalla **Joint jog**, jolloin pyöritysjalan nivel on ilmestynyt **Joints tree** -listaan kuvan 32 mukaisesti.



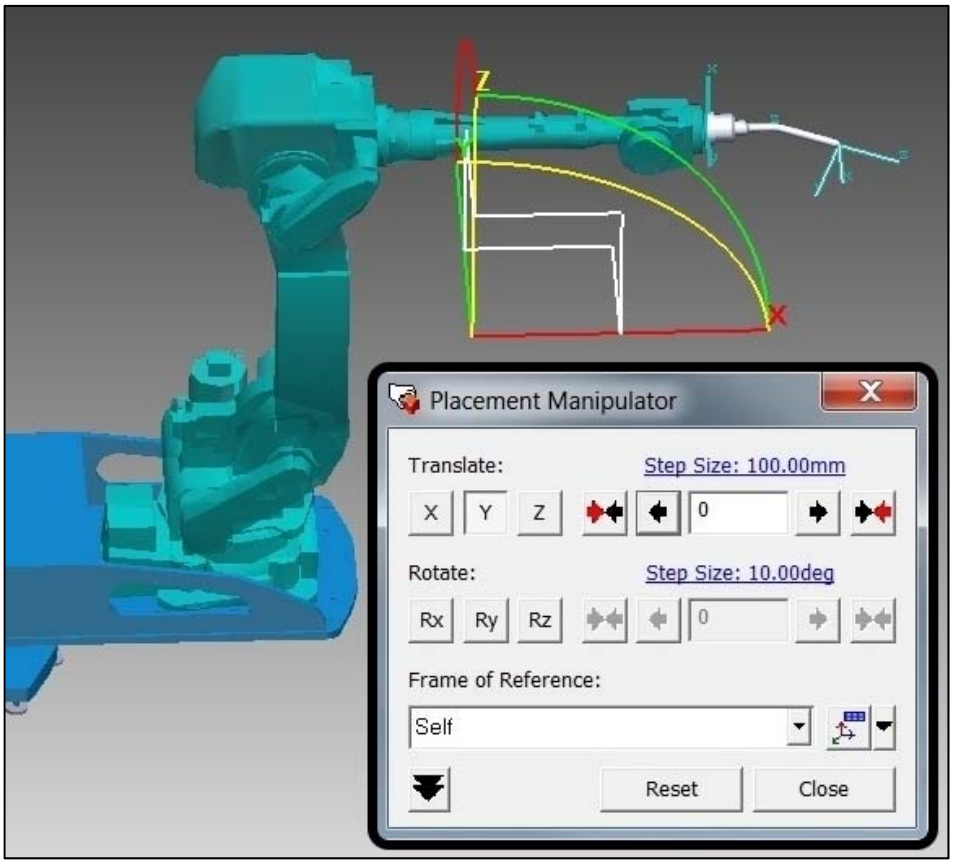
Kuva 32. Pyöritysjalan nivel osana robottia.

Robotti voidaan kiinnittää pyöritysjalkaan siirtämällä se ensin oikeaan kohtaan **Relocate**-toiminnolla. Klikataan robottia hiiren oikealla ja valitaan **Relocate**, jolloin avautuu kuvan 33 mukainen ikkuna. **From**-kohtaan valitaan Self ja **To frame** -kohtaan klikataan haluttua kohtaa pyöritysjalasta → **Apply**.



Kuva 33. Robotin siirtäminen Relocate-toiminnolla.

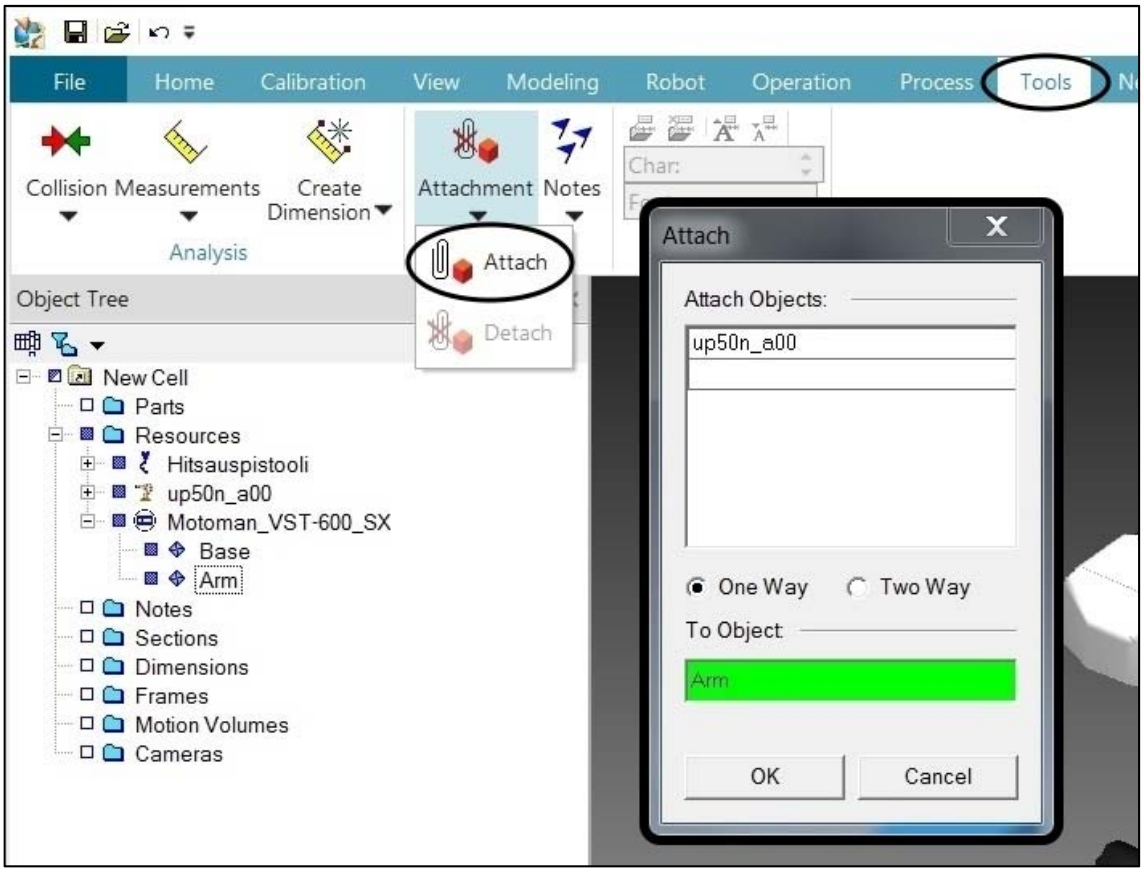
Jos robotti ei siirtynyt aivan haluttuun kohtaan, sen paikkaa voidaan muuttaa ja sitä voidaan kääntää helposti klikkaamalla robottia hiiren oikealla ja valitsemalla **Placement Manipulator** kuvan 34 mukaisesti. Kun luodaan uutta robottisolua, **Relocate** ja **Placement Manipulator** -toiminnoilla voidaan solun laitteet ja muut komponentit sijoittaa layout-mittojen mukaisesti paikoilleen.



Kuva 34. Robotin asettaminen oikeaan kohtaan pyöritysjalassa.

Jotta robotti pyörisi pyöritysjalan mukana, se täytyy vielä kiinnittää pyöritysjalkaan. Avataan **Tools**-välilehdeltä **Layout**-kohdasta **Attachment**-pudotusvalikko ja valitaan **Attach**, jolloin avautuu kuvan 35 mukainen ikkuna. **Attach Objects** -kohtaan valitaan robotti, kiinnitystavaksi valitaan **One Way** ja **To Object** -kohtaan valitaan operaatiopuusta se osa pyöritysjalasta, johon robotti halutaan kiinnittää → **OK**.

One Way ja Two Way -kiinnitystapojen ero on siinä, että One Way -tavassa kiinnitettyä komponenttia, tässä tapauksessa robottia, liikutettaessa pyöritysjalka ei liiku, mutta pyöritysjalkaa liikutettaessa robotti liikkuu. Two Way -tavassa toista komponenttia liikutettaessa myös toinen liikkuu, riippumatta siitä onko se kiinnitetty komponentti (Attach Objects) vai se komponentti, johon on kiinnitetty (To Object).



Kuva 35. Robotin kiinnittäminen pyöritysjalkaan.

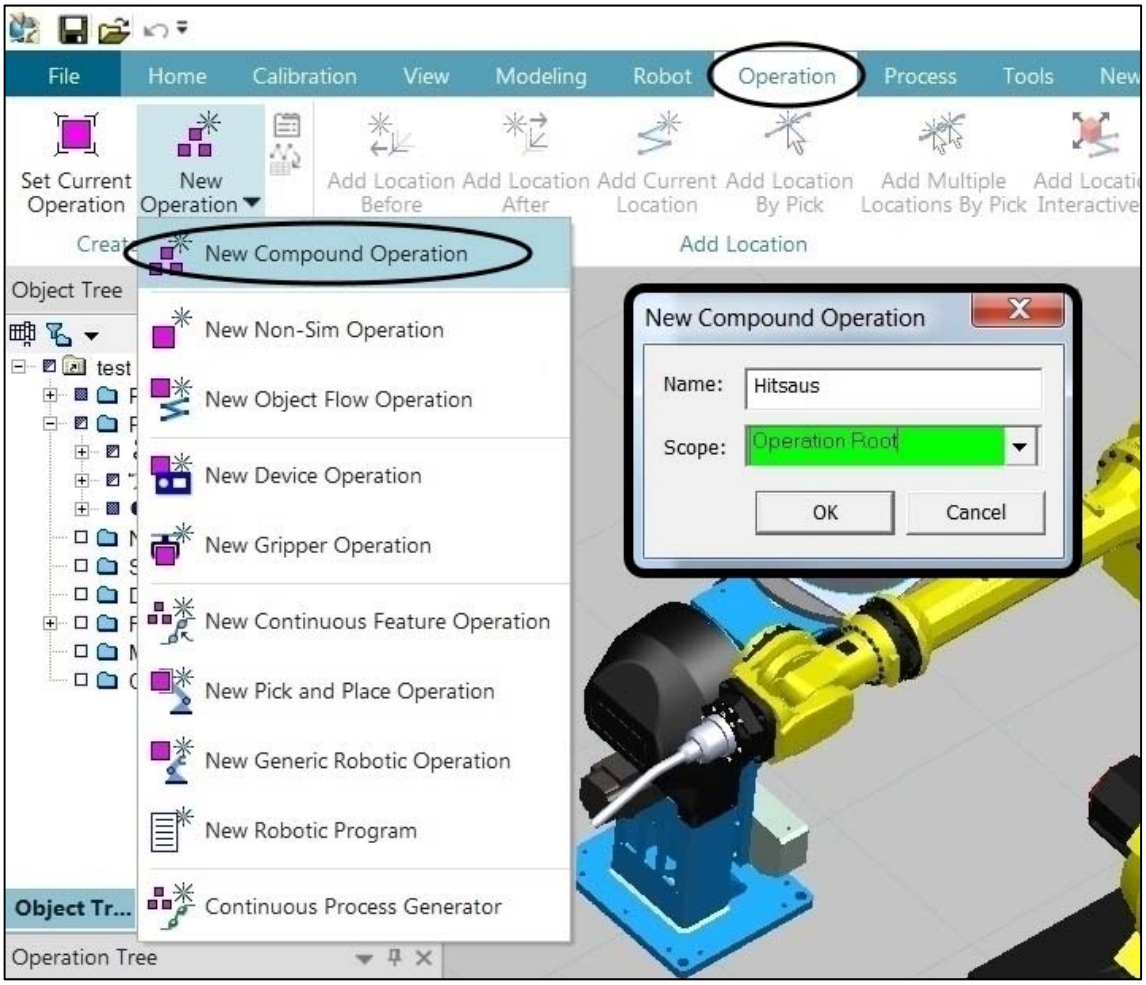
Robotin kiinnittyminen voidaan tarkistaa klikkaamalla pyöritysjalkaa hiiren oikealla ja valitsemalla **Joint Jog**, jolloin pyöritysjalkaa liikuttamalla robotin pitäisi liikkua mukana.

7. Liikeradan luominen ja simulointi

7.1 Hitsausesimerkki

Kuvien esimerkissä soluun on robotin ja hitsauspistoolin lisäksi tuotu pyörityspöytä ja hitsattava kappale. Hitsauspistoolista on tehty työkalu ja se on kiinnitetty robottiin kuten kohdassa neljä. Pyörityspöydälle on luotu kaksi niveltä, jotka on määritetty robotin ulkoisiksi akseleiksi kuten kohdissa viisi ja kuusi. Lisäksi hitsattava kappale on kiinnitetty pyörityspöytään, kuten robotti kiinnitettiin pyöritysjalkaan kohdassa kuusi.

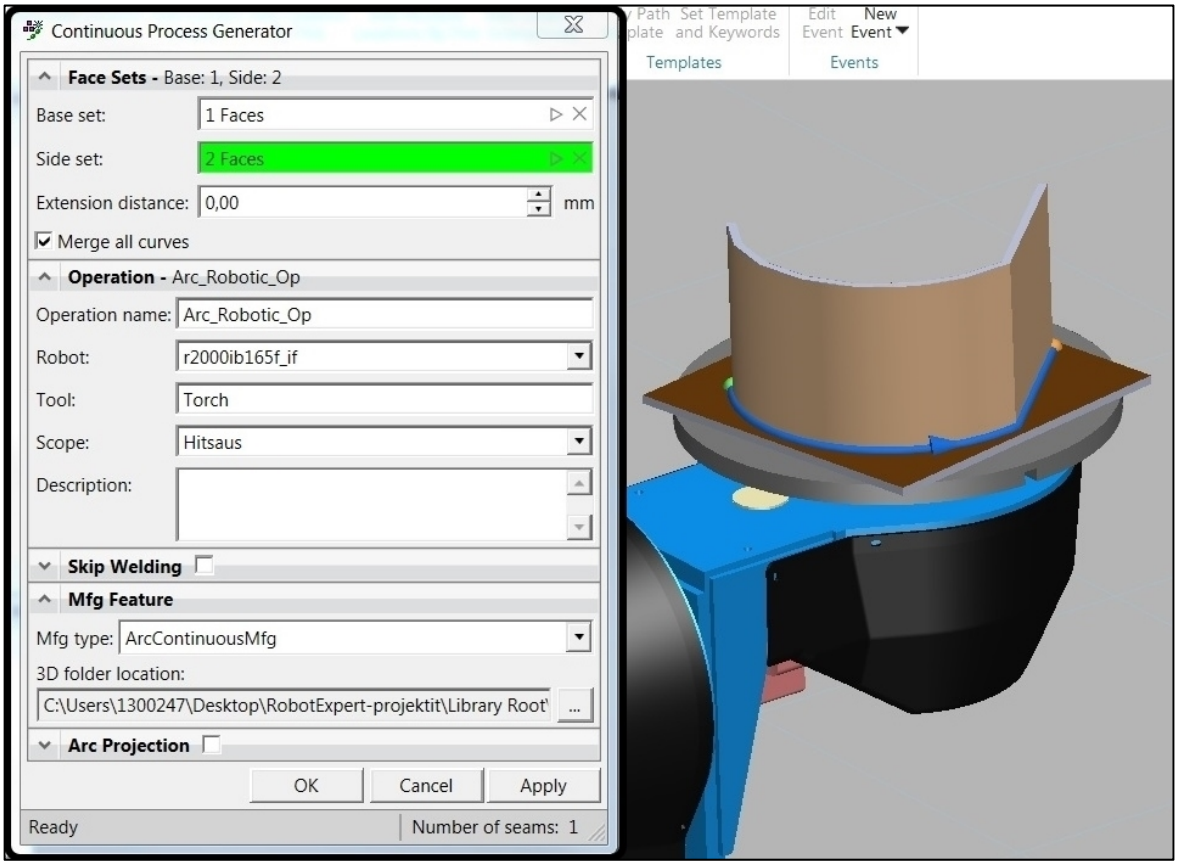
Operation-välilehdeltä **Create Operation** -kohdasta avataan **New Operation** -valikko, josta valitaan **New Compound Operation**, jolloin avautuu kuvan 36 mukainen ikkuna. Operaatiolle annetaan nimi ja **Scope**-kohtaan valitaan **Operation Root → OK**. Uusi operaatio näkyy nyt operaatiopuussa. Compound Operation on yhdistelmäoperaatio, joka muodostuu muista operaatioista ja voi sisältää myös muita yhdistelmäoperaatioita.



Kuva 36. Uuden yhdistelmäoperaation luominen.

Hitsaussaumat ts. liikeradat hitsauspistoolille luodaan klikkaamalla Hitsaus-operaatio aktiiviseksi ja valitsemalla **New Operation** -valikosta **Continuous Process Generator**,

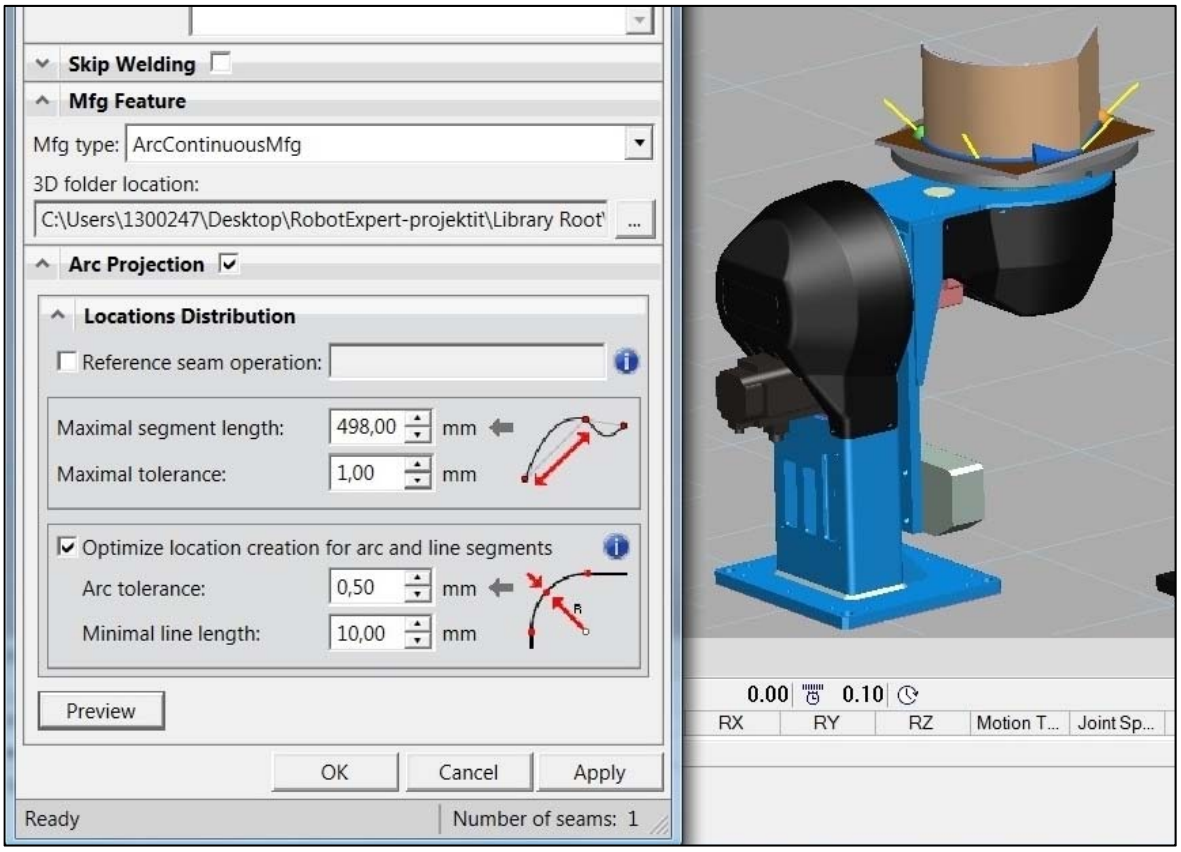
jolloin avautuu kuvan 37 mukainen ikkuna. Ohjelma luo hitsaussaumat hitsattavan kappaleen geometrian perusteella. **Base set** -kohtaan valitaan kappaleen pohja klikkaamalla kyseistä osaa mallista (kuvassa tummanruskealla) ja **Side set** -kohtaan hitsattavat sivut (kuvassa vaaleanruskealla). Sininen nuoli kuvastaa luotua liikerataa.



Kuva 37. Hitsaussauman luominen.

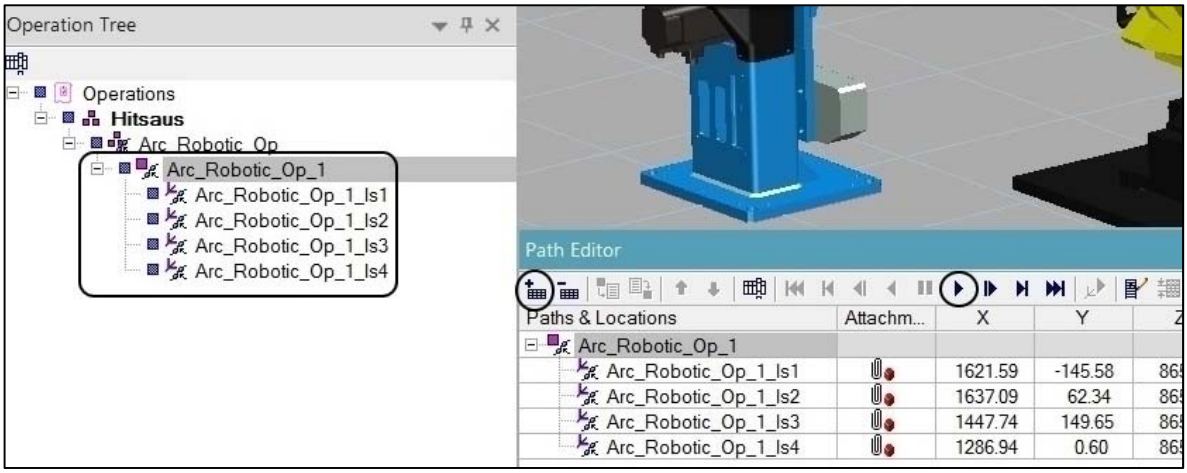
Operation-kohdassa voidaan operaatiolle antaa nimi ja kuvaus sekä valita robotti ja työkalu. Jos halutaan käyttää katkohitsausta, voidaan **Skip Welding** -kohdassa määrittää katkohitsauksen asetuksia, kuten osahitsien pituus ja lukumäärä. **Mfg Feature** -kohtaan määritetään tuotantomenetelmä (kaarihitsaus, liimaus, laserleikkaus, maalaus jne.) sekä mihin luodun operaation 3D-tiedot tallennetaan.

Valitaan **Arc Projection**, jolloin hitsaussauman ja kappaleen geometrian perusteella robotille määritetään liikepisteet hitsausta varten. Nuolesta saa auki kuvan 38 mukaisen valikon. **Locations Distribution** -kohtaan voi määrittää, miten pisteet jakautuvat pitkin saumaa tai käyttää jo luodun sauman parametreja (**Reference seam operation**). Valitaan **Preview**, jos halutaan nähdä pisteiden paikat hitsaussaumassa **→ OK**.



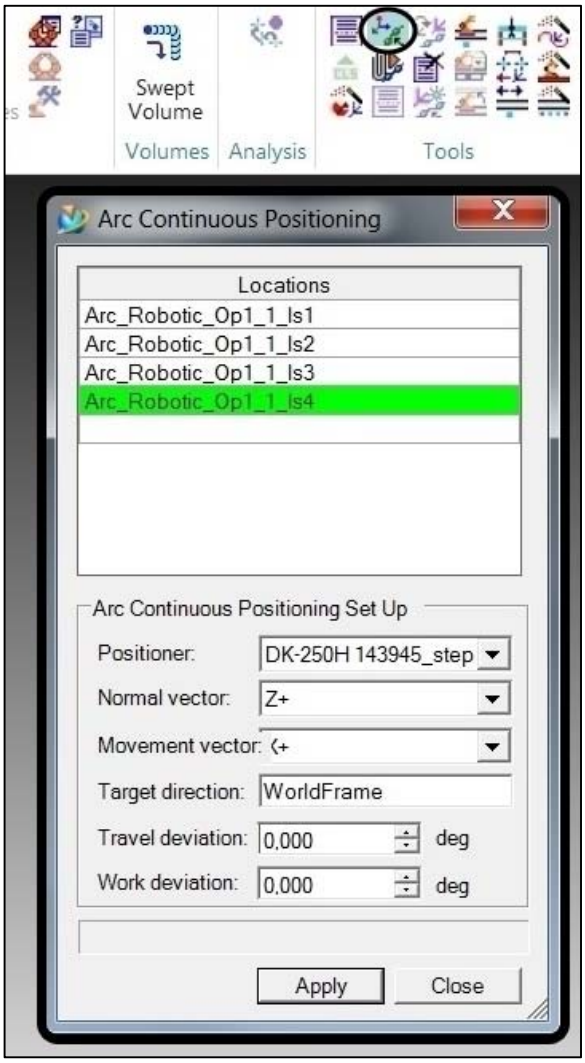
Kuva 38. Liikepisteiden luominen hitsauspistoolille.

Luodut liikepisteet näkyvät kuvan 39 mukaisesti operaatiopuussa. Lisätään pisteet Path Editoriin valitsemalla haluttu operaatio aktiiviseksi operaatiopuusta ja painamalla **Add Operations to Editor** . Simulointi voidaan nyt käynnistää painamalla **Play Simulation Forward** .



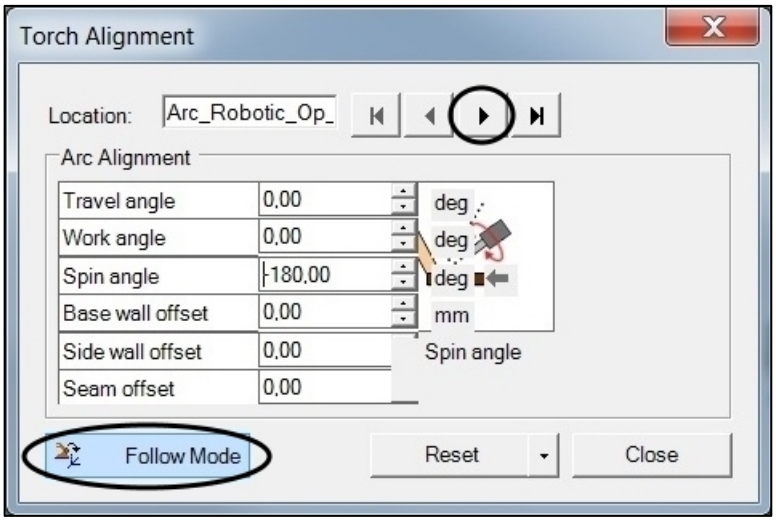
Kuva 39. Liikepisteiden lisääminen Path Editoriin ja simuloinnin käynnistys.

Tässä vaiheessa robotti ei välttämättä vielä liiku luotua rataa pitkin, vaan otetaan käyttöön pyörityspöytä, jotta robotti ulottuisi paremmin haluttuihin pisteisiin. Valitaan operaatio aktiiviseksi Path Editorista tai operaatiopuusta ja **Robot**-välilehdeltä **Tools**-kohdasta valitaan **Arc Continuous Positioning**, jolloin avautuu kuvan 40 mukainen ikkuna. **Locations**-kohdassa näkyy ne pisteet, joihin asemointi kohdistuu ja **Positioner**-kohtaan valitaan pyörityspöytä. Tässä esimerkissä muita arvoja ei muuteta **→ Apply → Close**.



Kuva 40. Arc Continuous Positioning -ikkuna.

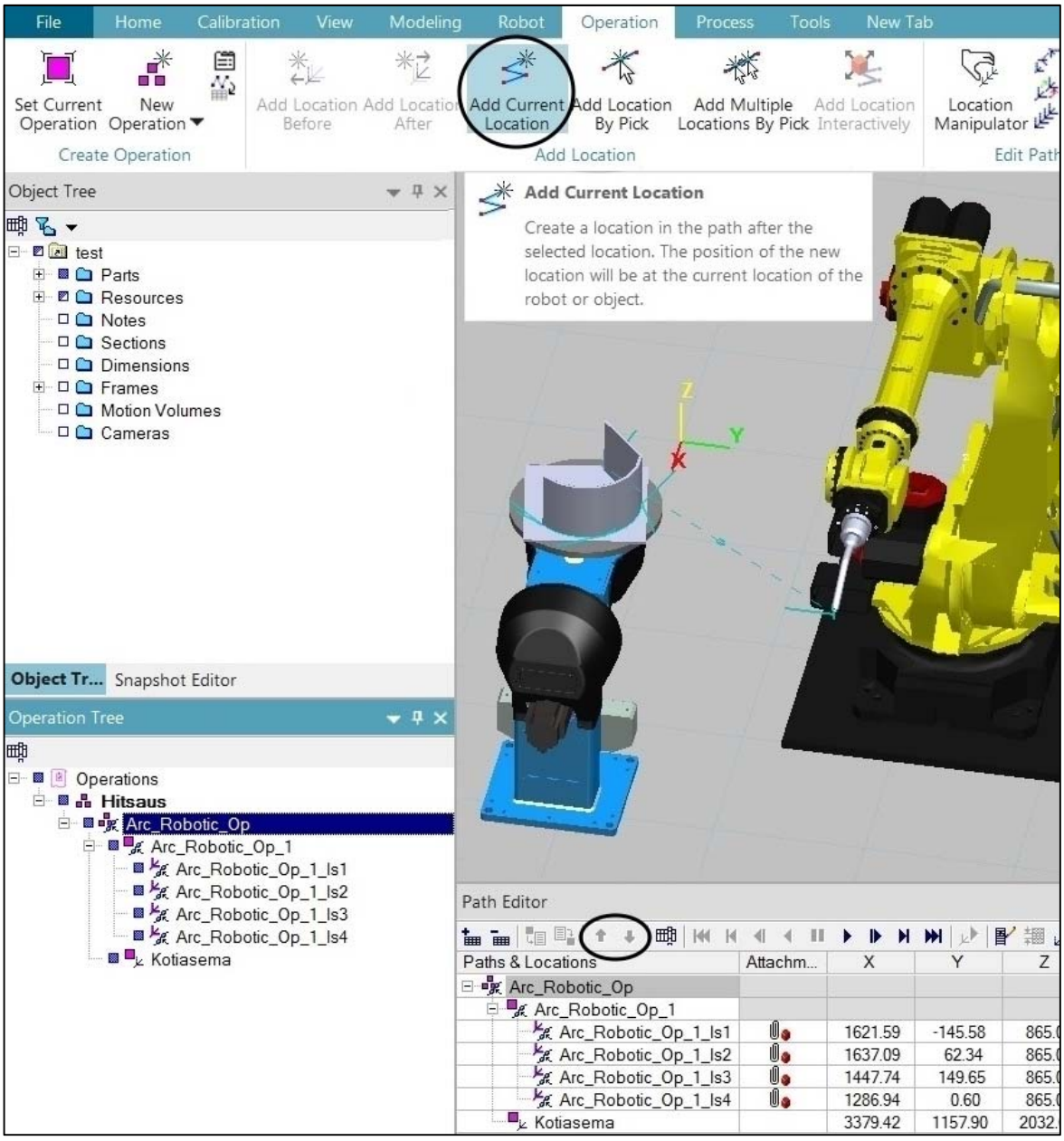
Tarvittaessa voidaan vielä muuttaa hitsauspistoolin asentoa jokaisessa liikepisteessä erikseen klikkaamalla haluttua pistettä Path Editorissa tai operaatiopuussa ja valitsemalla **Torch Alignment**, jolloin avautuu kuvan 41 mukainen ikkuna.



Kuva 41. Hitsauspistoolin asennon muuttaminen.

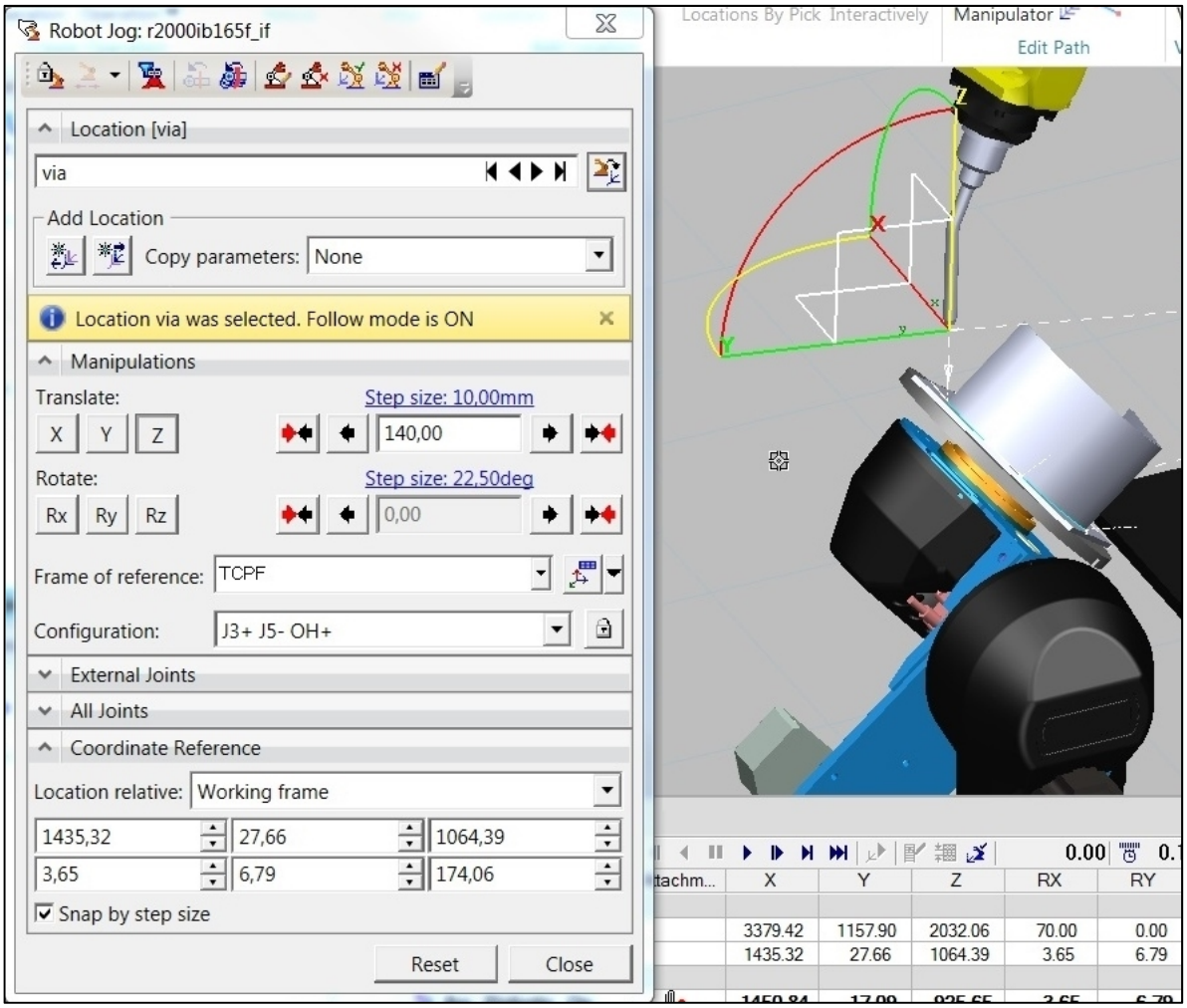
Valitsemalla **Follow Mode**n aktiiviseksi, nähdään hitsauspistoolin asennon muuttuminen **Arc Alignment** -arvoja muutettaessa. Säädetään arvot halutuksi ja valitaan seuraava piste nuolella. Kun halutut muokkaukset on tehty → **Close**. Kuvien esimerkissä jokaiseen pisteeseen asetettiin **Spin angle** -180 asteeseen ja muiden arvojen annettiin olla nolla. Kun simulointi nyt käynnistetään, hitsauspistoolin pitäisi seurata hitsaussaumaa oikeassa asennossa ja pyörityspöydän pitäisi pyöriä hitsauksen edetessä.

Pisteitä voi lisätä **Operation**-välilehdellä **Add Location** -kohdasta esimerkiksi valitsemalla **Add Current Location**, jolloin robotin nykyinen sijainti lisätään operaatiopuuhun. Ohjelma nimeää uudet pisteet via, via1, via2 jne. Kuvan 42 esimerkissä lisätty piste on jälkeenpäin nimetty Kotiasemaksi. Pisteen paikkaa Path Editorissa voidaan muuttaa raahaamalla ja pudottamalla se haluttuun kohtaan. Jos jokin piste halutaan kopioida siirtämisen lisäksi, painetaan samalla Ctrl. Esimerkissä Kotiasema kopioitiin myös ohjelman alkuun. Pisteitä voidaan siirtää ylös ja alas myös ohjelman nuolinäppäimillä.



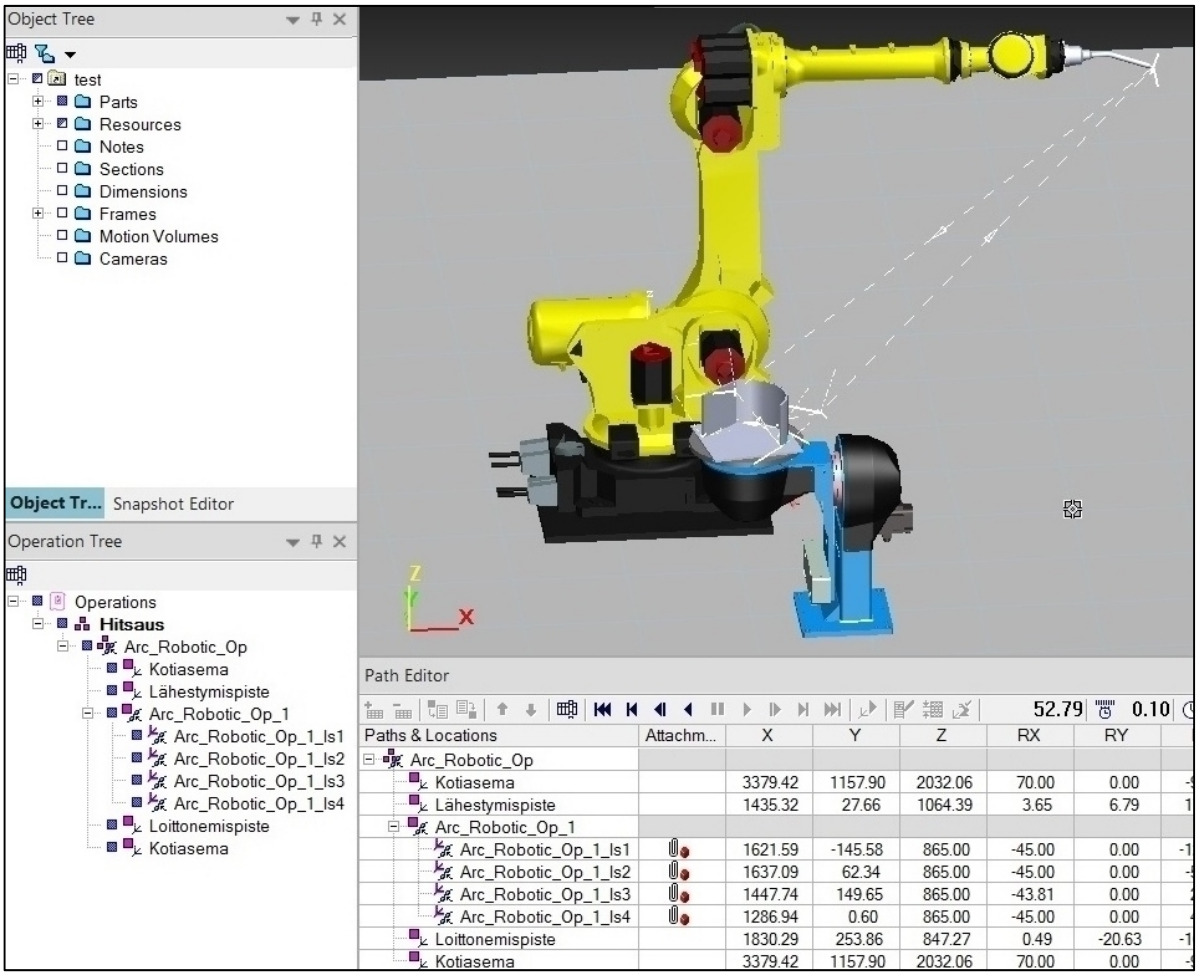
Kuva 42. Robotin nykyisen sijainnin lisääminen liikerataan.

Pisteitä voidaan lisätä myös ennen tai jälkeen tietyn pisteen **Add Location Before** ja **Add Location After** -toiminnoilla. **Operation**-välilehden lisäksi nämä toiminnot löytyvät klikkaamalla haluttua pistettä Path Editorissa hiiren oikealla ja valitsemalla haluttu pisteenlisäystapa. **Add Location After** ja **Add Location Before** -toiminnot avaavat kuvan 43 mukaisen ikkunan, jolloin pisteen paikkaa voidaan muuttaa halutuksi, joko **Manipulations**-kohdasta tai liikuttamalla robotti työkaluun ilmestyneen koordinaatiston avulla. **Configuration**-pudotusvalikosta voidaan vaihtaa robotin asentoa työkalun pysyessä paikallaan. Kun haluttu piste ja robotin asento on saavutettu → **Close**.



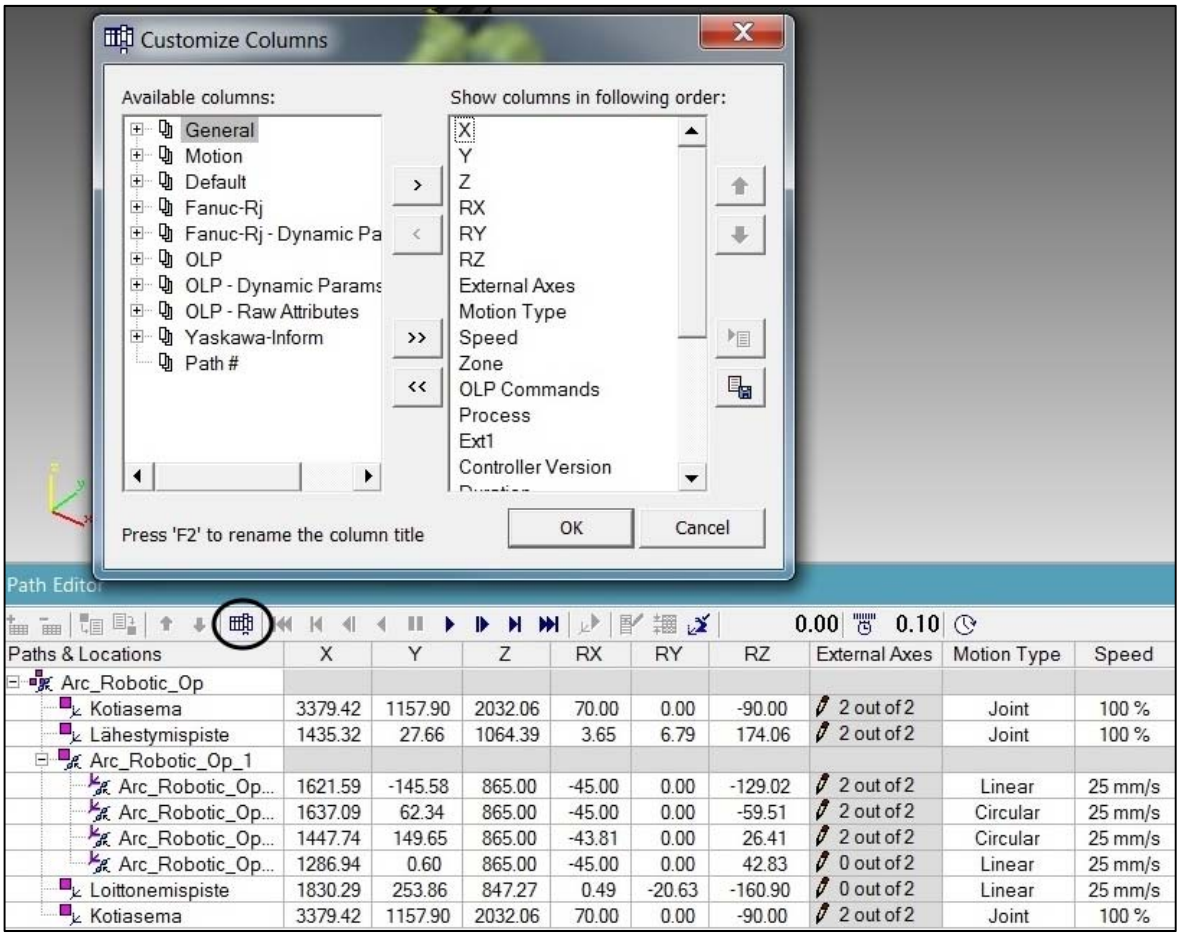
Kuva 43. Liikepisteen lisääminen ja sijainnin muokkaaminen.

Kuvassa 44 operaation alkuun lisättiin **Add Location After** -toiminnolla Kotiaseman jälkeen Lähestymispiste. Lisäksi loppuun ennen robotin siirtymistä Kotiasemaan lisättiin **Add Location Before** -toiminnolla Loittonemispiste. Pisteitä voi lisätä myös **Add Location By Pick** ja **Add Multiple Locations By Pick** -toiminnoilla, jolloin liikepisteitä voi lisätä valitsemalla hiirellä halutut kohdat mallista.



Kuva 44. Valmis hitsausesimerkki.

Path Editorissa näkyy luotujen pisteiden koordinaatit, **Motion Type** -kohtaan voidaan määrittää liiketyyppi esim. lineaarinen, ympyräkaari tai nivelliike, **Speed**-kohtaan voidaan määrittää nopeus prosentteina maksiminopeudesta (jos liiketyyppi on Joint) tai nopeus mm/s (jos liiketyyppi on Linear tai Circular). Path Editoriin voidaan lisätä tai siitä voidaan poistaa sarakkeita ja sarakkeiden keskinäistä järjestystä voidaan muuttaa valitsemalla **Customize Columns** kuten kuvassa 45.

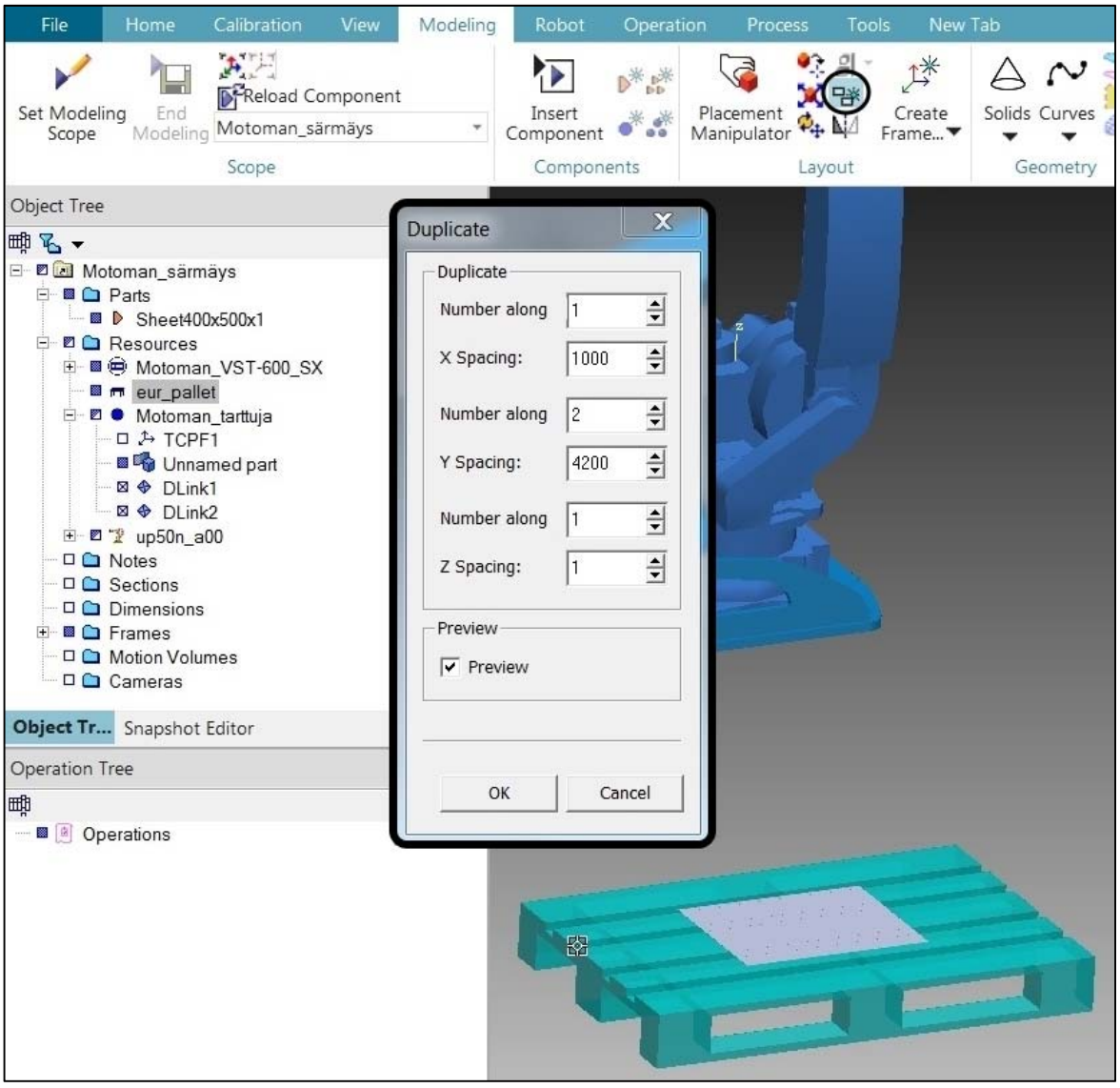


Kuva 45. Path Editorin sarakkeiden muokkaus.

7.2 Kappaleenkäsittelyesimerkki

Seuraavassa esimerkissä soluun on tuotu robotti, pyöritysjalka, imukuppitarttuja, EUR-lava ja ohutlevy. Tarttujasta on tehty työkalu kuten kohdassa neljä sillä erotuksella, että **Tool Type** -kohtaan on valittu **Gripper**, jolloin **Tool Definition** -ikkunaan avautuu **Gripping Entities** -kohta, johon määritetään se osa tai ne osat työkalusta, jotka tarttuvat kappaleeseen. Pyöritysjalan kinematiikka, robotin kiinnittäminen jalkaan ja ulkoisen akselin luominen on tehty kuten kohdissa viisi ja kuusi.

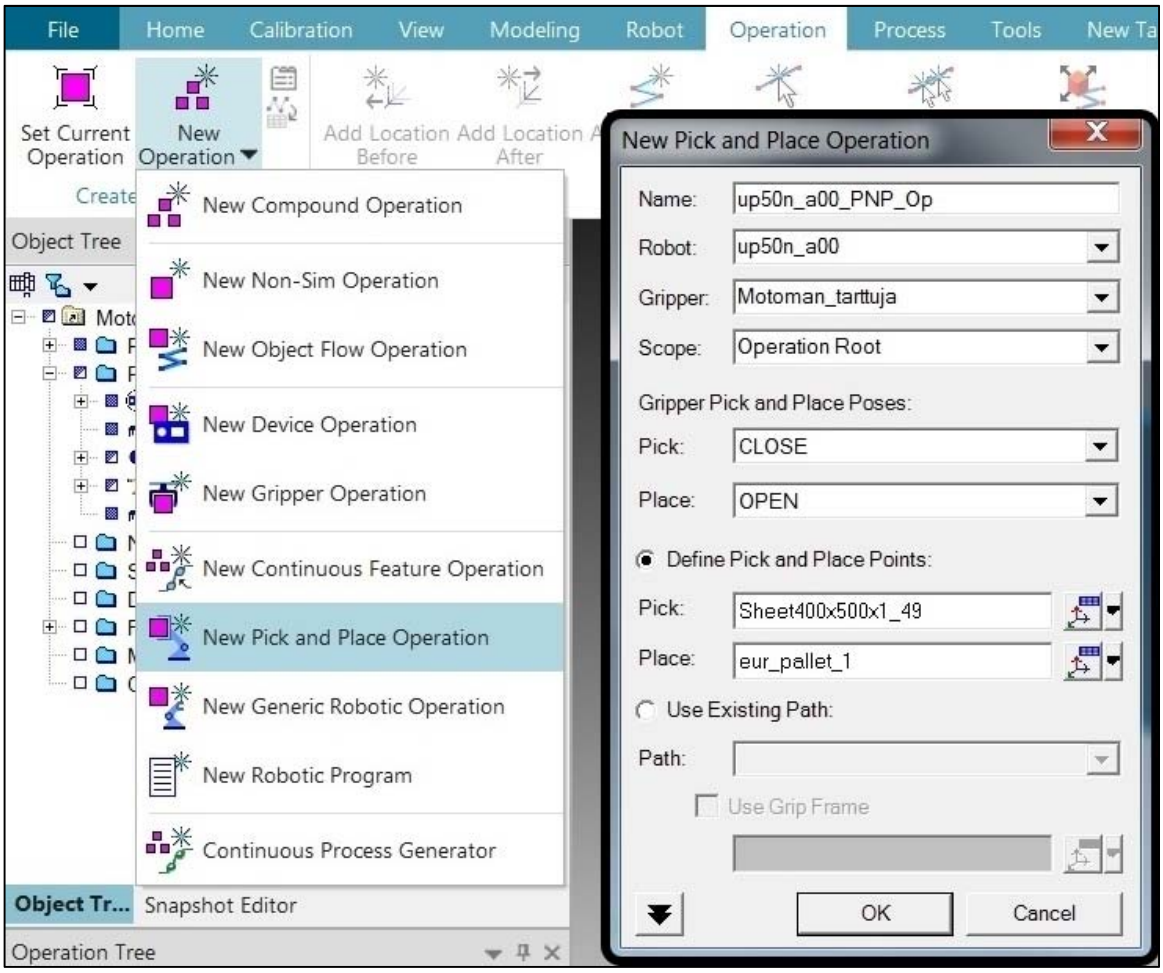
Jos jotakin samaa komponenttia tarvitaan solussa useampia, ei niitä tarvitse tuoda ohjelmaan useampaan kertaan, vaan komponentti pystytään kopioimaan klikkaamalla ensin haluttu komponentti aktiiviseksi ja valitsemalla **Modeling**-välilehdeltä **Layout**-kohdasta **Duplicate Objects**, jolloin avautuu kuvan 46 mukainen ikkuna.



Kuva 46. Kappaleiden kopioiminen.

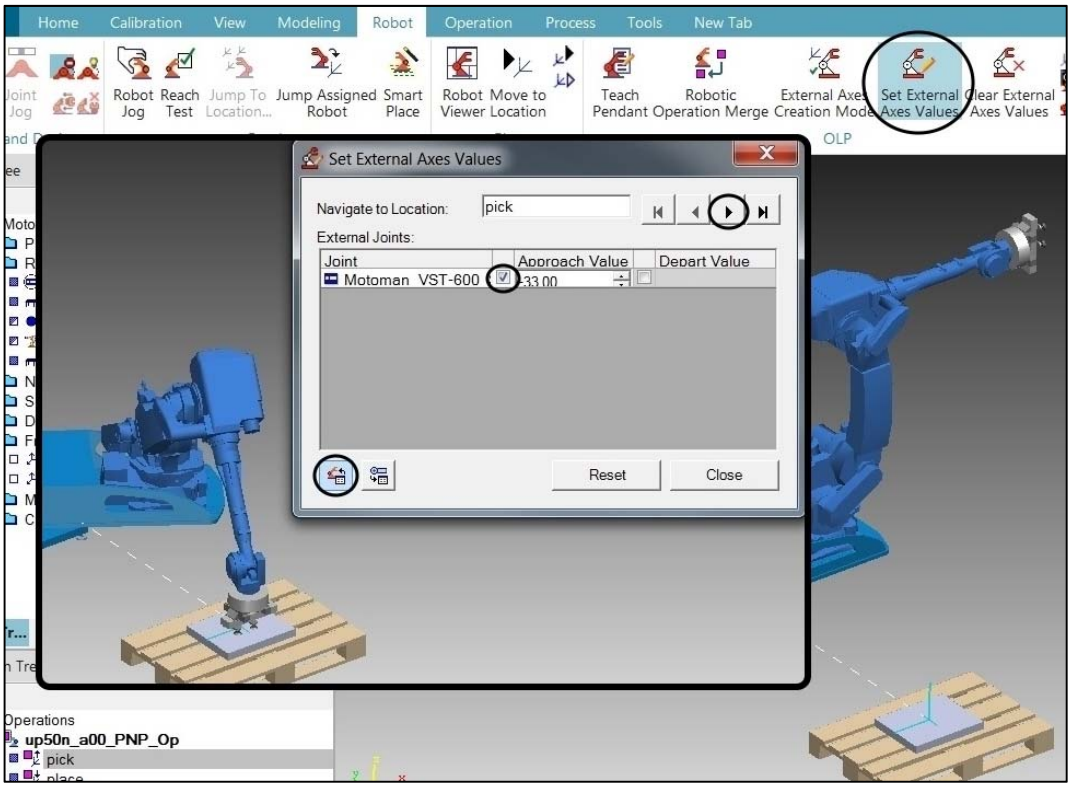
Ikkunaan määritetään, kuinka monta kappaletta halutaan minkäkin akselin suuntaan ja millä etäisyydellä niiden halutaan olevan toisistaan. **Preview** kannattaa pitää valittuna, jotta **Graphic Viewer**issä nähdään, mihin kopiot tulevat. → **OK**. Näin soluun lisättiin toinen EUR-lava ja pino ohutlevyjä.

Operation-välilehdeltä **Create Operation** -kohdasta avataan **New Operation** -pudotusvalikko, josta valitaan **New Pick and Place Operation**, jolloin avautuu kuvan 47 mukainen ikkuna, johon määritetään operaation nimi, robotti, tarttuja ja käyttöalue. **Gripper Pick and Place Poses** -kohtaan määritetään tarttujan asennot tarttumis- ja irrottamisvaiheissa, mutta tässä esimerkissä näillä ei ole merkitystä, sillä tarttujassa ei ole liikkuvia osia, eikä sille ole määritelty asentoja. **Define Pick and Place Points** -kohtaan valitaan mihin tartutaan ja mihin kappale viedään → **OK**.



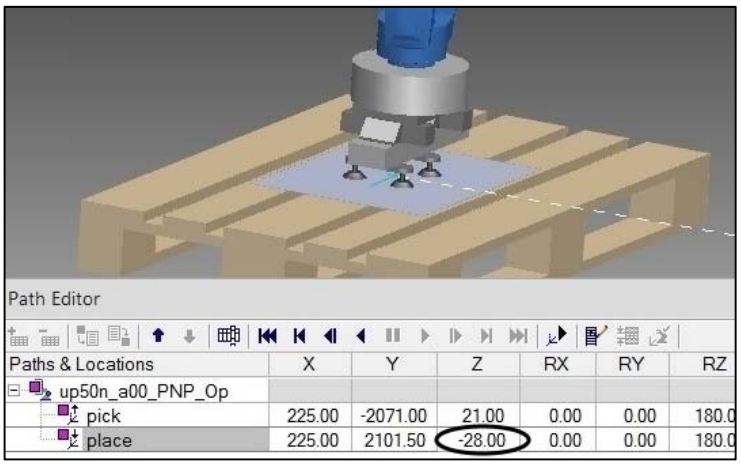
Kuva 47. Kappaleenkäsittelyoperaation luominen.

Tässä vaiheessa robotti ei vielä yllä tarttumaan levyyn, joten otetaan käyttöön pyöritysjalka. Klikataan se piste aktiiviseksi, jossa pyöritysjalka halutaan ottaa käyttöön. Valitaan sitten **Robot**-välilehdeltä **OLP**-kohdasta **Set External Axes Values**, jolloin avautuu kuvan 48 mukainen ikkuna. Valitaan **Approach Value** ja määritetään pyöritysjalan asento asteina. **Follow Mode** kannattaa pitää päällä, koska silloin robotti tarttuu kappaleeseen, kun se on sopivalla etäisyydellä. Jos pyöritysjalkaa halutaan käyttää muissakin pisteissä, nuolesta voidaan siirtyä seuraavaan. Kun tarvittavat muokkaukset on tehty → **Close**.



Kuva 48. Pyöritysjalan käyttöönotto.

Operaatio voidaan nyt simuloida lisäämällä se Path Editoriin ja painamalla **Play**. Koska **pick**-piste on ohutlevyn pinnassa ja **place**-piste määritettiin toisen EUR-lavan pintaan, levy uppoaa lavan sisään. **Place**-pistettä voidaan nostaa klikkaamalla sitä hiiren oikealla ja valitsemalla **Placement Manipulator** tai muuttamalla sen Z-akselin arvoa Path Editorissa levyn paksuuden verran positiiviseen suuntaan kuten kuvassa 49.

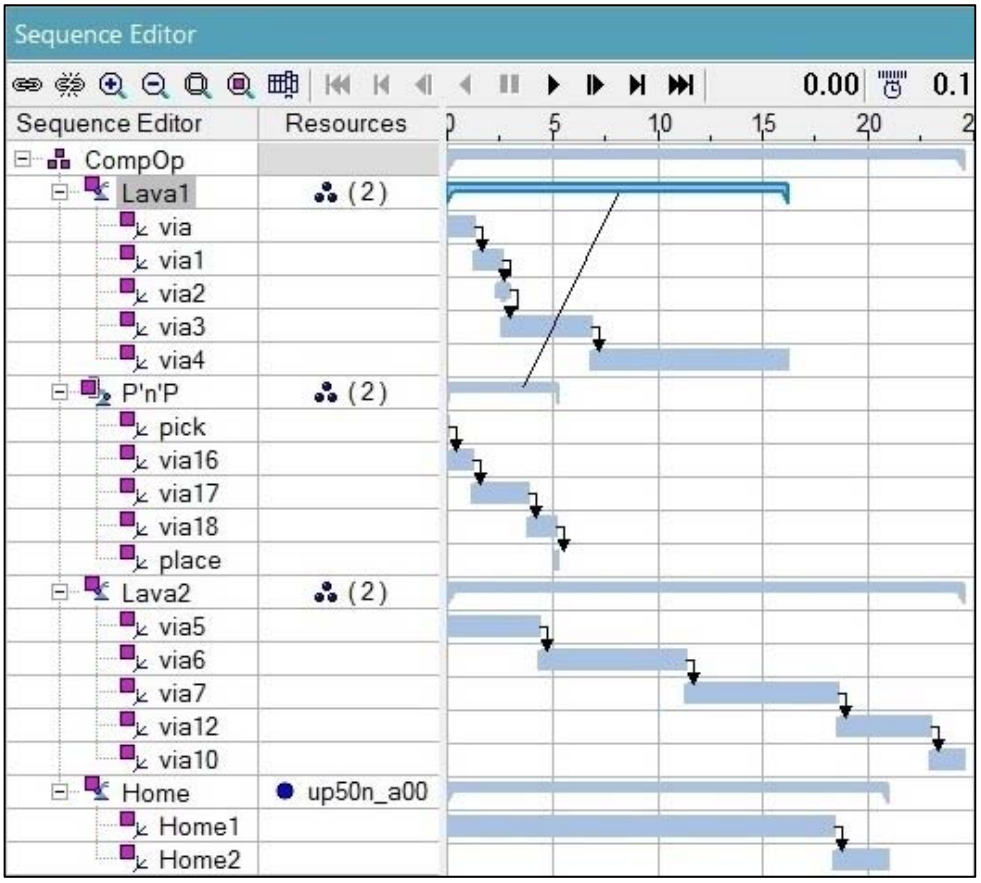


Kuva 49. Liikepisteen sijainnin muuttaminen Path Editorissa.

Uusia liikepisteitä voidaan nyt tarpeen mukaan lisätä ja niiden liiketyyppiä ja -nopeutta muokata kuten edellisessä esimerkissä.

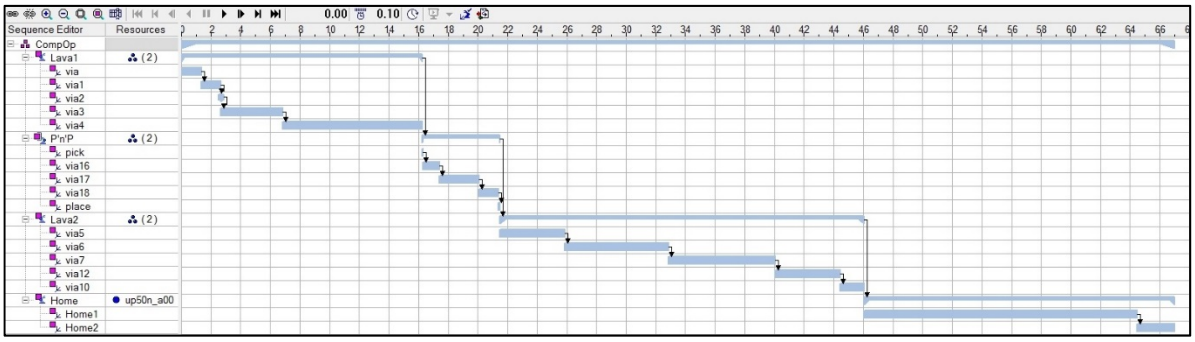
7.3 Sequence Editor

Path Editoria voidaan käyttää tietyn operaation tai ohjelman muokkaamiseen ja simuloimiseen. Sequence Editoria käytetään silloin, kun halutaan yhdistää useampia operaatioita yhdeksi simuloitavaksi kokonaisuudeksi. Jotta operaatio saadaan siirrettyä Sequence Editoriin, klikataan haluttua operaatiota operaatiopuussa hiiren oikealla ja valitaan **Set Current Operation**. Saman yhdistelmäoperaation eri alaoperaatiot pystytään yhdistämään samoin kuin erilliset ylätason operaatiot. Erillisten yhdistelmäoperaatioiden alaoperaatioita ei pystytä yhdistämään. Kun halutaan yhdistää alaoperaatiot yhdeksi kokonaisuudeksi, vedetään Gantt-kaaviosta halutun operaation palkista viiva toiseen operaatioon, kuten kuvassa 50.



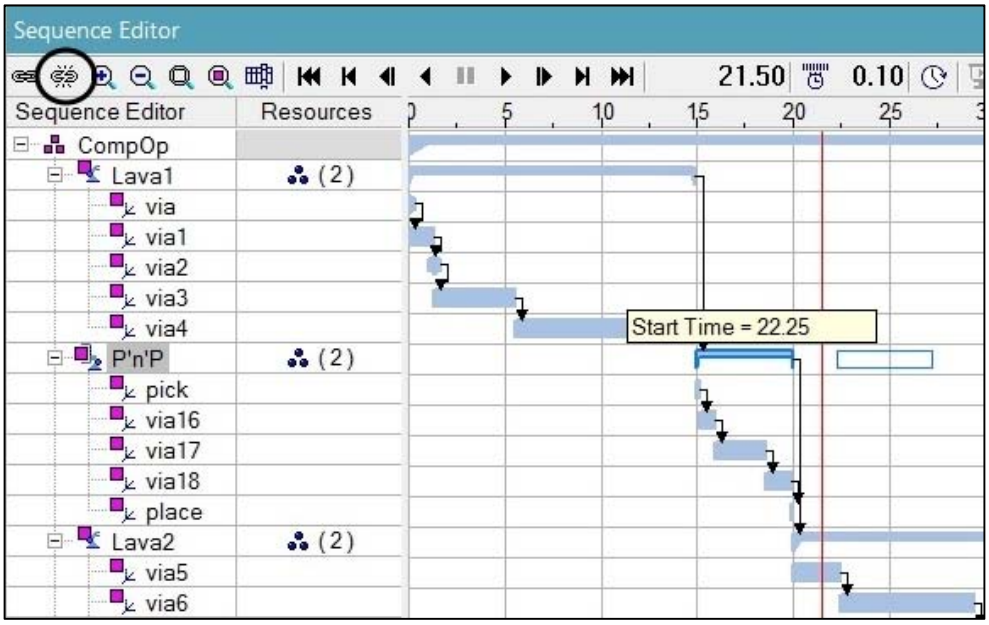
Kuva 50. Operaatioiden yhdistäminen.

Kuvassa 51 operaatiot on yhdistetty halutussa järjestyksessä. Kun simulointi nyt käynnistetään, operaatiot etenevät Gantt-kaavion mukaisessa järjestyksessä.



Kuva 51. Valmis operaatiosekvenssi.

Operaatioiden aloitusaikaa voidaan muuttaa vetämällä palkkia Gantt-kaaviossa eri paikkaan, kuten kuvassa 52. Jos operaatioiden halutaan alkavan samaan aikaan, vedetään operaation palkki samalle viivalle toisen operaation kanssa ilman operaatioiden yhdistämistä.



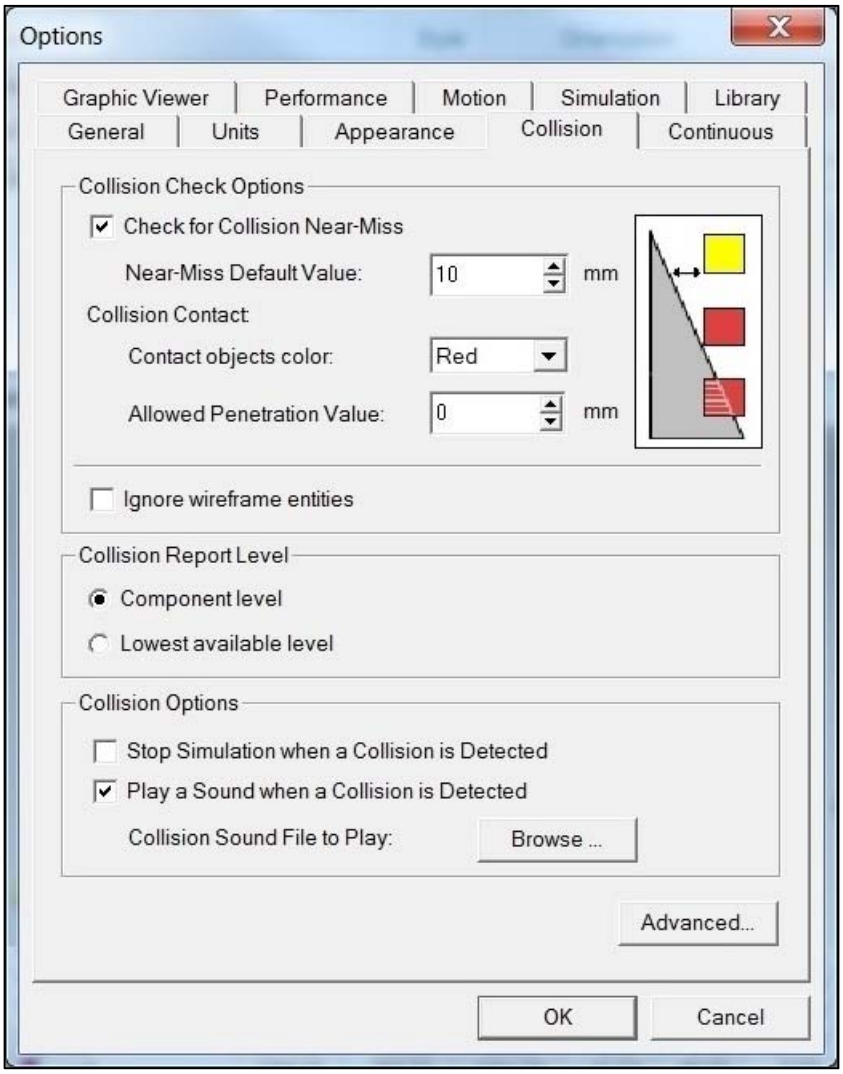
Kuva 52. Operaation aloitusajan muuttaminen.

Jos yhdistetyt operaatiot halutaan erottaa, valitaan operaatioita yhdistävä nuoli kaaviosta ja valitaan **Unlink** .

7.4 Törmäystarkastelu

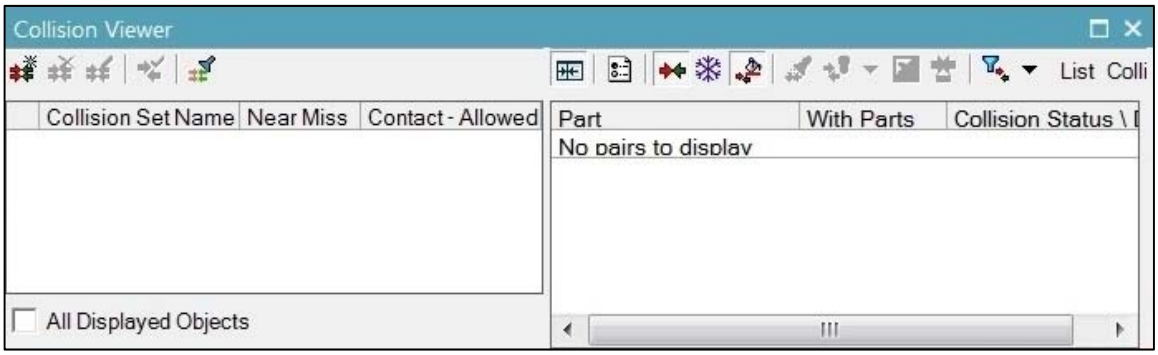
Törmäystarkasteluun tarkoitetulla työkalulla voidaan simuloinnin aikana selvittää, törmääkö robotti solussa oleviin muihin komponentteihin. Jos törmäystarkastelussa halutaan ottaa huomioon myös läheltä piti -tilanteet, valitaan **File → Options → Collision**, jolloin avautuu kuvan 53 mukainen näkymä. Valitaan **Check for Collision**

Near-Miss ja syötetään **Near-Miss Default Value** -kohtaan haluttu arvo → **OK**. **Collision**-välilehdellä voidaan muokata muitakin törmäystarkasteluun liittyviä asetuksia, kuten pysäytetäänkö simulaatio törmäyksen sattuessa.



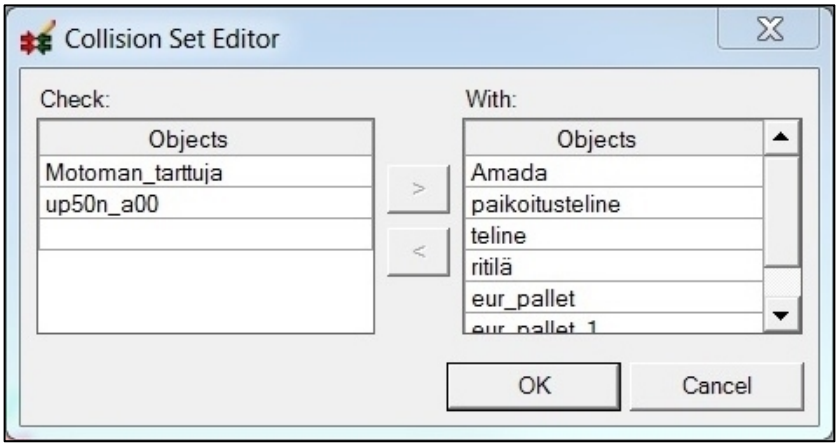
Kuva 53. Törmäystarkastelun asetuksia.

Valitaan **View**-välilehdeltä **Viewers**-kohdasta **Collision Viewer**, jolloin avautuu kuvan 54 mukainen näkymä. Valitaan **Collision Mode On** ja **Color Colliding Objects** , jolloin törmäävien komponenttien väri muuttuu. Jos halutaan, että törmäystarkastelu ottaa huomioon kaikki solussa näkyvillä olevat komponentit, valitaan **All Displayed Objects**. Tässä esimerkissä näin ei tehty, koska ohjelma pitää läheltä piti -tilanteena sitä, että robotti on kiinnitetty pyöritysjalkaan. Lisäksi ohjelmisto varoittaa siitä, että ohutlevy on EUR-lavan päällä.



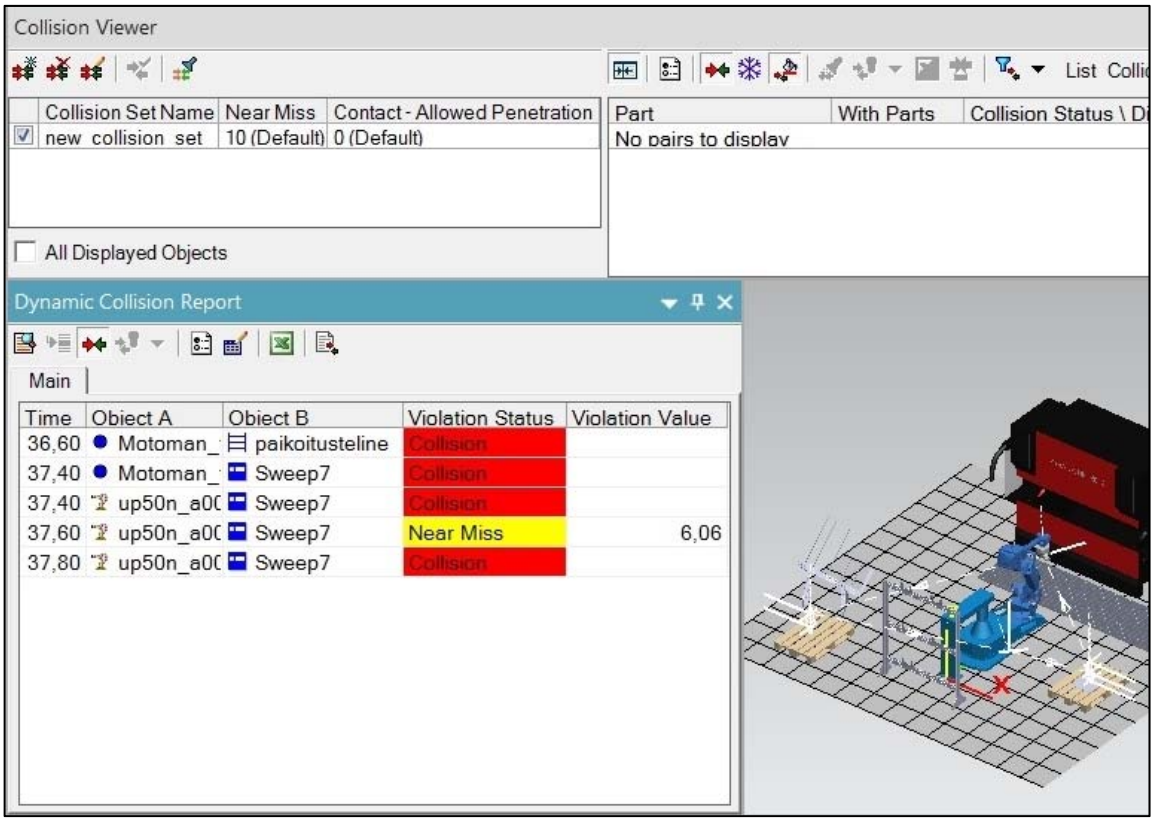
Kuva 54. Collision Viewer.

Valitaan sen sijaan **New Collision Set** , jolloin avautuu kuvan 55 mukainen ikkuna. **Check**-kohtaan valitaan ne objektit, joiden törmäystä halutaan tarkastella ja **With**-kohtaan valitaan, minkä kanssa nämä saattavat törmätä → **OK**. Kun simulaatio nyt käynnistetään, mahdolliset törmäykset ja läheltä piti -tilanteet nähdään **Graphic Viewer**issä.



Kuva 55. Collision Set Editor.

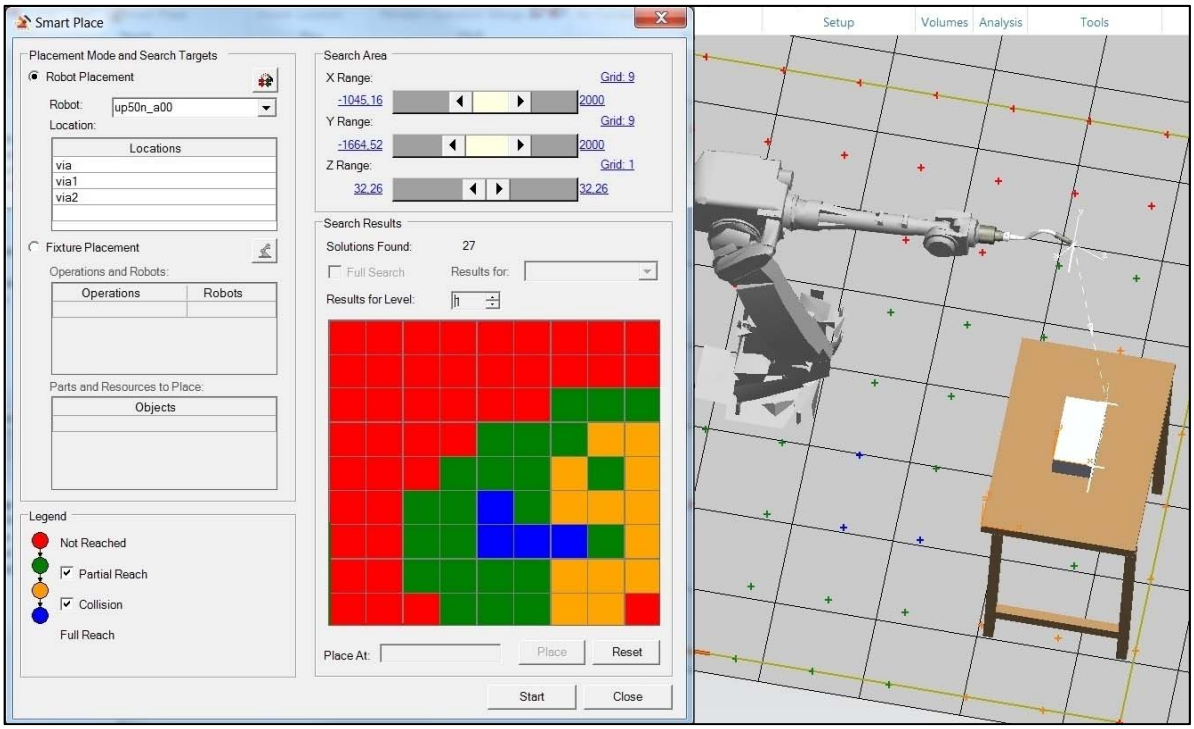
Törmäyksistä voidaan luoda raportti valitsemalla **Tools**-välilehdeltä **Analysis**-kohdasta **Collision** ja **Dynamic Collision Report**. Asetetaan tarkasteltava operaatio nykyiseksi operaatioksi (klikataan hiiren oikealla operaatiopuusta haluttua operaatiota ja valitaan **Set Current Operation**). Valitaan **Start Generating Report** **,** jolloin mahdollisista törmäyksistä ja läheltä piti -tilanteista saadaan kuvan 56 mukainen raportti. Raportin tiedoista voidaan tehdä Excel-tiedosto valitsemalla **Export To Excel** . Jos törmäyksistä halutaan yksityiskohtaisempaa tietoa, valitaan **Generate Detailed Report** .



Kuva 56. Törmäystarkastelun raportti.

7.5 Smart Place

Smart Place -työkalu auttaa robottisolujen suunnittelussa, kun halutaan löytää optimaalinen sijainti robotille tai muille laitteille siten, että robotti ulottuu tarvittaviin pisteisiin. Valitaan **Robot**-välilehdeltä **Reach**-kohdasta **Smart Place**. Valitaan **Robot**-kohtaan haluttu robotti ja **Locations**-kohtaan ne pisteet, joihin robotin pitäisi yltää. **Search Area** -kohtaan voidaan määrittää, miten suurelta alueelta optimaalista sijaintia etsitään. Valitaan **Start**, jolloin **Search Results** -kohdasta nähdään kuvan 57 mukaisesti, mistä paikasta robotti yltää haluttuihin pisteisiin (siniset ruudut) ja mistä ei (punaiset ruudut). Lisäksi nähdään, mistä robotti yltää osittain (vihreät ruudut) ja missä robotti törmää solun muihin komponentteihin (oranssit ruudut). Robotin voi siirtää klikkaamalla haluttua ruutua ja valitsemalla **Place**.

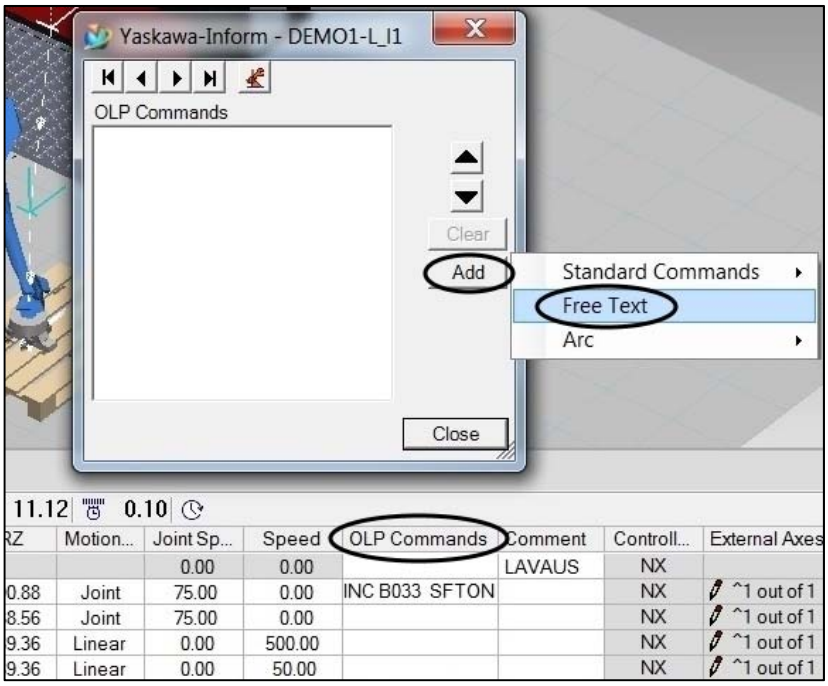


Kuva 57. Smart Place -työkalu.

Tässä työssä ei suunniteltu uutta solua, vaan keskityttiin pääasiassa jo olemassa olevaan soluun, joten **Smart Place** -työkalun toimintaan ei perehdytty tarkemmin. Lisätietoa löytyy tarvittaessa ohjelmiston Helpistä.

8. OLP-käskyjen lisääminen ja ohjelman luominen robotille

OLP- eli offline-ohjelmointikäskyjä voi lisätä Path Editorin **OLP Commands** -sarakkeeseen klikkaamalla saraketta sen liikepisteen rivillä, jonka jälkeen käskyt halutaan toteuttaa, jolloin avautuu kuvan 58 mukainen ikkuna.



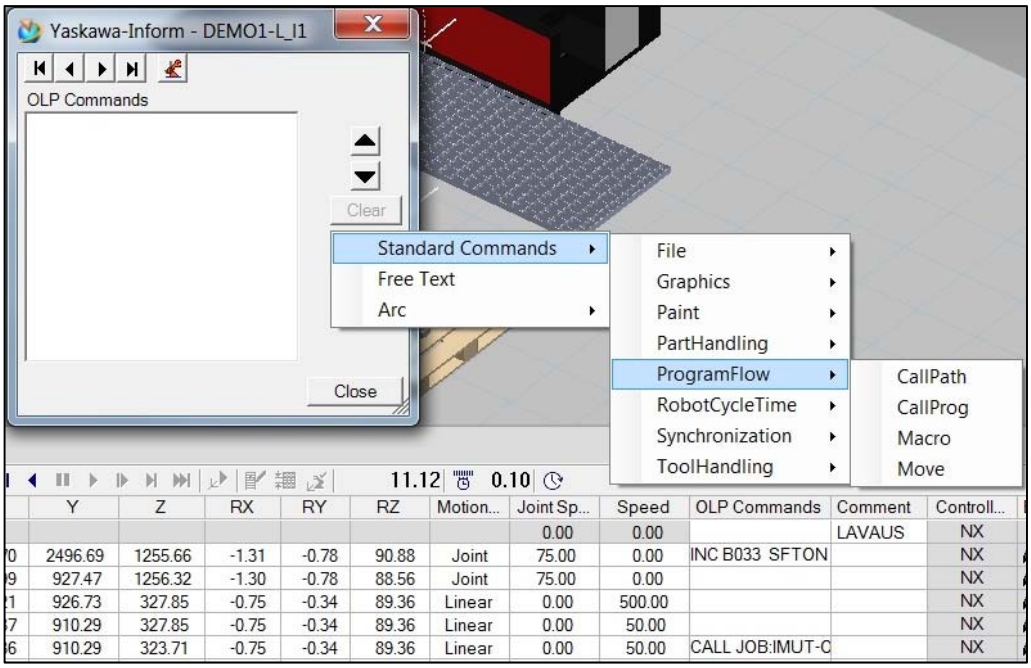
Kuva 58. OLP-käskyjen lisääminen.

Valitaan **Add → Free Text**, jolloin avautuu kuvan 59 mukainen ikkuna, johon käskyt voi kirjoittaa robotin omalla ohjelmointikielellä **→ OK → Close**.

Kuva 59. OLP-käskyjen kirjoittaminen robotin omalla ohjelmointikielellä.

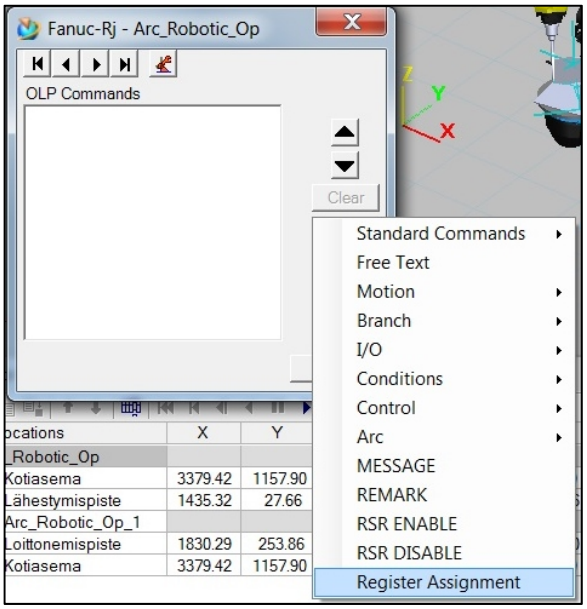
Ohjelmistossa on valmiina joitain geneerisiä OLP-käskyjä, joita voi lisätä valitsemalla **Add → Standard Commands**, jolloin avautuu kuvan 60 mukainen valikko. Ohjelmisto kääntää geneeriset käskyt robotin omalle kielelle ohjelmaa luotaessa, mutta tämä edellyttää, että robotin ohjain on asennettu RobotExpertiin. Robottikohtaisia ohjaimia

voi pyytää mm. robottien valmistajilta. Siemensin asiakkaat pystyvät lataamaan ohjaimia Siemensin Global Technical Access Centeristä (GTAC).



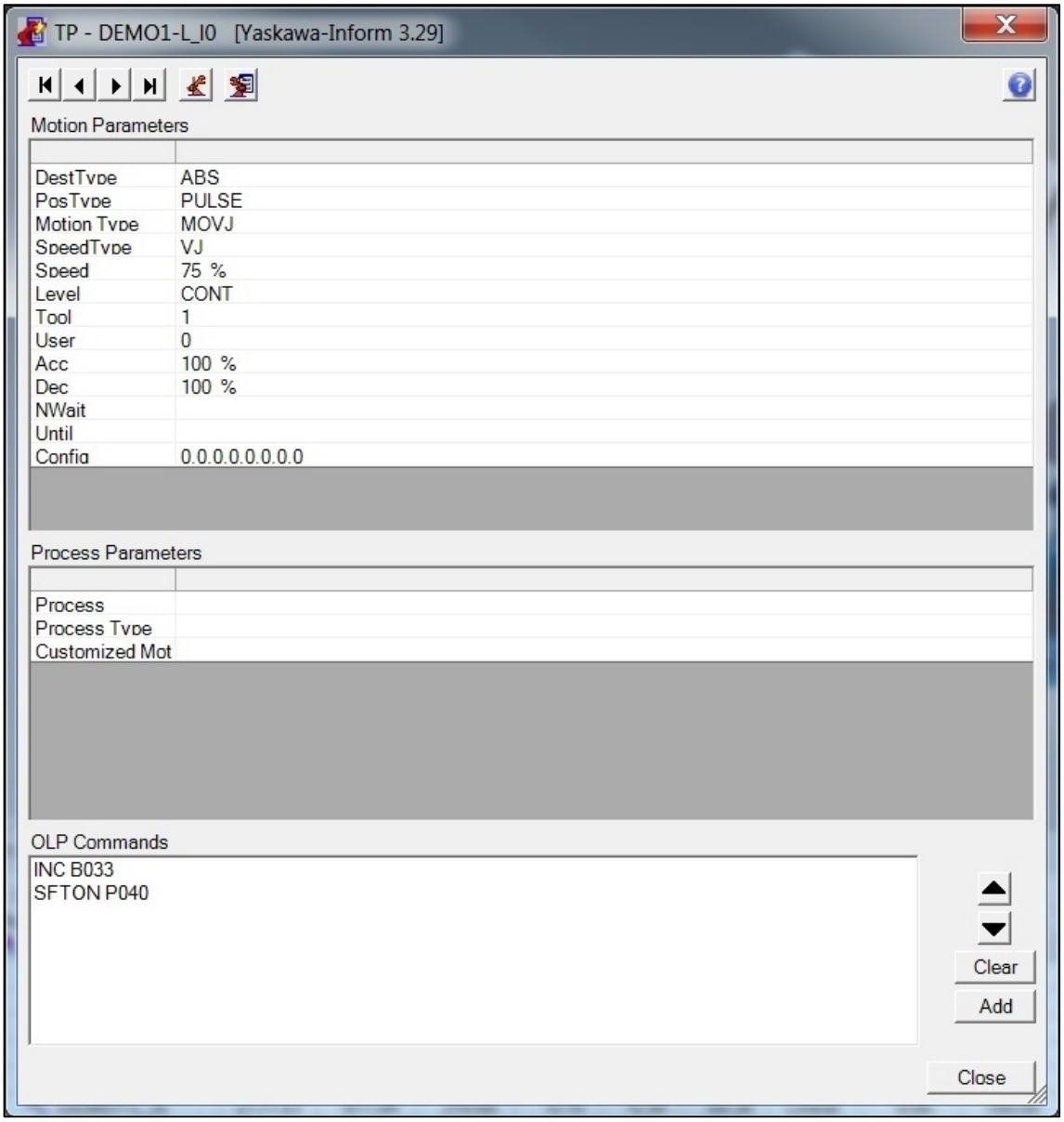
Kuva 60. Ohjelmiston omien OLP-käskyjen lisääminen.

Huomioi, että **Add**-painikkeesta avautuva valikko riippuu käyttämästäsi ohjaimesta. Esimerkiksi Fanuc RJ -ohjaimen kanssa valikko näyttää kuvan 61 mukaiselta. Jos käytössä on ohjelman oletusohjain, valikossa ei ole **Standard Commands** ja **Free Text** -vaihtoehtojen lisäksi muita vaihtoehtoja.



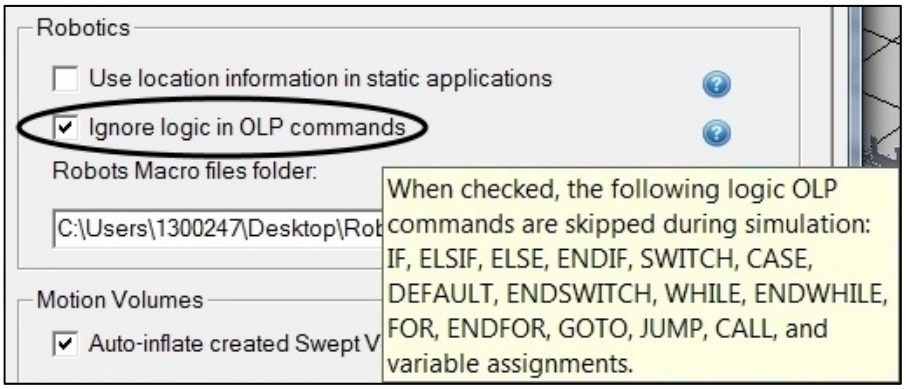
Kuva 61. OLP-käskyjen valikko Fanuc RJ -ohjaimella.

OLP-käskyjä voi lisätä myös **Teach Pendant**illa klikkaamalla liikepistettä Path Editorissa hiiren oikealla ja valitsemalla **Teach Pendant**, jolloin avautuu kuvan 62 mukainen ikkuna. Käskyt lisätään **OLP Commands** -kohtaan kuten edellä. **Teach Pendant**illa voidaan myös muokata liikepisteen muita parametreja, kuten liiketyyppiä ja -nopeutta sekä työkalu- ja käyttäjäkoordinaatistoa.



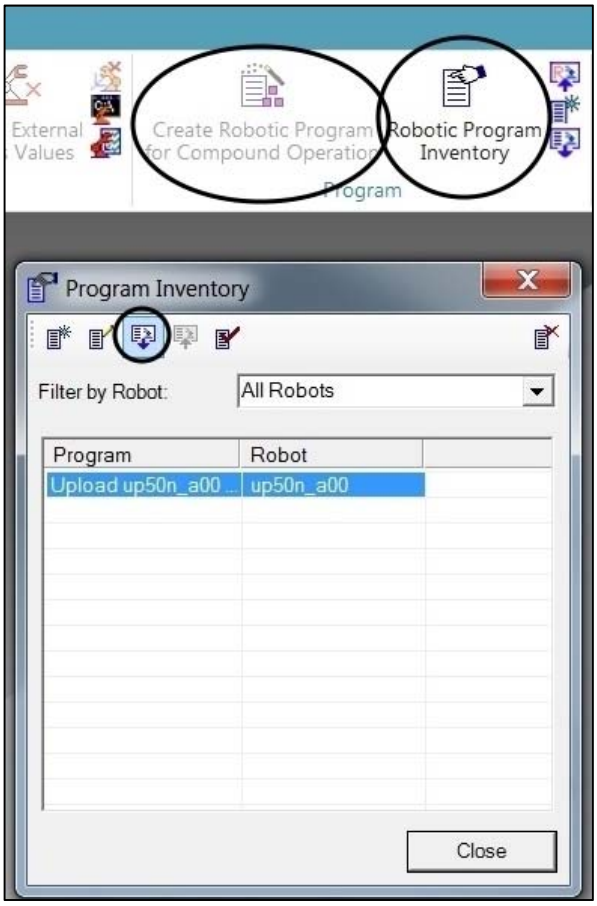
Kuva 62. OLP-käskyjen lisääminen Teach Pendantilla.

Joskus voi olla tarpeen simuloida pelkästään robotin liikkeitä ilman, että loogisia OLP-käskyjä otetaan huomioon. Tällöin valitaan **File → Options → Motion**. **Robotics**-kohtaan valitaan **Ignore logic in OLP commands → OK**, jolloin simulaation aikana hypätään kuvan 63 mukaisten käskyjen yli.



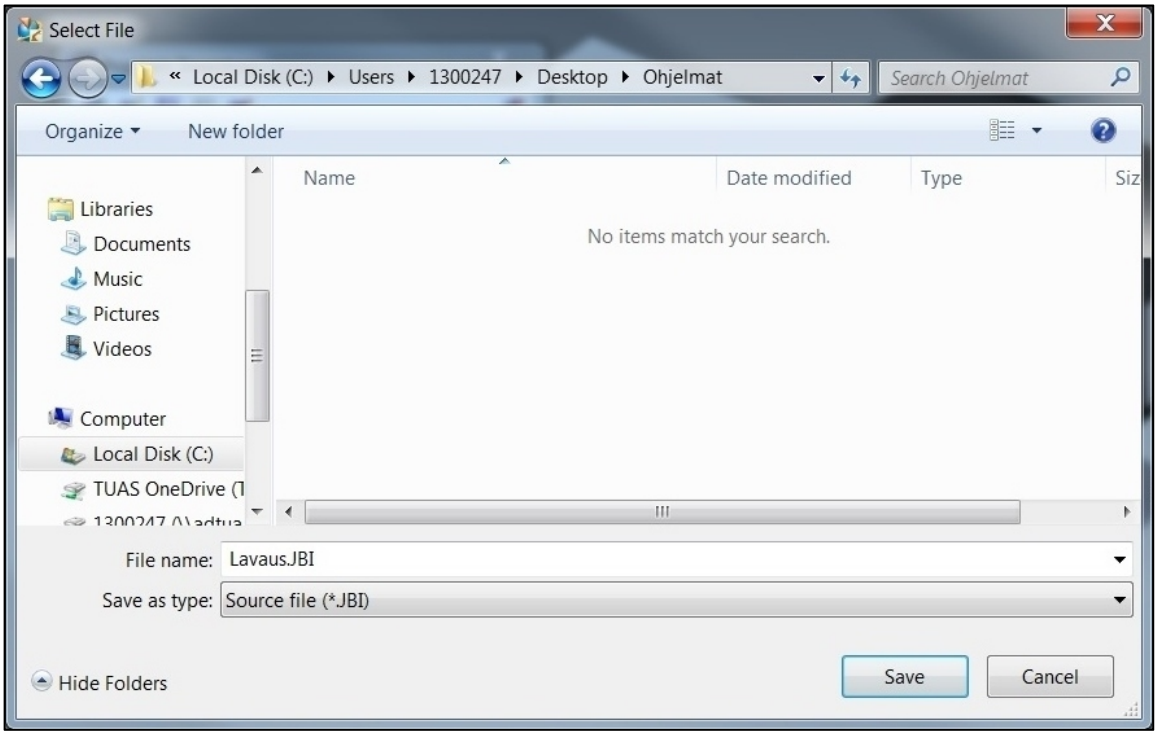
Kuva 63. Loogisten OLP-käskyjen huomiotta jättäminen.

Kun tarvittavat operaatiot on saatu tehtyä ja OLP-käskyt lisätty, liikepisteet vielä opetetaan robotille. Valitaan Path Editorista **Auto Teach** ja käynnistetään simulaatio, jolloin pisteiden koordinaatit ja robotin konfiguraatio kussakin pisteessä tallentuvat ohjelmaa varten. Nyt voidaan luoda ohjelma, joka pystytään siirtämään oikealle robotille. Jos ohjelma koostuu useammasta operaatiosta, klikataan haluttu yhdistelmäoperaatio aktiiviseksi ja valitaan **Robot**-välilehdeltä **Program**-kohdasta **Create Robotic Program for Compound Operation**. Luotu ohjelma löytyy valitsemalla **Robotic Program Inventory**, jolloin avautuu kuvan 64 mukainen ikkuna.



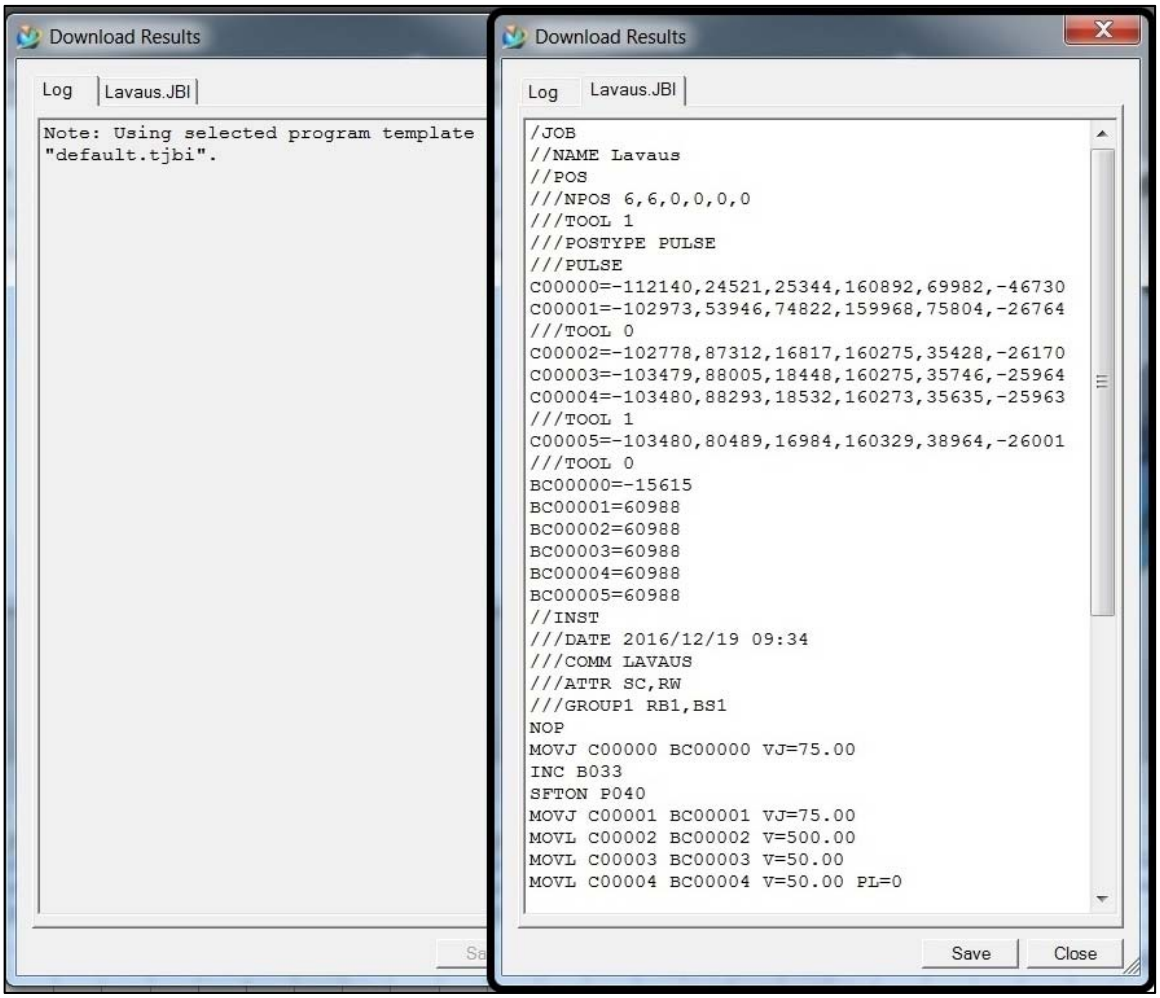
Kuva 64. Ohjelman luominen robotille.

Klikataan listasta ohjelma aktiiviseksi ja valitaan **Download to Robot**, jolloin avautuu kuvan 65 mukainen ikkuna, johon annetaan ohjelmalle nimi ja valitaan tallennuskansio. Esimerkin ohjelma on tehty Motoman-robotille, joten tiedostomuoto on JBI ja ohjelman nimi saa olla maksimissaan kahdeksan merkkiä pitkä → **Save**.



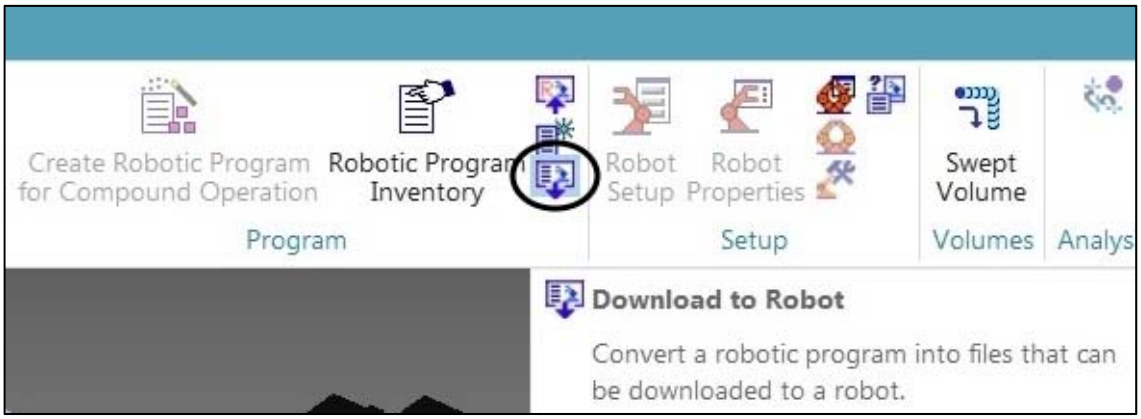
Kuva 65. Ohjelman nimeäminen ja tallentaminen tietokoneelle.

Tallentamisen jälkeen avautuu kuvan 66 mukainen ikkuna. **Log**-välilehdeltä nähdään, onko ohjelman lataaminen onnistunut (mahdolliset varoitukset ja muut huomiot) ja ohjelman välilehdeltä nähdään luotu ohjelma. Jos ohjelmaa halutaan muokata, se voidaan tehdä tässä ja valita sitten **Save**. Jos ohjelmaa ei haluta muokata, valitaan **Close**. JBI-tiedostot pystytään avaamaan tekstinkäsittelyohjelmilla, esimerkiksi Notepadilla, jolloin sitä pystytään muokkaamaan tarvittaessa myöhemmin.



Kuva 66. Ladattu ohjelma.

Yksittäisestä operaatiosta pystytään luomaan ohjelma klikkaamalla haluttu operaatio aktiiviseksi ja valitsemalla **Robot**-välilehdeltä **Program**-kohdasta **Download to Robot** kuten kuvassa 67. Tallentaminen käy kuten edellä.



Kuva 67. Ohjelman luominen yksittäisestä operaatiosta.

9. Ohjelman siirtäminen robotilta ja lataaminen robotille (esimerkkinä Motoman-robotti)

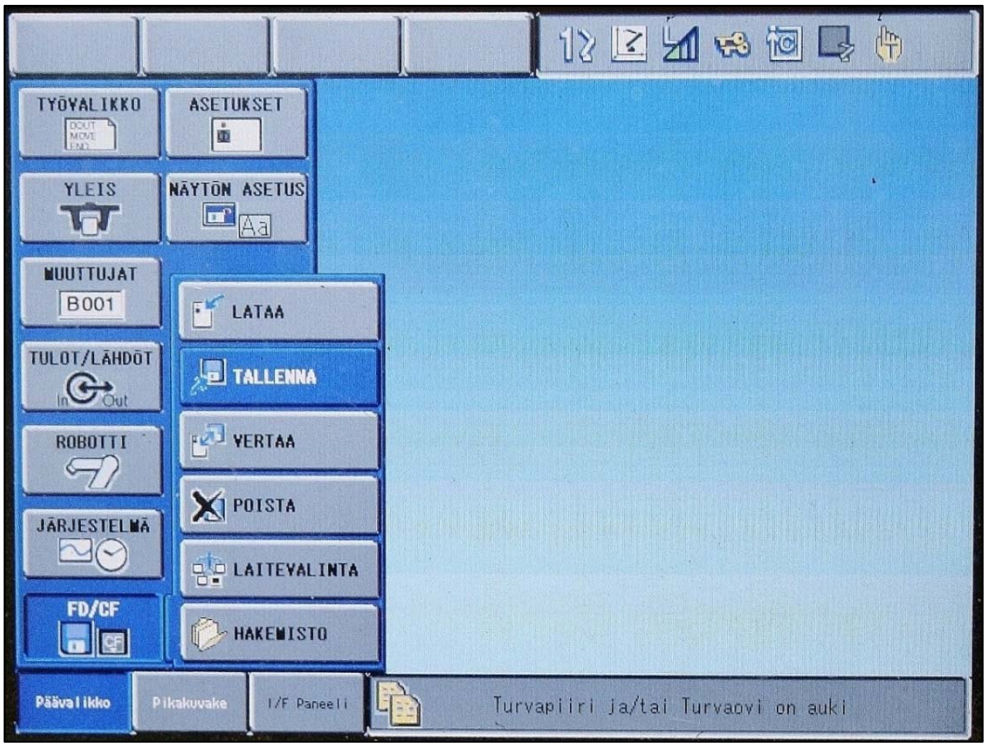
Valmiin ohjelman siirtäminen oikealta robotilta RobotExpert-ohjelmistoon ja RobotExpertillä luodun ohjelman lataaminen oikealle robotille vaatii robottikohtaisen ohjaimen asentamisen ohjelmistoon. Ohjelmansiirtotapa riippuu käytössä olevasta robotista ja ohjaimesta. Seuraavassa esimerkissä on käytetty Motoman UP50N -robottia, jossa on NX100-ohjauskeskus. Ohjelmien siirtämistä varten tarvitaan CompactFlash-muistikortti ja muistikortinlukija.

9.1 Ohjelman siirtäminen robotilta RobotExpert-ohjelmistoon

Kun ohjelma halutaan siirtää oikealta robotilta ohjelmistoon, asetetaan muistikortti robotin opetusyksikön sivussa olevaan CF-korttipaikkaan kuten kuvassa 68.

Kuva 68. Muistikortin asettaminen robotin opetusyksikköön.

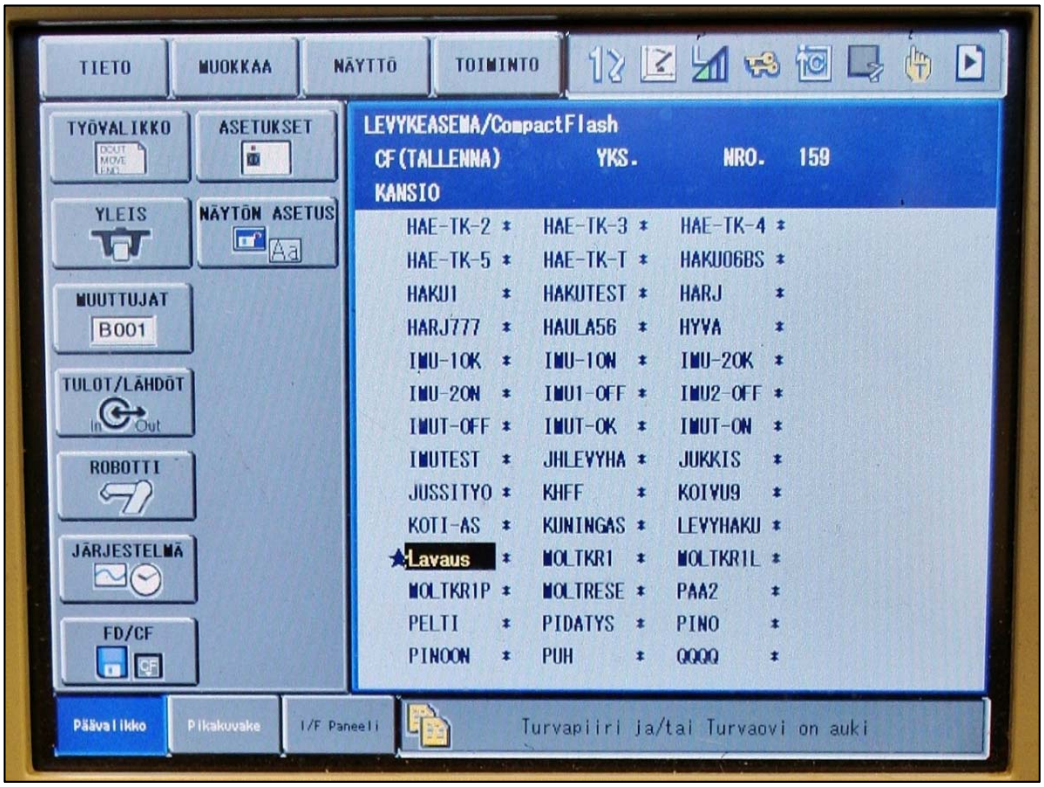
Valitaan **FD/CF → TALLENNA** kuten kuvassa 69, jolloin avautuu kuvan 70 mukainen valikko.



Kuva 69. Tallennusvalikon avaaminen.

Kuva 70. Työlistan avaaminen.

Valitaan **TYÖT** painamalla **SELECT**, jolloin avautuu kuvan 71 mukainen lista, josta etsitään työ, joka halutaan tallentaa muistikortille. Valitaan työ painamalla **SELECT**, jolloin sen eteen ilmestyy pieni tähti. Tallennettavaksi voidaan valita myös useampi työ kerralla, jolloin jokaisen valitun työn eteen ilmestyy edellä mainittu tähti.

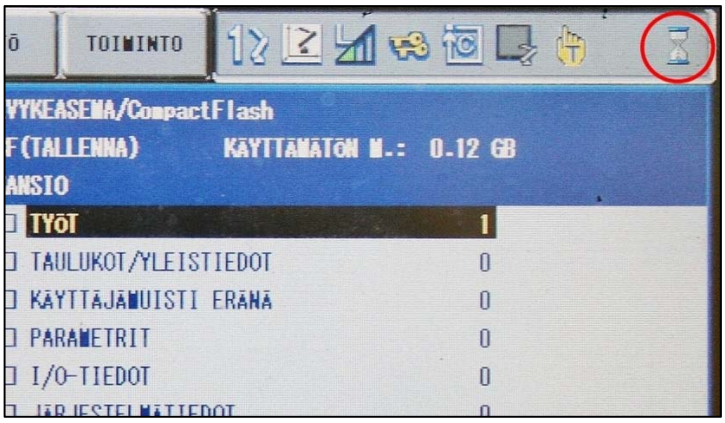


Kuva 71. Halutun työn etsiminen työlistasta.

Painetaan **ENTER**, jolloin näytölle avautuu kuvan 72 mukainen ikkuna. Valitaan **KYLLÄ**.

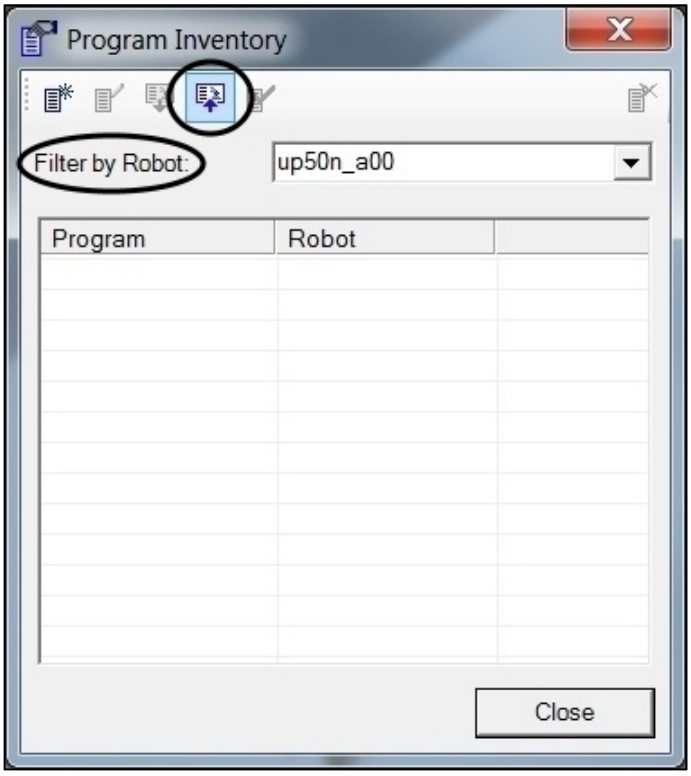
Kuva 72. Työn tallentaminen.

Työ on tallentunut, kun kuvan 73 mukainen tiimalasi on kadonnut näytön oikeasta yläkulmasta.



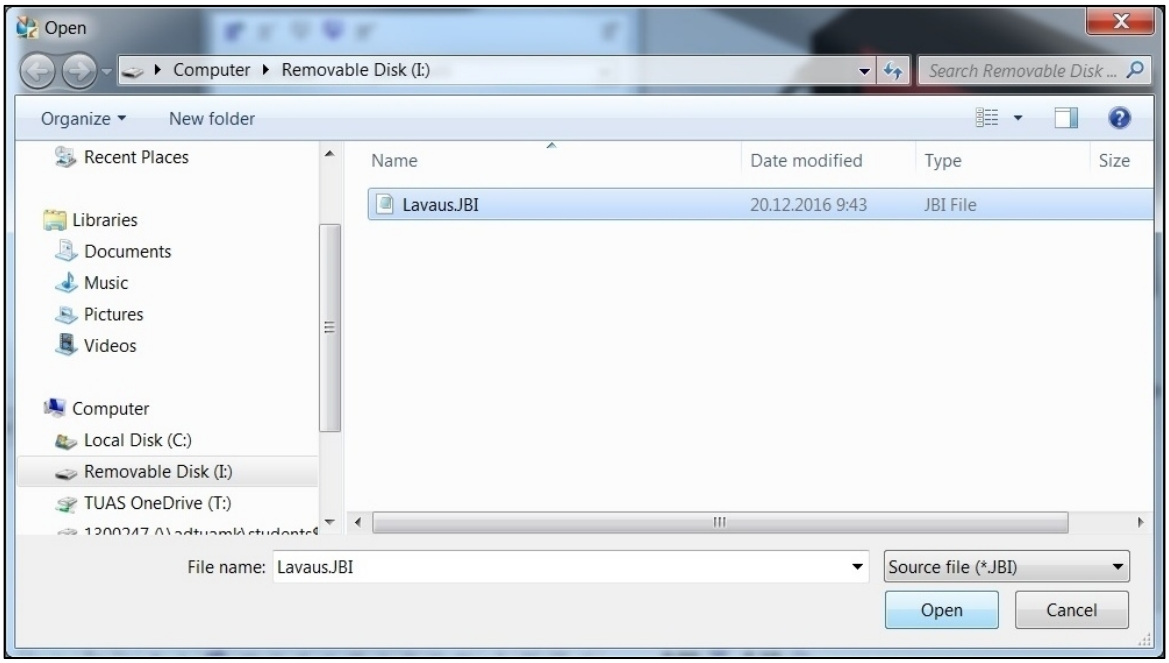
Kuva 73. Tallennus kesken.

Muistikortti voidaan nyt poistaa opetusyksiköstä. Ohjelma siirretään muistikortilta tietokoneelle muistikortinlukijan avulla. RobotExpert-ohjelmistossa valitaan **Robot**-välilehdeltä **Program**-kohdasta **Robotic Program Inventory**, jolloin avautuu **Program Inventory** -ikkuna kuten kuvassa 74.



Kuva 74. Ohjelman lataaminen virtuaaliselle robotille.

Filter by Robot -kohtaan valitaan robotti, jolle ohjelma halutaan ladata ja valitaan **Upload Program**, jolloin avautuu kuvan 75 mukainen ikkuna. Etsitään ohjelma tietokoneelta → **Open**.

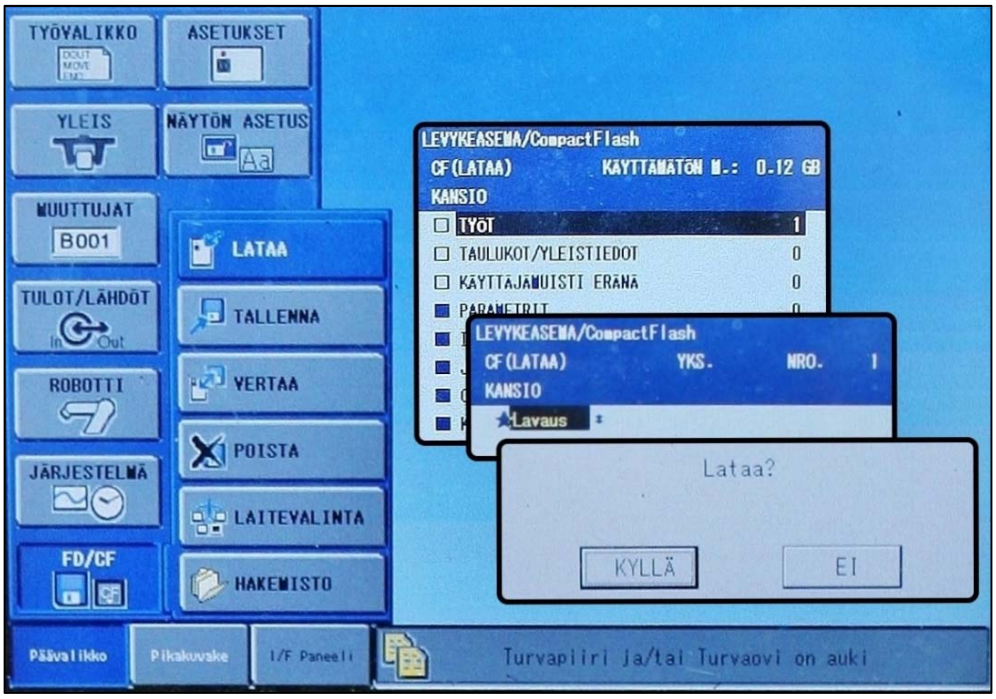


Kuva 75. Ohjelman hakeminen tietokoneelta.

Työ ilmestyy **Program Inventory** -listaan sekä operaatiopuuhun, josta se voidaan siirtää Path Editoriin simuloitavaksi ja muokattavaksi.

9.2 Ohjelman lataaminen RobotExpert-ohjelmistosta robotille

Kun ohjelma halutaan ladata ohjelmistosta oikealle robotille, se tallennetaan tietokoneelta muistikortille, joka asetetaan robotin opetusyksikköön kuten edellä. Valitaan **FD/CF → LATAA**. Valitaan **TYÖT**, jolloin avautuu lista muistikortilla olevista töistä. Valitaan työ, joka halutaan ladata robotille, jolloin työn eteen ilmestyy tähti, kuten työtä tallennettaessa. Painetaan **ENTER**, jolloin ohjain kysyy ”**Lataa?**”, valitaan **KYLLÄ**. Työn lataamisen vaiheet on esitetty kuvassa 76. Kun ohjelma on latautunut (tiimalasi näytön oikeasta yläreunasta on kadonnut), muistikortti voidaan poistaa opetusyksiköstä.



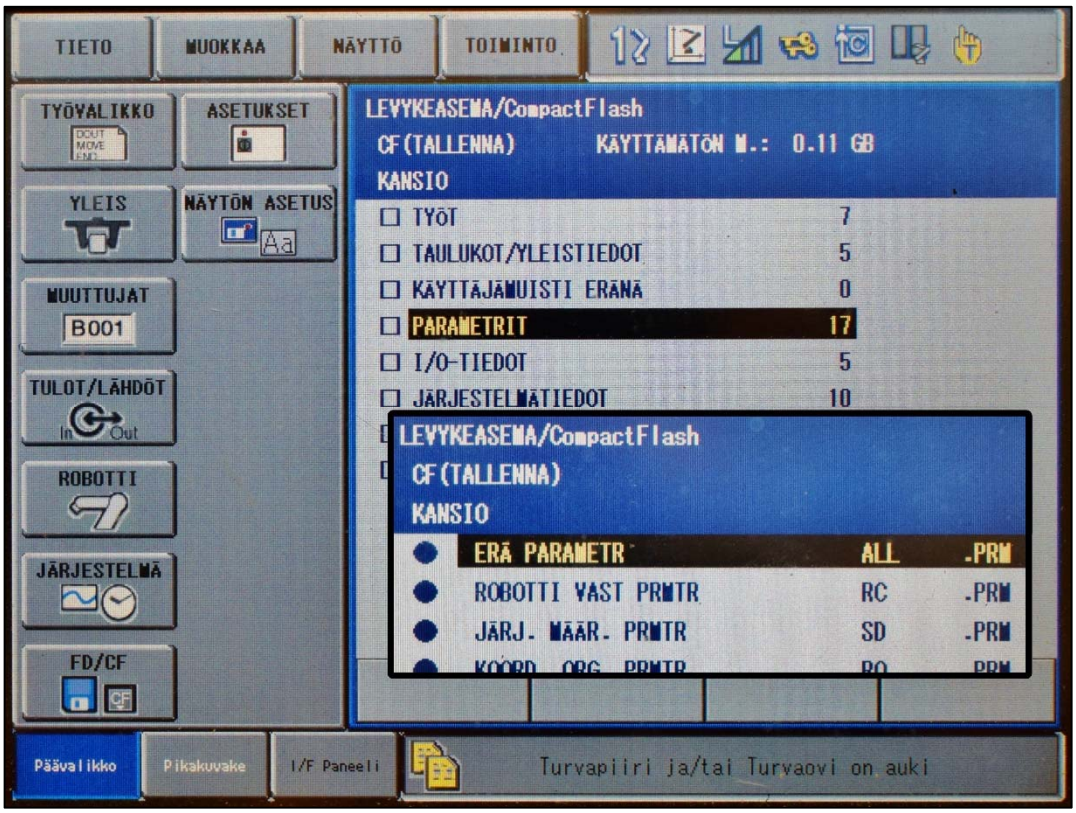
Kuva 76. Työn lataaminen muistikortilta robotille.

Ohjelman pitäisi nyt löytyä robotin työlistasta. Tämän voi tarkistaa valitsemalla **TYÖVALIKKO → VALITSE TYÖ,** jolloin avautuu kuvan 77 mukainen lista, josta löytyy kaikki robotilla olevat ohjelmat.

Kuva 77. Robotin työlista.

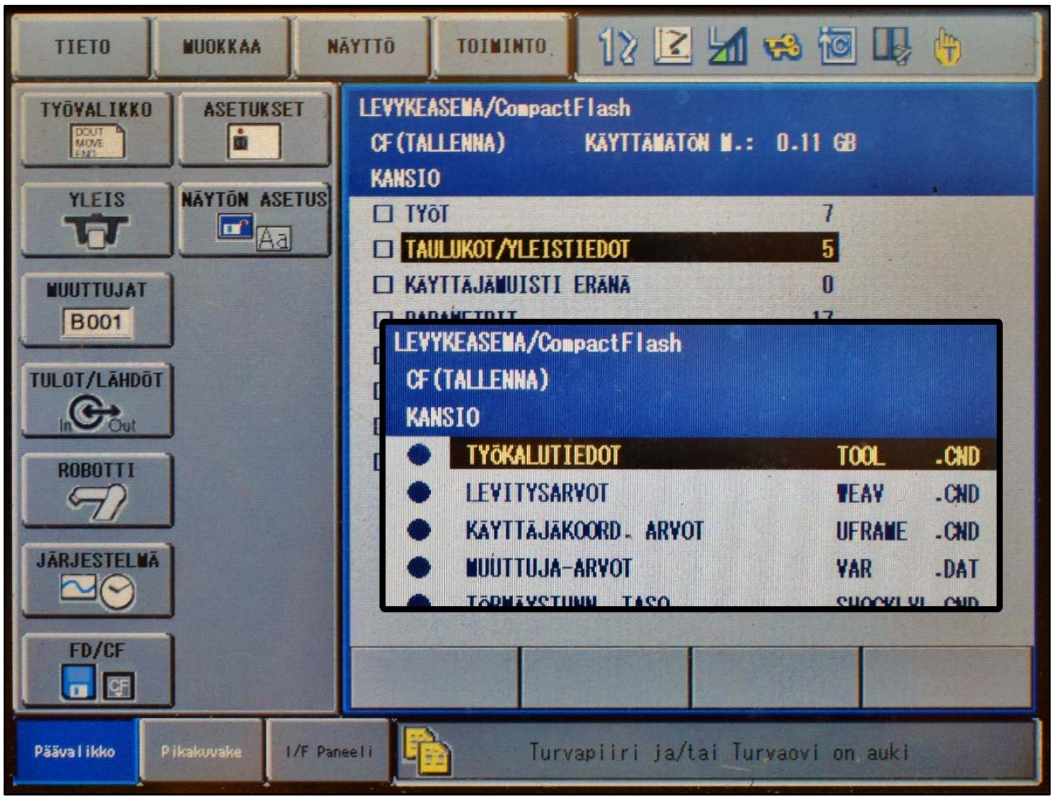
9.3 Robotin asetukset

Jotta robotti liikkuisi ohjelmistossa oikean robotin tavoin, täytyy robotin asetuksiin määrittää joitain robotin ominaistietoja. Tallennetaan muistikortille robotin parametrit valitsemalla **FD/CF → TALLENNA → PARAMETRIT → ERÄ PARAMETR** (**ALL.PRM**), kuten kuvassa 78.



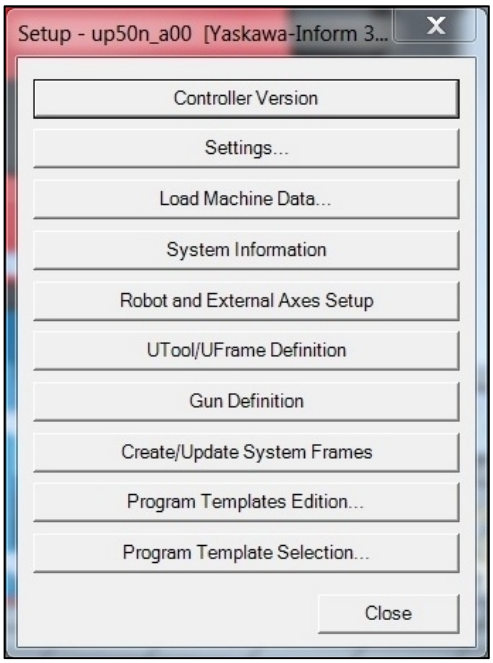
Kuva 78. Robotin parametrien tallentaminen.

Samalla voidaan halutessa tallentaa robotin työkalu- ja käyttäjäkoordinaatistot valitsemalla **TAULUKOT/YLEISTIEDOT → TYÖKALUTIEDOT** (**TOOL.CND**) ja **KÄYTTÄJÄKOORD. ARVOT** (**UFRAME.CND**), kuten kuvassa 79.



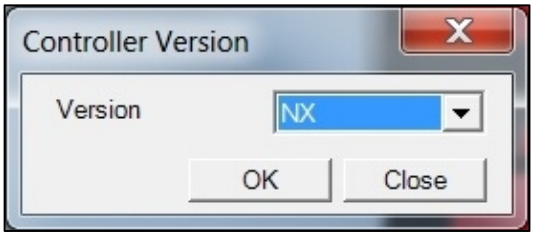
Kuva 79. Robotin työkalu- ja käyttäjäkoordinaatistojen tallentaminen.

Tiedostot viedään tietokoneelle ja ohjelmiston **Robot**-välilehdeltä **Setup**-kohdasta valitaan **Robot Setup**, jolloin avautuu kuvan 80 mukainen valikko. Huomioi, että Setup-valikko riippuu käyttämästäsi ohjaimesta ja saattaa poiketa kuvasta 80.



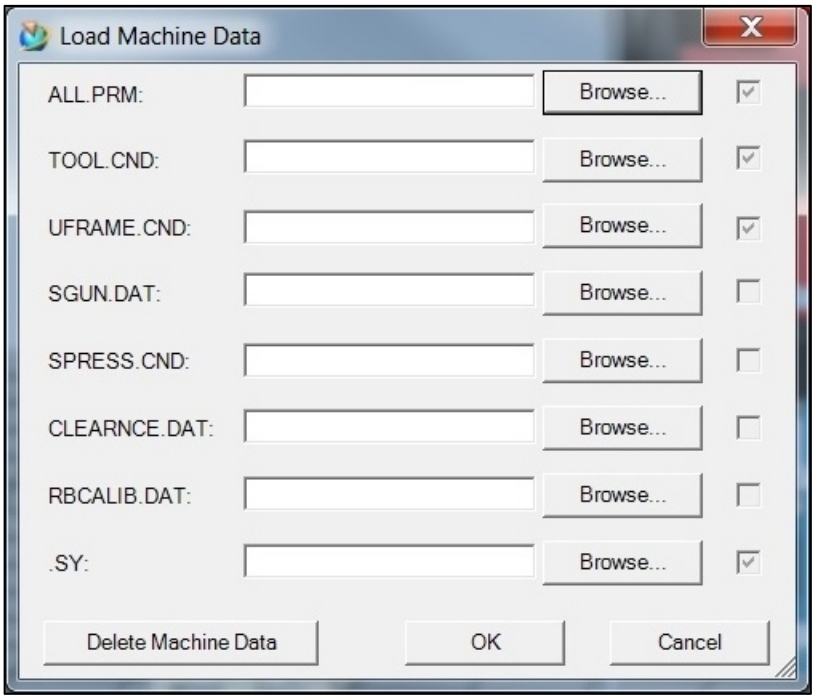
Kuva 80. Robot Setup -valikko.

Controller Version -kohtaan valitaan käytössä oleva ohjain, kuten kuvassa 81.



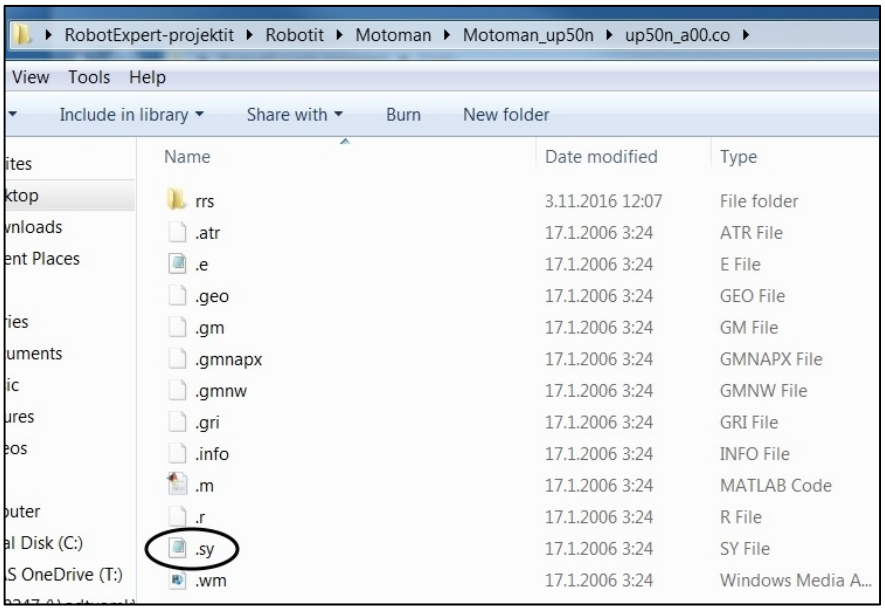
Kuva 81. Ohjaimen valinta.

Load Machine Data -kohdasta avautuu kuvan 82 mukainen ikkuna. **ALL.PRM**-, **TOOL.CND**- ja **UFRAME.CND**-kohtiin etsitään robotilta tuodut vastaavat tiedostot.



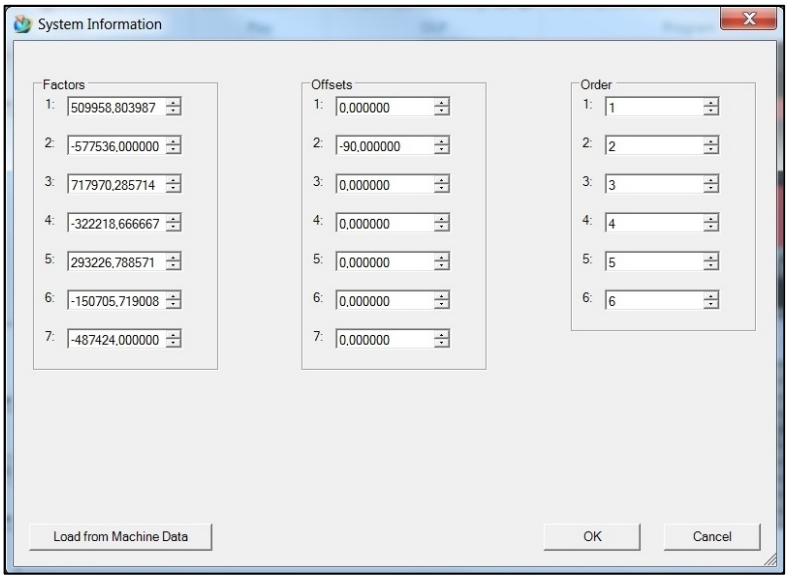
Kuva 82. Robotin ominaistietojen lataaminen.

.SY-tiedosto löytyy robottimallin kansiosta, kuten kuvassa 83.



Kuva 83. .SY-tiedoston lisääminen.

System Information -kohdasta avautuu kuvan 84 mukainen ikkuna. Valitaan **Load from Machine Data**, jolloin järjestelmän tiedot haetaan automaattisesti **Load Machine Data** -kohtaan ladatuista ALL.PRM ja .SY-tiedostoista. Näitä tietoja tarvitaan, jotta robotilta saatavat pulssitiedot voidaan muuttaa vastaaviksi koordinaateiksi RobotExpertissä.



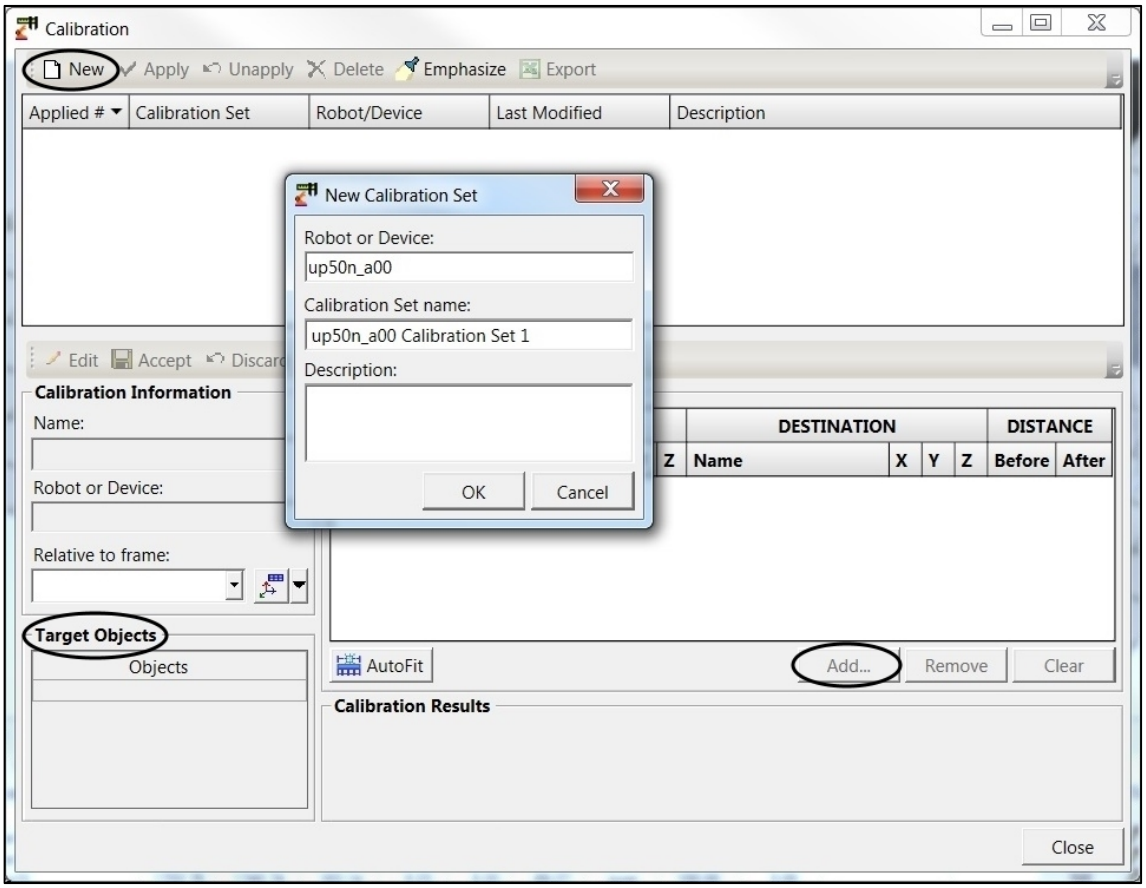
Kuva 84. Järjestelmän tiedot.

Robotin asetukset eivät välttämättä päivity heti, joten tässä vaiheessa solu kannattaa tallentaa ja ohjelmisto käynnistää uudelleen. Nyt robotilta tuotujen ohjelmien liikepisteiden pitäisi asettua simulointimallissa samoihin paikkoihin kuin oikealla robotilla.

10. Kalibrointi

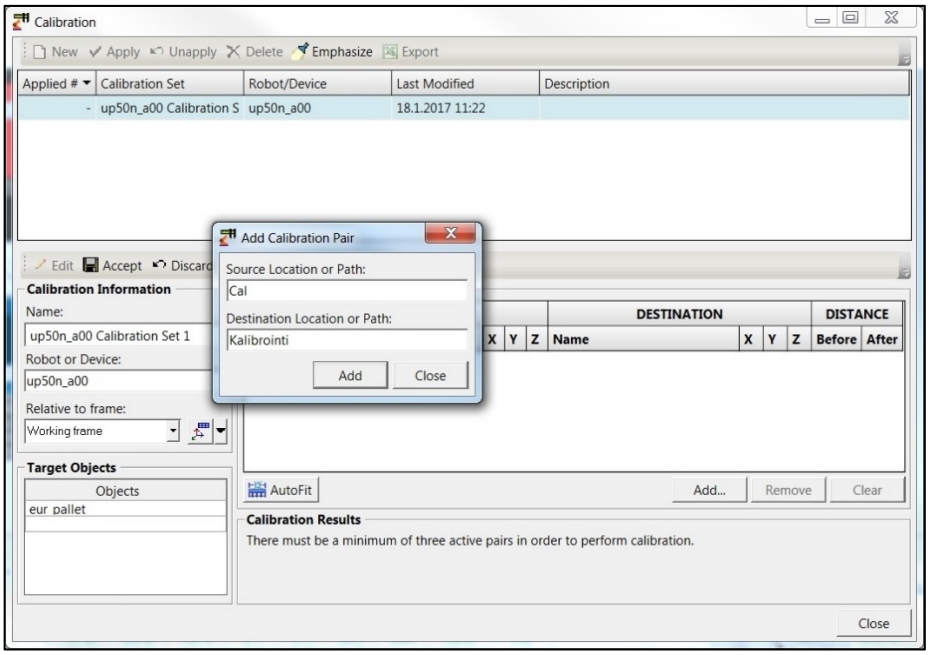
Tässä esimerkissä kalibroidaan solussa olevan EUR-lavan sijainti ja asento. Oikealla robotilla tehtiin ohjelma, jossa robotin työkalupiste käy lavan kulmissa ja vastaava liikerata luotiin RobotExpertillä.

Calibration-välilehti ei näy oletuksena ohjelmistossa. Kalibrointityökalu voidaan lisätä klikkaamalla ohjelmiston välilehtipalkkia hiiren oikealla ja valitsemalla **Customize Ribbon**. Kun välilehti on lisätty, valitaan **Calibration → New**, jolloin avautuu kuvan 85 mukainen ikkuna.



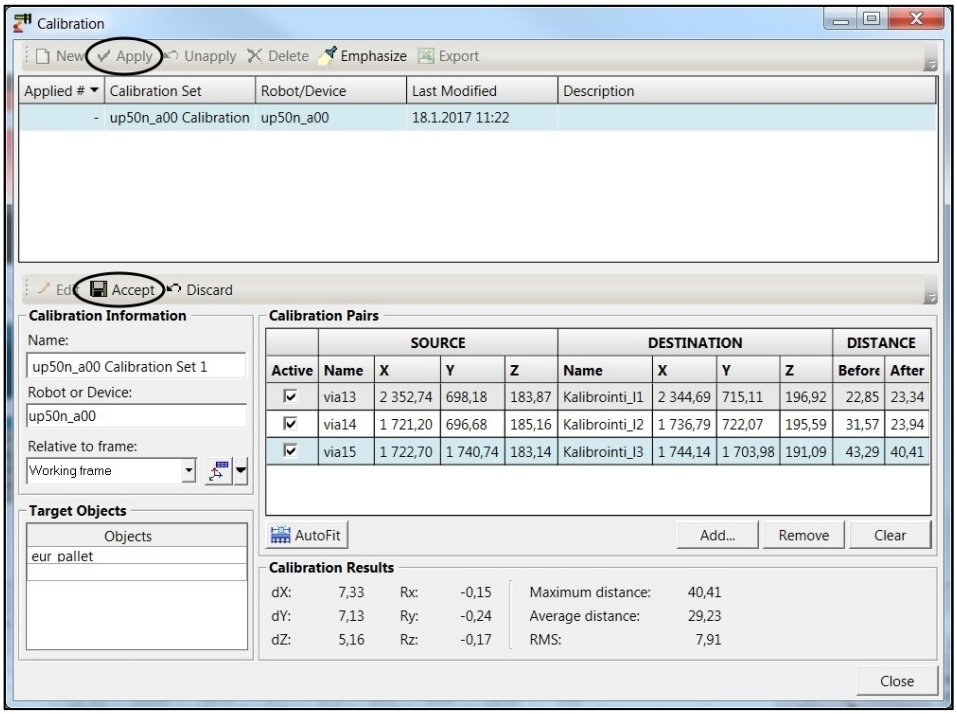
Kuva 85. Uuden kalibrointijoukon luominen.

Robot or Device -kohtaan valitaan robotti tai muu laite ja annetaan uudelle kalibrointijoukolle nimi → **OK**. **Target Objects** -kohtaan valitaan ne komponentit, joiden suhteen transformaatio suoritetaan. Valitaan **Add…** jolloin avautuu kuvan 86 mukainen ikkuna.



Kuva 86. Kalibrointiparien lisääminen.

Source Location or Path -kohtaan valitaan pisteet simulointimallissa ja **Destination Location or Path** -kohtaan robotilta saadut mittauspisteet → **Add** → **Close**. Kalibrointipareja täytyy olla vähintään kolme, jotta kalibrointi onnistuu. **Calibration**-ikkunan alareunassa näkyy kalibroinnin tulos, kuten kuvassa 87. Jos kalibrointiin ollaan tyytyväisiä, valitaan **Accept** ja **Apply**, jolloin **Target Objects** kohtaan määritettyjen komponenttien sijainti ja asento muuttuu kalibrointitulosten mukaan.



Kuva 87. Kalibroinnin tulos.