Tampereen ammattikorkeakoulu
Kone- ja tuotantotekniikka
Opinnäytetyö
Tuukka Antero Heinä

ABB IRB 140 robotin käyttöönotto-opas

Työn ohjaaja Olavi Kopponen
Työn tilaaja TAMK, Seppo Mäkelä
Tampere 11/2010

Tiivistelmä

Tämän opinnäytetyöni tavoitteena on antaa opiskelijoille perustavaa pohjaa robotin ohjelmoimiselle. Robotteja usein pidetään ehkä vähän vaarallisina, ja opiskelijat saattavat niitä hieman välttää niiden arvokkuutensa perusteella. Ohjelmointia pidetään usein haastavana, ja aloittelija harvoin tietää miten itse robotti toimii ja käyttäytyy. Työni on tarkoitettu aloittelijoille, joille työ pyrkii antamaan avaimia perusasioiden ymmärtämiseen ja niiden ohjelmoimiseen.

Työssäni on kaksi erillistä osaa. Ensimmäisessä osassa on teoriaosuus, jossa käydään robotin ohjelmistoja ja rutiineja läpi aina robotin käynnistämisestä asti. Teoriaosiossa kyetään antamaan hyviä ohjeita ohjelmantekoa varten ja kertomaan mitä robotilla ohjelmoiminen on.

Työni toisessa osassa on harjoitustöitä, joita opiskelijat voivat itse harjoitella. Työt eivät ole suoraan teollisuudesta vaan töissä pyritään saamaan opiskelijalle kiteytymään perusasiat mieleen ja saada erilaisia mielikuvia, joista on helppo lähteä kehittämään itseään teollisuuden pariin. Harjoitustyöt lähtevät kehittymään helpoista perustehtävistä haastavimmiksi. Töiden ohella opiskelijan pitää itse miettiä ja ajatella, mitä hyötyä on mistäkin tehtävästä ja millaisessa paikassa vastaavanlaista käskyä voidaan soveltaa. Kaikki harjoitustyöt tehdään opettamalla ohjelmoimalla.

Abstract

The target of my master's thesis is to give students some basic information on programming a robot. People may hesitate to use robots because they think using them might be dangerous and expensive if the robots break up. Students think that programming is a hard task. This is because students do not know enough about robotics and how the robots will act. My master's thesis is meant for beginners and I try to give several good hints on programming and on how to use a robot.

My thesis is divided into two different parts. First, I will explain the theory of robotics. In the theory part, I go through how to start a robot and how to make and use subprograms and main programs. In the theory part, I will give lots of good instructions on programming. The best way to learn to program is to make a program.

Second, I will present a lot of different exercises for students. The exercises will become more difficult step-by-step. These exercises are not taken straight from the industry but I wanted to focus on the basic tasks and elements. When you are able to build a program with basic elements, it will be easy to learn more about programming itself. The students must think new uses for existing programs and also try to adapt these tasks to the industry.

Alkusanat

Opinnäytetyön tekeminen Tampereen ammattikorkeakoululle oli minulle luonnollinen valinta, koska sain koulusta mielenkiintoisen työn. Opinnäytetyön valinta oli minulle henkilökohtaisesti kiinnostava, koska olen aina ollut innostunut robotiikasta. Kiitokset Tampereen ammattikorkeakoulun opettajille, joiden kanssa olen tehnyt yhteistyötä. Asiantuntevaa apua ja vastauksia minun kysymyksiini sain kiitettävästi. Kiitettävää on ollut myös opettajien antama inspiraatio ja tuki. Siitä suuri kiitos Ammattikorkeakoulun opettajille. Suuri kiitos kuuluu myös minun perheelleni, sukulaisilleni ja ystävilleni, jotka ovat auttaneet minua eteenpäin elämässä ja koulunkäynnissä.

2.2.2011

Tuukka Antero Heinä

Sisällysluettelo

Symboliluettelo

AGV	Automated guided vehicle	Vihivaunu, automaattitrukki
”	Tuuman merkintä tyyli	
I/O	In/Out Put	Sisä/ulos tulot
DI	Digital input	Tulo
DO	Digital output	Lähtö
CAD	Computer aided design	Tietokoneavusteista suunnittelua
CNC	Computerized numerical control	Tietokoneistettu numeerinen ohjaus

2. Robotiikka ja robottien rakenteet

Robotti sanalla usein tarkoitetaan mekaanista laitetta tai konetta, joka toimii maailmassa pääsääntöisesti omin avuin. Esimerkiksi teollisuusrobotit usein muistuttavat ihmisen kättä, joka tekee työtä pyyteettömästi. Robottien käyttö on kuitenkin pääsääntöisesti teollisuudessa, mutta se on kokoajan lisääntymässä esimerkiksi terveydenhuollon pariin. Hyvä esimerkki viihdekäytön robotista on Aibo. Usein kuitenkin nähdään robotteja tekevän likaisia, vaarallisia tai liian yksinkertaisia tehtäviä ihmisten tekemäksi. Robotiikka on hioutunut vuosien varrella, siten että robotti voidaan jakaa kuuteen erilaiseen muottiin.

2.1 Robottien rakenteet

Tampereen ammattikorkeakoulussa on kaksi opetuskäytössä olevaa robottia, jotka on suunnattu kone- ja tuotantotekniikan opiskelijoille. Toinen on Motomanin SK 6 kiertyvänivelinen robotti ja toinen on ABB IRB 140- robotti. Keskitymme tässä työssä ainoastaan ABB robottiin, joka on myös kiertyvänivelinen robotti. Muiden valmistajien vastaavat kokoluokan robotteja on Fanuc LR Mate 200ic, Motoman HP5 ja Kukan KR5.

Teollisuusrobottien lisäksi on myös erilaisia erikoisrobotteja, jotka eivät ole kuitenkaan niin yleisessä käytössä kun teollisuusrobotit. Perinteisesti robotiikka on suunnattu tuotantolinjoille, mutta nykyään suurtenikäluokkien kasvaessa robotiikka näkyy kokoajan kasvavan jokapäiväisessä elämässä. Robotiikkaa on alettu hyväksikäyttämään mm. palvelutehtävissä.

Erikoisrobotteja, jotka ovat tietokoneen ohjaamia robotteja kutsutaan mobiiliroboteiksi. 70-luvulla käyttöön tulleet AGV:t ovat hoitaneet mm. palontorjunta- ja pelastustehtäviä.

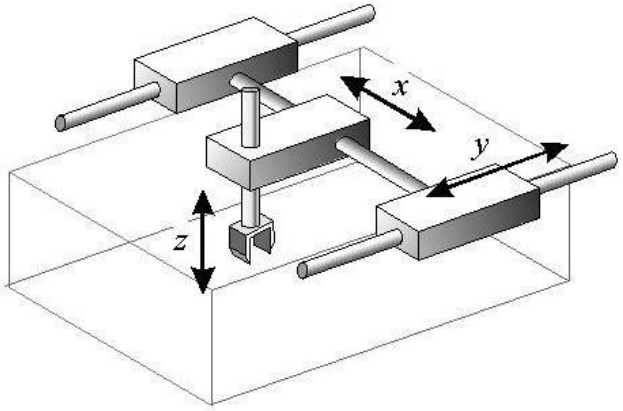
Teollisuudessa olevat robotit luokitellaan erilaisiin luokkiin mekaniikkansa perusteella. Koulun robotit kuuluvat molemmat kiertyvänivelisiin robotteihin. Muita rakenne vaihtoehtoja on:

- Suorakulmainen robotti
  - Sylinterirobotti
- Napakoordinaatistorobotti
  - Scara- robotti
- Kiertyvänivelinen robotti

2.1.1 Suorakulmainen robotti

Suorakulmaisen robotin toimintaperiaatteena on kolme ensimmäistä vapausastetta, jotka ovat lineaarisia. Tyypillisimmät robotit ovat portaalirobotteja. Portaalirobotin rakenne on tuettu palkeilla sen työalueen nurkista.

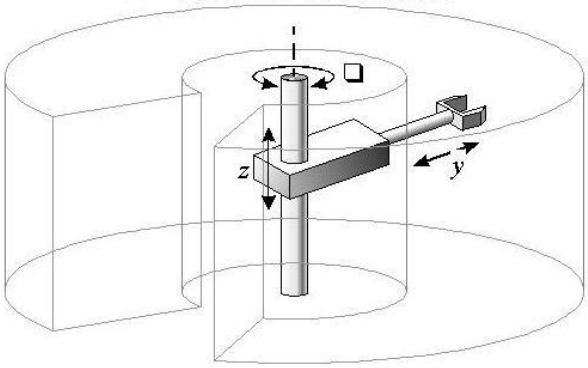
Suorakulmaisen robotin rungon tekevät pystypalkit ja niiden välissä oleva poikittainen palkki. Näin ollen robotilla on kolme vapausastetta. Tavallisimmin näemme suorakulmaisia robotteja erilaisissa logistiikkatehtävissä.



Kuva 1 Suorakulmaisen robotin toimintamalli

2.1.2 Sylinterirobotti

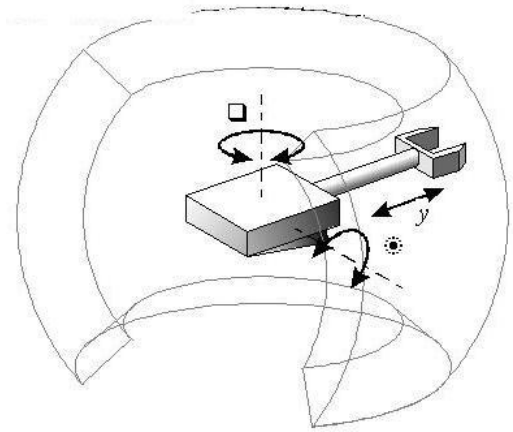
Sylinterirobotin nimi tulee sen koordinaatiojärjestelmästä. Robotissa on yksi pyörivä nivel ja kaksi lineaarisesti liikkuvaa niveltä. Sylinterirobotteja käytetään yleisesti manipulaattorina.



Kuva 2 Sylinterirobotin toimintamalli

2.1.3 Napakoordinaatistorobotti

Napakoordinaatistorobotin työskentelytila voidaan nähdä täysin pyöreäksi vaikka todellisuudessa sen on vaikea päästä joka paikkaan. Kiertyvänivelisenrobotin liikkeet perustuvat napakoordinaatiorobotin liikkeisiin. Napakoordinaatistorobottia käytetään yleisesti pistehitsaukseen, ruiskuvaluun, työstökoneiden panostukseen, kaasu ja kaarihitsaukseen.



Kuva 3 Sylinterirobotin toimintamalli

2.1.4 Scara-robotti

Scara-robotin nimi tulee englanninkielisistä sanoista Selective Compliance Assembly Robot Arm. Scara-robotti on suunniteltu pystysuoraan asennus- ja kokoamistyöhön. Scara robotti muistuttaa vaakatasossa ihmisen kättä, paitsi ranteessa on pystysuoraan liikkuva pystyjohde. Scara-roboteilla on yleisesti vain neljä vapausastetta. Robotin vahvuuksia on sen rungon jäykkyys, nopeus ja tarkkuus. Scara-robotteja käytetään yleisemmin pienien kappaleiden kokoonpano- ja tarkastustöihin. Scara-robotteja näkee mm. matkapuhelimien kokoonpanolinjoissa.

Kuva 4 Scara-robotti

2.1.5 Kiertyvänivelinen robotti

Kiertyväniveliseksi robotiksi lasketaan ne robotit, joissa on vähintään kolme kiertyvää niveltä. Yleisimmissä robottimalleissa on kuitenkin kuusi niveltä. Kiertyvänivelisen robotin toimintaperiaate muistuttaa hyvin pitkälle ihmisen kättä. Nykyiset teollisuusrobottien rakenteet perustuvat lähes aina tähän mekaniikkaratkaisuun sen monipuolisuuden perusteella.

Kiertyvänivelistenrobottien eduksi lasketaan niiden hyvä monipuolisuus ja suurehko ulottuvuus. Huonona puolena voidaan kuitenkin pitää pienehköä kuormankantokykyä.

Kiertyvänivelisiä robotteja nähdään teollisuudessa hitsaus- ja pakkaustehtävissä.

Kuva 5 Kiertyvänivelinen robotti

2.1.6 Rinnakkaisrakenteinen robotti

Rinnakkaisrakenteisenrobotin mekaaninen rakenne muodostuu kolmesta liikeakselista, jotka on asennettu rinnan. Nämä robotit ovat nopeita ja tarkkoja. Huonona puolena rinnakkaisrakenteisissa roboteissa on pienityöalue. Rinnakkaisrakenteisia robotteja näkee teollisuudessa pick&place-työkohteissa, joissa linjastolta otetaan kappale ja viedään toiselle linjastolle. Esimerkkikohteina rinnakkaisrakenteisia robotteja nähdään ruokateollisuudessa.

Kuva 6 Rinnakkaisrakenteinen robotti

2.1 Robotin tekniset tiedot

Käytössäni oli ABB:n IRB 140 robotti, joka on toiseksi pienin valmisteilla oleva ABB teollisuusrobotti. Robotilla on kuusi akselia. Robotti kyetään asentamaan tarpeensa mukaisesti maahan, seinään tai kattoon riippuen sen käyttötarkoituksesta.

Kuva 7 ABB IRB140 robotti

Robotin tyyppi	IRB 140
Ohjausyksikön tyyppi	IRC5
Akselien Lukumäärä	6kpl
Robotin massa	250kg
Kappaleen käsittely paino	6kg
Asemointi tarkkuus	0,03mm
Lämpötila manipulaattorille	5 – 45C°
Ilmankosteus	max.95%
IP	IP67
Äänen taso	max.70dB

Liikeradat	1-akseli	360°
	2-akseli	200°
	3-akseli	280°
	4-akseli	Rajoittamaton, oletuksena 400°
	5-akseli	240°
	6-akseli	Rajoittamaton, oletuksena 800°
Maksiminopeudet	1-akseli	200°/s
	2-akseli	200°/s
	3-akseli	260°/s
	4-akseli	360°/s
	5-akseli	360°/s
	6-akseli	450°/s

2.3. Robotin turvallisuusvaatimukset

Robotin rakenteen avulla täytetään normin ISO 10218 (tammikuu 1992) teollisuusrobotteja koskevia turvamääräykset. Robotti täyttää myös normin ANSI/RIA (15.06-1999) vaatimukset.

Turvatoimintojen/- määräysten määritykset

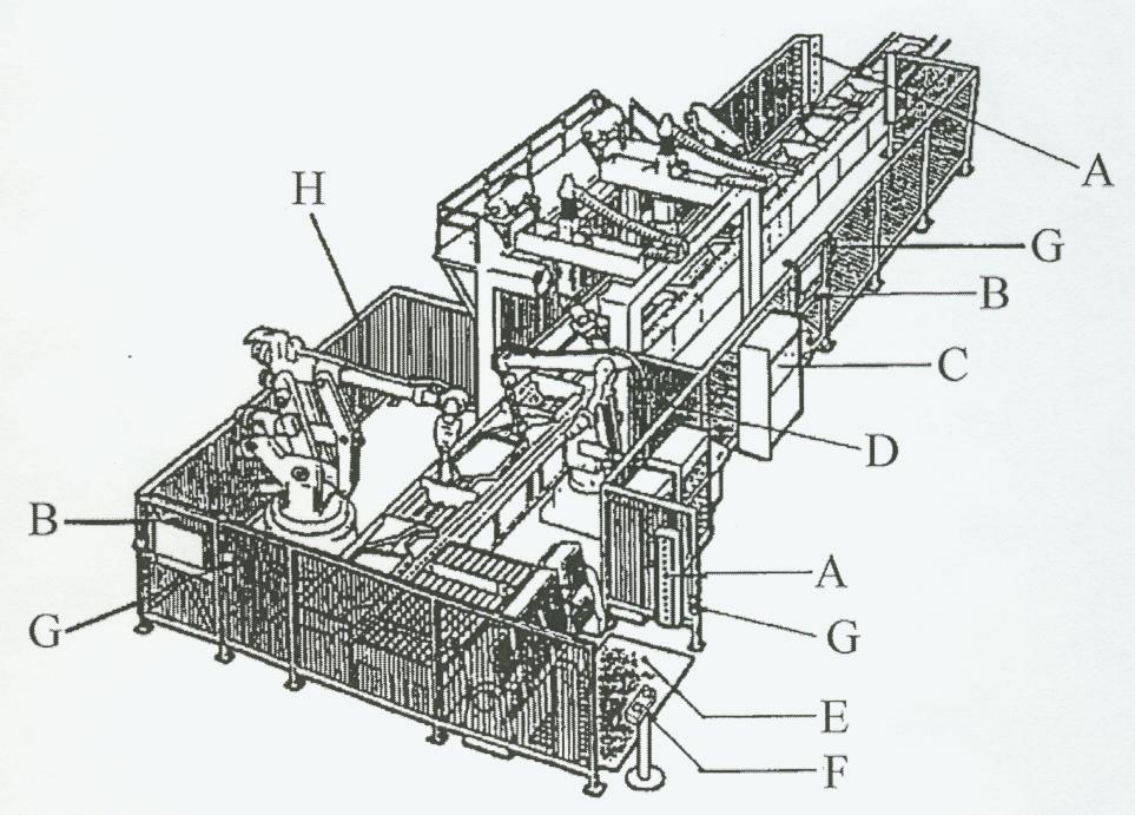
- Hätäseis – IEC 204-1, 10.7
- Sallintakytkin- ISO 11161, 3.4
- Turvalaite ISO 10218 (EN 775), 6.4.3
- Alennettu nopeus –ISO 10218 (EN 775) 3.2.17
- Lukitukset – ISO 10218 (EN 775) 3.2.8
- Pidä – Aja – ISO 10218 (EN 775) 3.2.7

2.3.1 Painettaessa hätäseis

Hätäseis painonapit löytyy ohjauspanelista sekä ohjauskaapista. Mikäli olemme ajautuneet tilanteeseen, jossa hätäseispainiketta on painettu ja tuotanto halutaan ajaa takaisin normaali asetelmiin menetellään täten seuraavasti:

Vaihe	Toimenpide
1	Varmista, että hätäseis - tilanteen aiheuttanut syy on poistunut
2	Paikallista ja nollaa se laite, tai laitteet, jotka aiheuttivat hätäseis – tilanteen
3	Kirjaa hätäseis – tilanne (20202) tapahtumalokiin
4	Paina Motors On (moottorit päälle)- painiketta hätäseis – tilan lopettamiseksi

2.3.2 Esimerkki solun turvallisuudesta



Kuva 8 Esimerkki solun turvallisuudesta

Robottisolun turvallisuus

1 Robottisolun turvallisuudesta huolehditaan erilaisten turvalaitteiden ja suojusten yhdistelmällä.
2 Esimerkiksi kuvan robotisoitu kokoonpanolinja saattaisi sisältää seuraavat turvalaitteet:

A. Valoverho sisäänsyöttöaukossa
B. Rajakytkimet ovissa
C. Ohjauskeskuksessa lukittava virransyötön erotuskytkin
D. Aita erottaa järjestelmän eri osat (ovien rajakytkimet vaikuttavat vain toiseen alueeseen)
E. Mahdolliseen puristumiskohtaan pääsy estetty tuntomaton ja
F. Kaksinkäsinhallintalaitteen avulla
G. Kuittauspainikkeet, joita painamalla varmistetaan, ettei kukaan ole jäänyt vaara-alueelle oven sulkemisen tai valoverhon vapautumisen jälkeen.
H. Alue on eristetty vähintään kahden metrin korkuisella aidalla. Aidan etäisyys työskentelyalueesta tulee olla sellainen, ettei tarttujista mahdollisesti irtoavat kappaleet lennä niiden ulkopuolelle.

3.Hallintalaitteet ja laitteen käynnistäminen

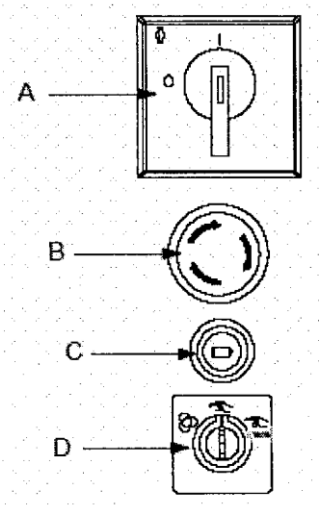
Osiossa käydään läpi robotin hallintalaitteet ja robotin käynnistys.

3.1. Ohjausmoduuli

Ohjausmoduuli sijaitsee robotin alla sijaitsevassa kaapissa. Robotilla on käytössä ABB:n IRC5 ohjausmoduuli. Ohjausmoduuli sisältää mm. prosessorin, joka suorittaa konekielisiä käskyjä, PCI- väylän, CPU-, massamuistia, ja USB- paikkoja.

Käyttäjänä tärkein osa on kuitenkin ohjausyksikkö, jolla voidaan ohjelmoida ohjelmia ja käyttää robottia halutulla tavalla.

3.1.1 Ohjausmoduulin painonapit



Kuva 9 Ohjausmoduulin painonapit

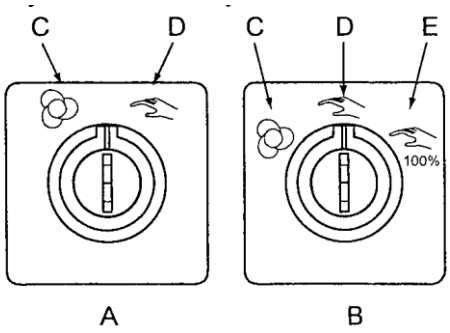
A. Kaksiasentoinen päävirtakytkin
B. Hätäseis - painike
C. Moottorien käyttöpainike
D. Käyttötavan valintakytkin

Hätäseis -painiketta painaessa robotti pysähtyy välittömästi kaikissa mahdollisissa tilanteissa katsomatta työn vaihetta. Hätäseis- painikkeen painonappi jää ala- asentoon ja se on käsin palautettava yläasentoon sen normaaliin asentoon. Painonapin ollessa ala-asennossa robotin servomoottorit eivät käynnisty.

Moottorien käyttöpainike kertoo moottorien tilan. Käyttöpainikkeen merkkivalo välkkyy moottorien ollessa eri tiloissa seuraavasti:

Merkkivalo palaa jatkuvasti	Valmiina ohjelman ajoon
Merkkivalo palaa vilkkuen	Robotti on kalibroimatta, kierroslukijat päivittämättä. Moottorit kytketty päälle.
Merkkivalo palaa hitaasti vilkkuen	Turvalaite aktivoitunut. Moottorit kytketty pois päältä.

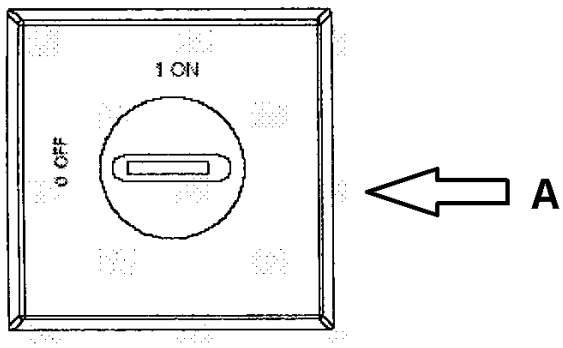
Robotin käyttötavalla tarkoitetaan robotin liikkumisen määrittämistä. Robotilla on eriasennoilla erinäisiä tapoja liikkua.



Kuva 10 Käyttötavan valintakytkimet

A. Kaksiasentoinen valintakytkin
B. Kolmiasentoinen valintakytkin
C. Automaattiajoa käytetään kun halutaan ajaa robottia valmiiden ohjelmien kanssa tuotannossa. Kyseisessä tilassa ei ole mahdollista liikuttaa robottia ohjauspaneelin ohjaussauvalla eli joystikillä.
D. Käsiajoa käytetään ohjelmien luomiseen ja robotin käyttöönoton yhteydessä. Käsiajo tilassa robotilla on rajoitettu nopeus. Kuolleenmiehen kytkin pitää olla aktiivisena kun haluamme liikuttaa robottia.
E. Käsinajo100% käytetään kun haluamme kokeilla ohjelman testaamista täydellä nopeudella. Voidaan myös testata robotin ja käsittelylaitteen välistä synkronointia eli kahden erillisen laitteen yhdessä toimimista.

Käyttömoduulin painike



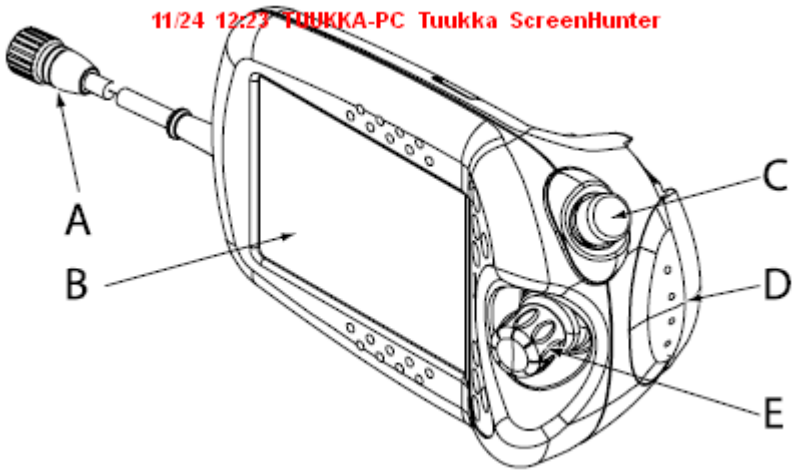
Kuva 11 Käyttömoduulin käyttöönotto kytkin

A. Käyttömoduulin käyttöönottokytkin. Kytkin on kaksiasentoinen päälle/pois-kytkin.

3.1.2 Ohjainyksikön tekniset tiedot ja painonapit

FlexPendant eli ohjausyksikkö on yksi osa IRC5- järjestelmää, joka on kytketty liitännän ja kaapelin avulla järjestelmään. Ohjainyksikkö on käsikäyttöinen käyttölaite, jolla ohjataan ohjelmia, siirretään käsittelijää ja muokataan robottiohjelmia.

- Ohjausyksikön paino – 1kg
- Näyttö – 7.7” värinäytöllinen kosketusnäyttö.



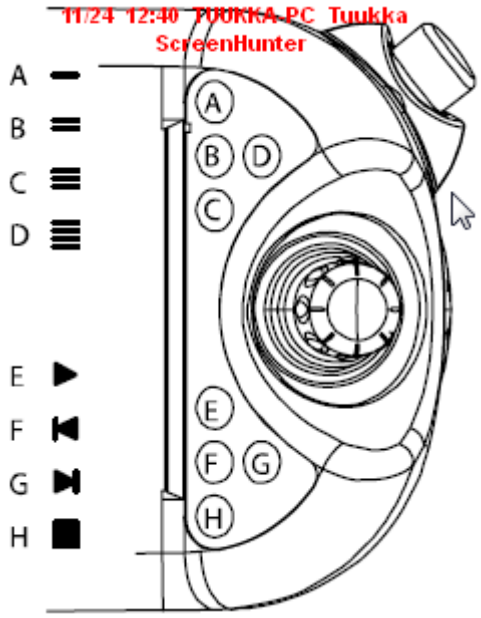
Kuva 12 Ohjainyksikkö

A. Liitin, jolla kaapeli kytketään ohjainyksikköön
B. Värinäytöllinen kosketusnäyttö
C. Hätäseis-painonappi
D. Kolmiasentoinen kuolleenmiehen kytkin
E. JoyStick

Ohjainyksikköä käyttäessä ja halutessa liikuttaa robottia on otettava huomioon robotin kuolleenmiehen kytkin. Kuolleenmiehen kytkin on kytkin, jota painattessa robotti liikkuu. Tämä on merkittävä tekijä robotin turvallisuudessa, koska se havaitsee ohjelmoijan läsnäolon. Ohjelmoijan pitää painaa painonappia halutessa liikuttaa robottia. Kytkimellä on näin kolme asentoa:

- Vapaana olo – Robotti pysähtyy /ei liiku
- Kevyt puristus – Robotti toimii ja on kykeneväinen liikkumaan. Servomoottorit ovat toiminnassa.
- Kova puristus – Katkaisee servomoottorien toiminnan ja robotti pysähtyy.

Ohjelmointiyksikössä on painonappeja helpottamaan ohjelmoijan ohjelmoimista.



Kuva 13 Ohjainyksikön painonapit

A – D Painonapit on itse määriteltävissä.

E – Painonappi. Käynnistyspainike. Toteuttaa ohjelman käynnistämisen.

F – Painonappi. Askellus taaksepäin painike. Ottaa ohjelmassa yhden askeleen taaksepäin.

G – Painonappi. Askellus eteenpäin painike. Ottaa ohjelmassa yhden askeleen eteenpäin.

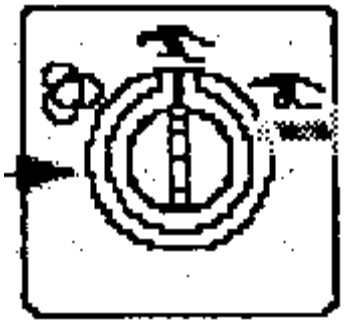
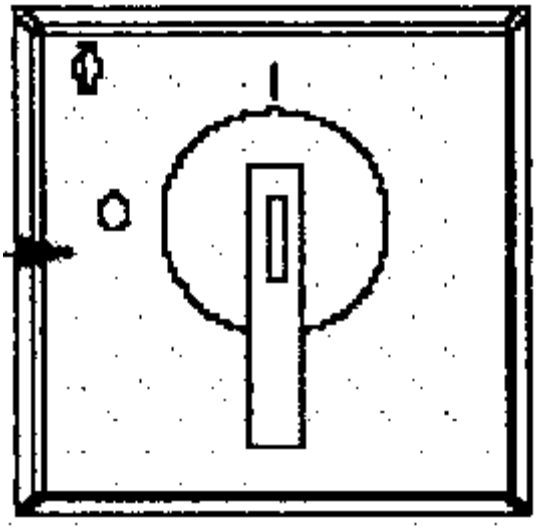
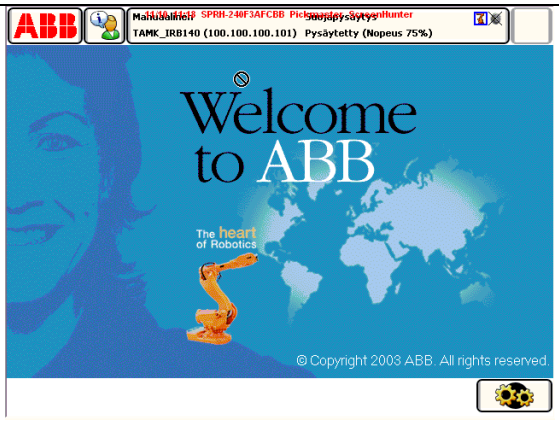
H – Painonappi. Pysäytyspainike. Pysäyttää ohjelman suorittamisen.

4. Robotilla ohjelmointi

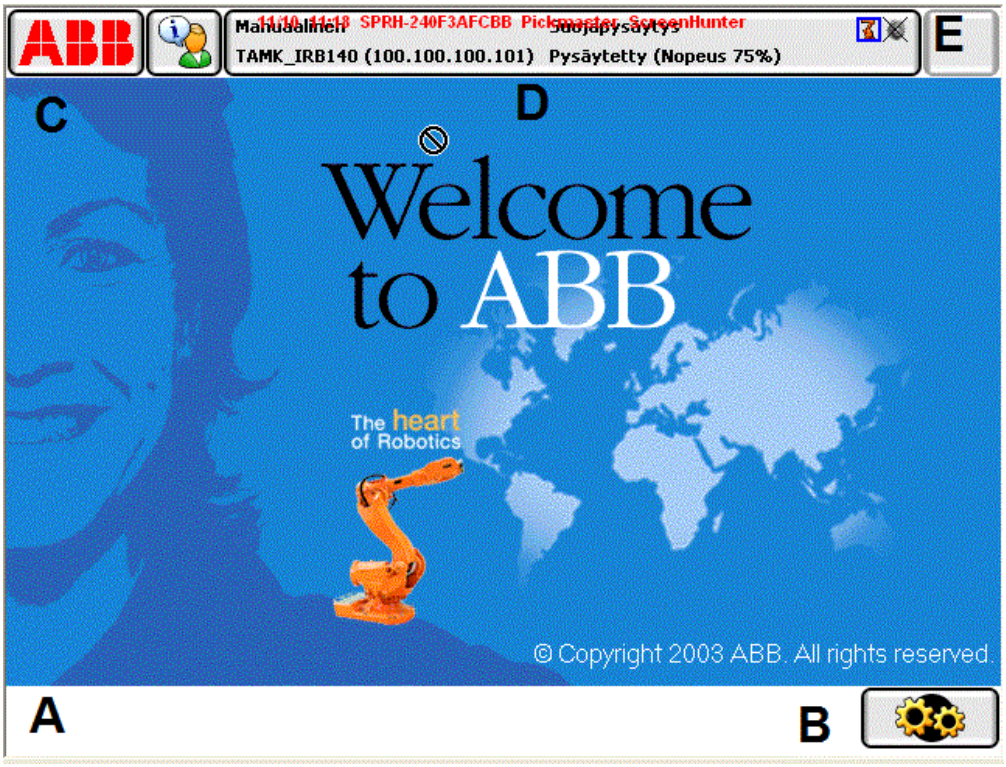
4.1. Robotin päälle kytkeminen

Robotti tarvitsee toimiakseen verkkovirtaa ja paineilmaa. Paineilmaa on otettu luokan nurkasta olevasta paineilmanottopisteestä. Ensimmäisenä otettaessa robottisolua käyttöön on tarkistettava robotin työalue sekä katsottava työturvallisuusasiat kuntoon ennen kuin työt voidaan aloittaa robotilla. Tämän jälkeen voidaan kytkeä paineilma ja sähköt päälle.

4.1.1. Robotin käynnistäminen

Vaihe1	Asetetaan käyttötavan valintakytkimestä haluttu asento. Ohjelmaa tehdessä asetetaan kytkin käsiajokohtaan.	
Vaihe2	Sähköt laitetaan päälle kääntämällä päävirtakytkimestä virrat päälle.	
Vaihe3	Käynnistyksen jälkeen odotetaan kunnes robotin ohjausyksikkö asettuu aloitusvalikkoon.	

Aloitussivulla on pikapainikkeita kaikkiin ohjausyksikön toimintoihin.

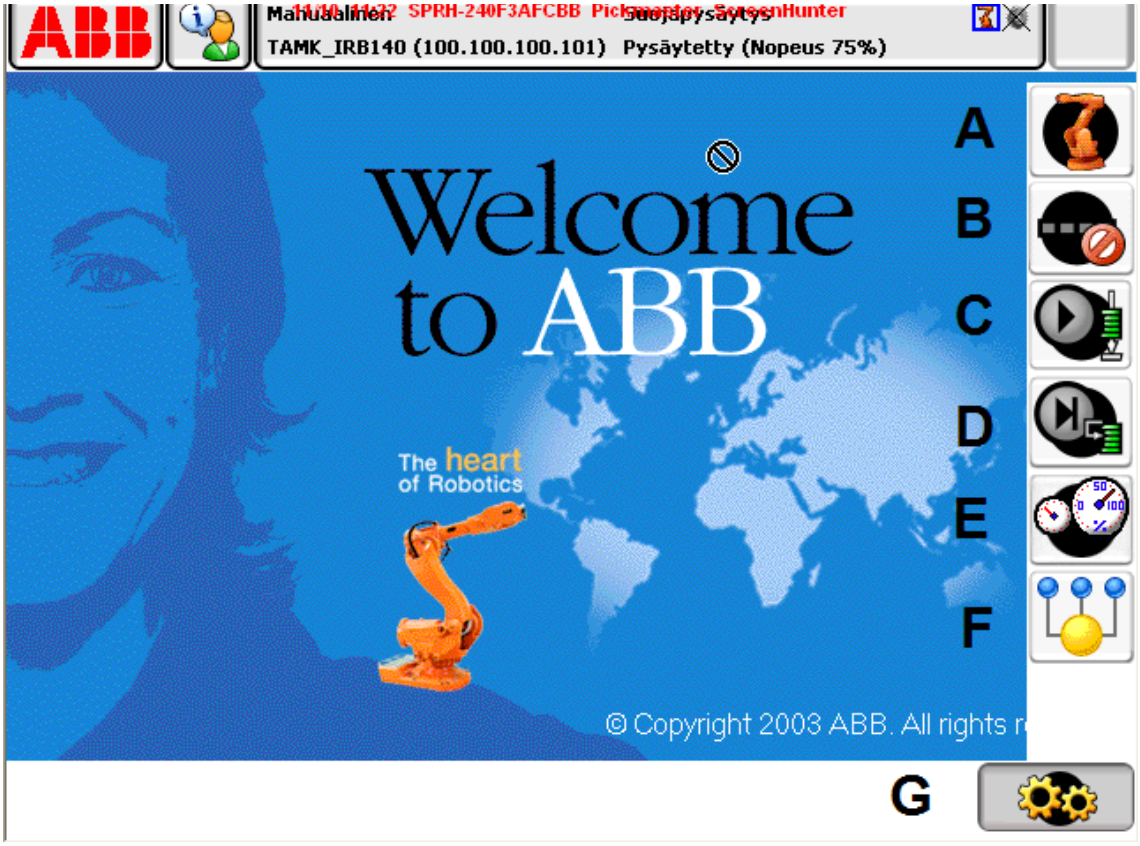


Kuva 14 Ohjainyksikön perusnäyttönäkymä

A. Tehtävärivi, jossa näkee kaikki sivut, jotka ovat auki. Sivuja voidaan vaihtaa napauttamalla haluttua sivua.
B. QuickSet-valikko nopeuttaa robotin ohjelmointia. Sisältää tärkeitä elementtejä robotin ohjelmoimiseen.
C. ABB-valikosta käynnistetään eri toimintoja robotille. Valikosta löytyy mm. ohjelmaeditori ja käsinajovalikko.
D. Tilariviltä näet kokoajan robotin tilan. Tilasta näkyy mm. onko robotin moottorit päällä vai ei.
E. Sivun sulkemispainike. Sulkee aktiivisena olevan ikkunan.

4.2. QuickSet-valikko

QuickSet -valikko nopeuttaa robotin ohjelmointia ja liikuttamista. QuickSet-valikko sisältää paljon hyödyllisiä elementtejä.



Kuva 15 Ohjainyksikön quickset-valikko avattuna.

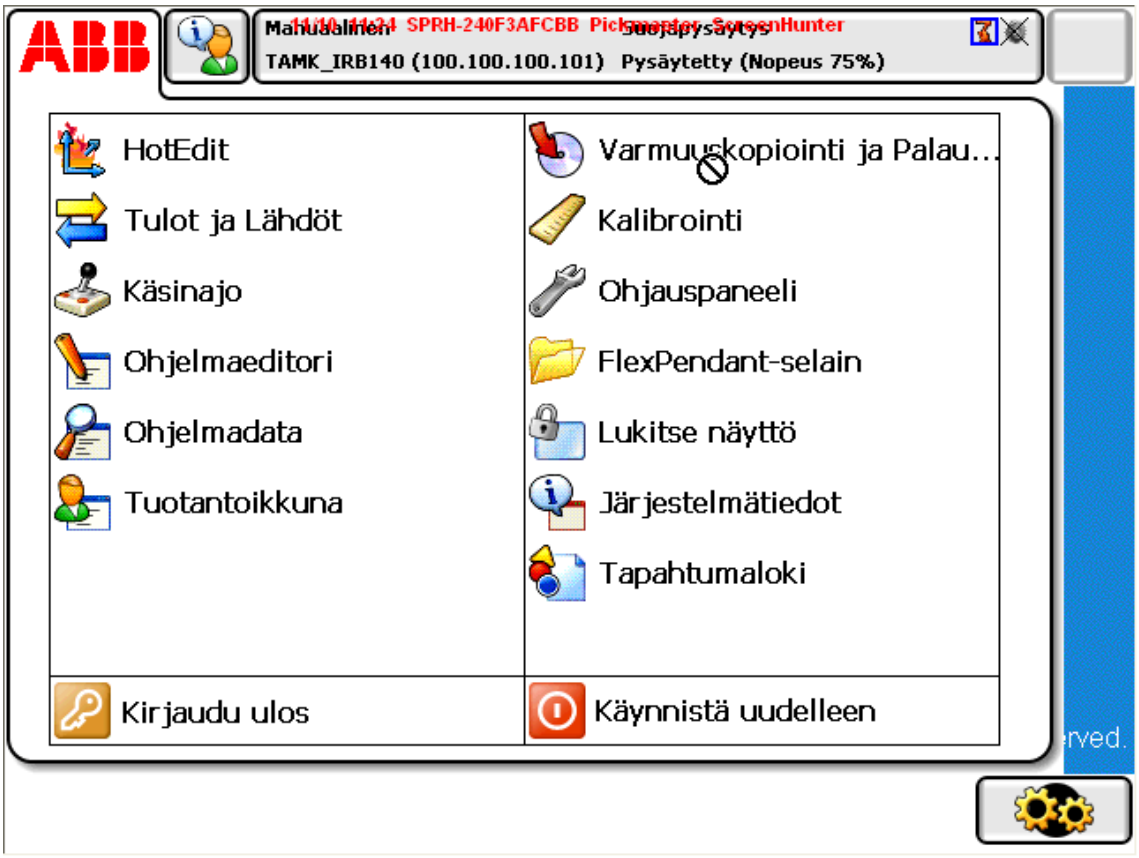
A. Mekaaninen yksikkö. Valikko, josta voidaan valita solusta haluttu mekaaninen yksikkö, liiketavat, työkaluasetukset, työkohdeasetukset ja koordinaatioasetukset.
B. Inkrementti on hyödyllinen liiketyyppi, jolla kyetään tekemään liike pienin askelluksin. Inkrementtiä käytetään yleisesti robotin paikantamiseen. Inkrementti toimii kun ohjaussauvaa käännetään haluttuun suuntaan robotti ottaa yhden askeleen eli inkrementin. Inkrementtejä on viisi erilasita. Koulun robottisolussa ei ole mahdollista käyttää käyttäjän määrittämää inkrementtiä.

Askelkoko	Matka	Kulmaliike
Ei mitään	-	-
Pieni	0,05mm	0,005°
Keskisuuri	1mm	0,02°
Iso	5mm	0,2°
Käyttäjän määrittämä	0,5 – 10,0mm	akselit: 0,01 – 0,20° uudell. orient.: 0,03 – 0,5°

C. Ajotilavalikko. Ohjelmistossa käytetään termiä suoritustila. Tässä tilassa valikoidaan robotin liike. Robotilla on joko käytössä yksittäinen tai jatkuva ajo. Valittaessa yksittäinen ajo, robotti toteuttaa ohjelman kerran kun vastaavasti jatkuvassa ajossa robotti toistaa annettua ohjelmaa toistuvasti.
D. Askellustila. Askellustilalla voidaan tehdä haluttu harppaus ohjelmassa.
E. Nopeus valikko josta saadaan valittua robotti ohjelmalle haluttu nopeus.
F. Tehtävien pysäytys ja käynnistys valikko. Valikosta voidaan pysäyttää ja käynnistää tehtyjä taskeja.
G. QuickSet-valikko avaa yllä olevat toiminnot.

4.3 ABB-valikko

ABB- valikosta löytyy erilaisia toimintosarakkeita. Kullakin toiminnolla on oma käskyrivinsä. ABB-valikosta käydään läpi seikkoja, jotka ovat tarpeellisia ja eniten käytössä olevia toimintoja.

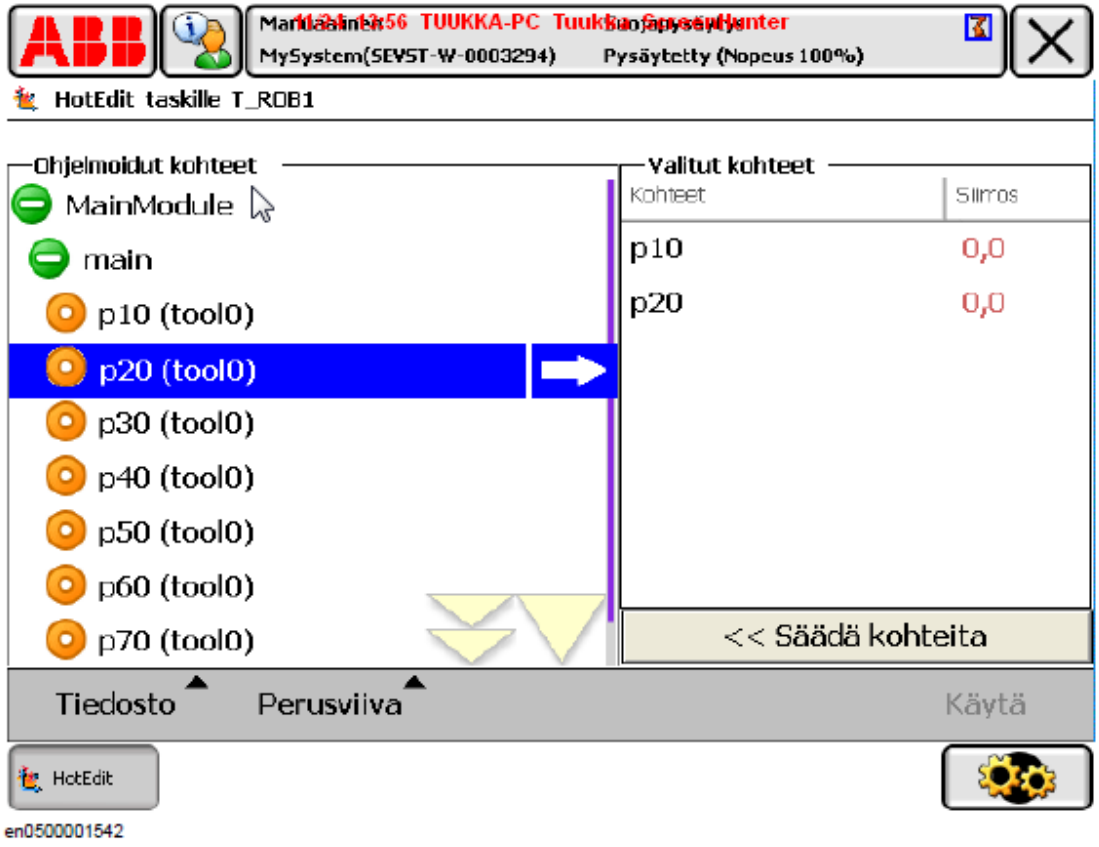


Kuva 16 Ohjainyksikön ABB-valikko avattuna

- Tulo ja lähtö valikosta löytyy robottisolun I/O tulot.
- Käsinajovalikosta löytyy sivu, jolla kyetään ajamaan robottia halutuilla liiketyypeillä ja koordinaatistoilla.
- Ohjelmaeditorissa kyetään tekemään robotille haluttuja ohjelmia.
- HotEdit valikkoa käytetään silloin kun halutaan hienosäätää ohjelmoitujen asentojen paikkaa. Tätä voidaan tehdä robotin ollessa ajossa. HotEdit-valikko on usein rajoitettu käyttäjiltä.

4.3.1HotEdit-valikko

HotEdit- valikko on valikko jossa kyetään tekemään hienosäätöä ohjelmille. Robotti voi olla hienosäädön aikana tuotannossa.



Kuva 17 HotEdit- valikko avattuna ABB-valikosta

Valikko listaa kaikki määrätyt pisteet, jotka ohjelmoija on tehnyt avaruuteen. Lista tulee näytölle puunäkymälistana. Listasta voidaan poistaa pisteitä matkan varrelta viemällä pisteen roskakoriin. Mikäli ohjelmassa on tehtyjä pisteitä, niitä voidaan ladata ja tallentaa uudestaan ohjelmaan.

HotEdit- valikkoa sisältää edistyneitä toimintoja ja käskyjä, joita pitää käyttää harkiten ja huolellisesti.

4.3.2. I/O Sisään- ja ulostulot

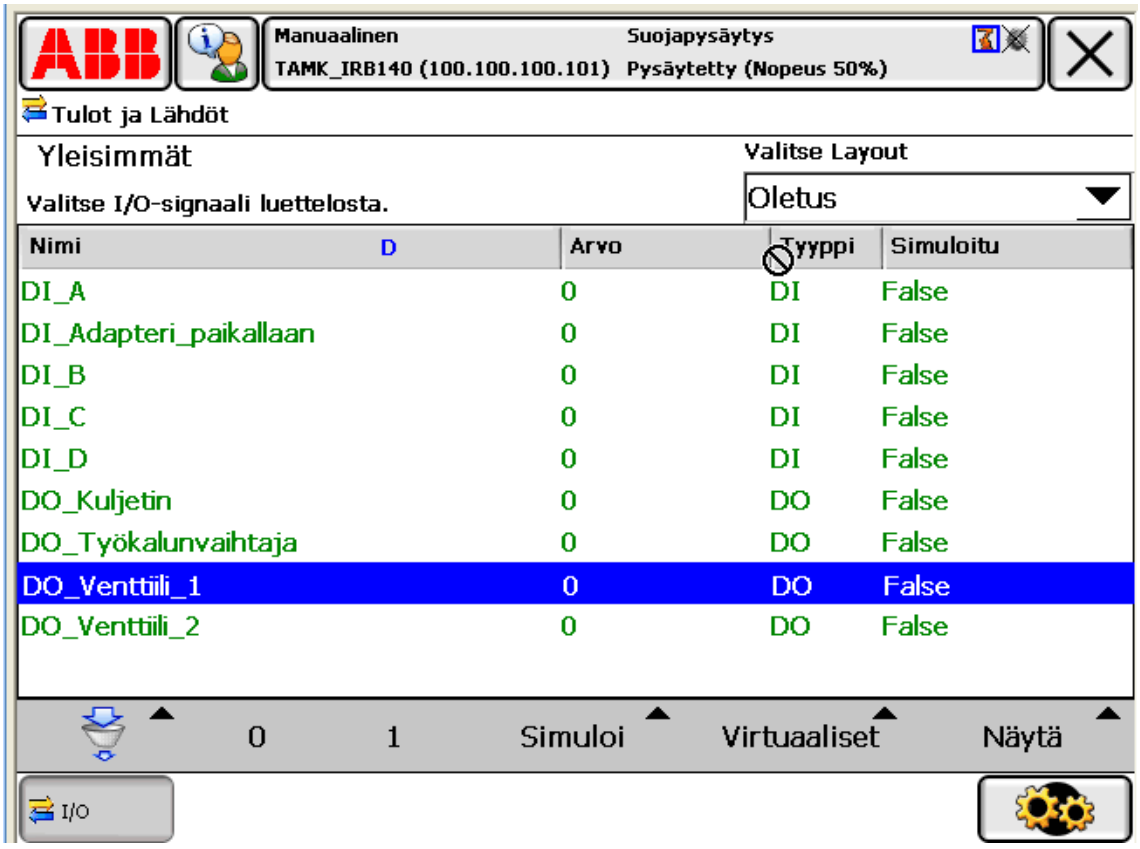
Tulo- ja lähtösignaaleilla tuetaan robotin ohjausta ja käskytetään sen oheislaitteita.

Robottijärjestelmässä on paljon robottia tukevia digitaalisia input eli DI tuloja mm. anturointi, valoverho ja kaikki kytkimet.

Robottijärjestelmän DO eli lähtöjä on mm. työkalujen toiminnot (Tarttuja kiinni/auki), käsittelylaitteiden ohjaukset.

Arvo (bitti)	Jännite (V)	Tila
0	0	Pois
1	24	Päällä

Tulo- ja lähtösignaalit ovat listattu omaan ikkunaan joka on sijoitettu ABB-valikon alle. Tulo- ja lähtösignaalit pitävät nimetä tulo-/lähtösignaalin tehtävän kuvaavalla tavalla.



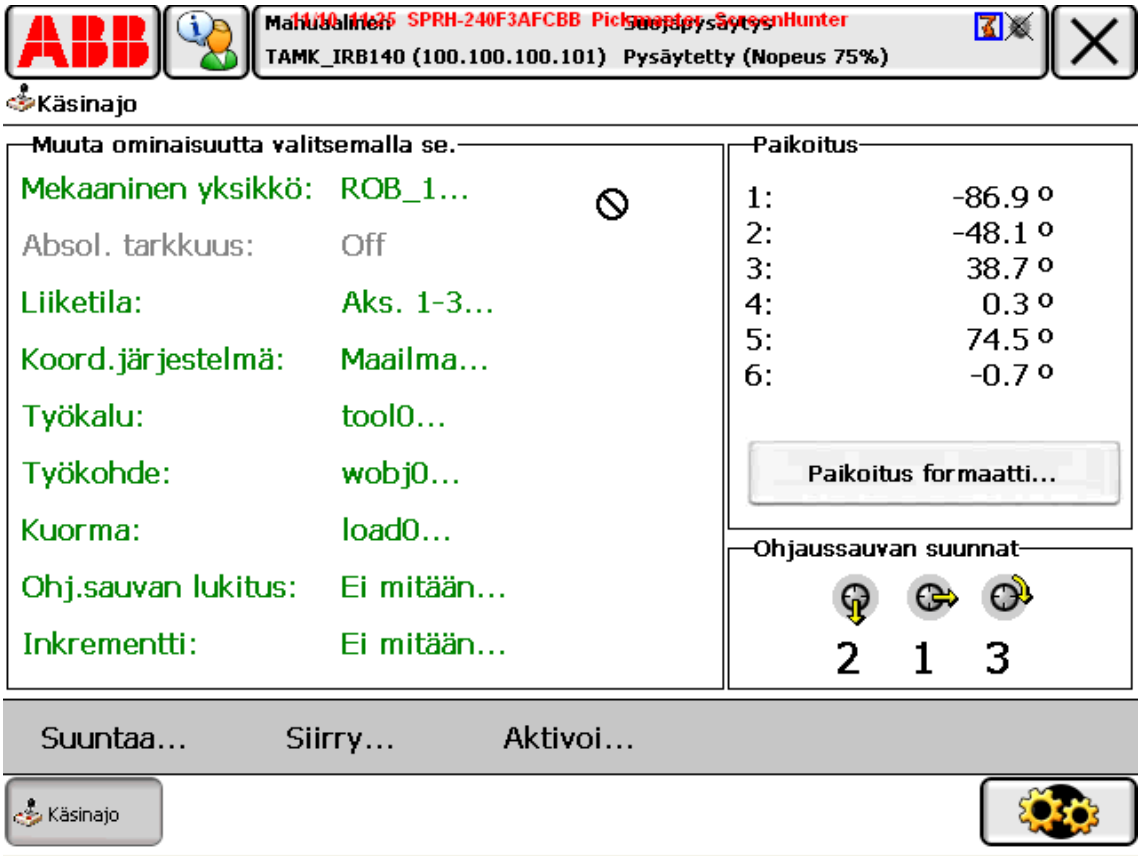
Kuva 18 I/O-valikko, josta nähdään robotin I/O-tulot

4.3.3 Käsinajovalikko

Käsiajovalikosta löytyy toiminnot, jolla kyetään siirtämään ja liikuttamaan robottia halutulla tavalla. Useimmin käytetyt toiminnot löytyvät myös quickset-valikosta.

- Mekaaninen yksikkö: Käyttöjärjestelmässä voi olla kytkettynä useampi robotti kuin vain yksi. Valikosta siis valikoidaan juuri haluttu robotti, jota halutaan hallinnoida. Jos käyttöjärjestelmässä on useampi robotti näkyy ne symboleina tilarivillä, josta valitaan haluttu robotti käyttöön. Valinta tapahtuu maalaamalla ohjauspaneelin näytöstä haluttu robotti ja painamalla se siniseksi ja hyväksymällä se ok-painikkeella.
- Absol. Tarkkuus: Oletus arvona on Ei käytössä. Mikäli robotissa on käytössä Absolute Accuracy optio näytössä lukee tarkkuus käytössä. Tarkoittaa robotin tarkkuutta pisteissä.
- Liiketilavalikko: Valikosta valikoidaan haluttu robotin liiketila. Valikosta löytyy neljä erilaista liiketilaa.
- Koord.järjestelmävalikko: Valikosta valitaan haluttu robotin koordinaatiojärjestelmä. Koordinaatiojärjestelmää käytetään siten että ensin pitää valita robotille liiketila ja tämän jälkeen vasta koordinaatio.
- Työkalu: Nähdään haluttu työkalu.
- Työkohde: Valitaan haluttu työkohde.

- Ohj. sauvan lukitus: Ohjainsauva voidaan lukita tietyissä suunnissa, jolloin yhden tai useamman akselin liikkeet eivät ole mahdollisia. Tämä on hyödyksi kun halutaan tehdä asentojen hienosäätöä. Lukituksen kanssa tehdessä hienosäätöä on huomattava, että liiketilaa muuttaessa eri akselit ovat aina lukittuina.
- Inkrementti: Tehdäänkö liikkeitä inkrementein.



Kuva 19 Robotin käsiajotila

4.3.3.1 Liiketilavalikko

Liiketilavalikosta valikoidaan robotin haluttu liikkumismuoto. Liiketilavalikkoon mennessä saadaan neljä eri liiketilaa.

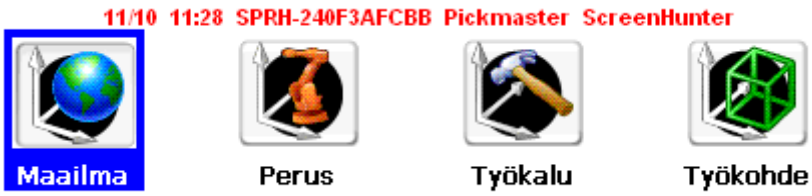
Kuva 20 Robotin liiketilamuodot joiden mukaan robotti liikkuu

- Aks. 1-3 liiketila tarkoittaa robotin akseleita 1-3. Kun Aks. 1-3 liiketila on valittuna robotti ei liikuta muita akseleita.
- Aks. 4-6 liiketila tarkoittaa robotin akseleita 4-6. Kun Aks. 4-6 liiketila on valittuna robotti ei liikuta muita akseleita.
- Lineaar. liikekäsky tarkoittaa robotin lineaarista liikettä. Robotti liikkuu lineaarisesti halutulla koordinaatiojärjestelmän mukaisesti.

- UudOrient. liikekäsky tarkoittaa työkalun liikettä. Koordinaatistosta valitulla koordinaatistolla työkalu liikkuu haluttuun suuntaan.

4.3.3.2 Koordinaatisto

Koordinaatistojärjestelmästä valitaan robotille haluttu koordinaatisto jota robotti noudattaa. Robotti liikkuu eritavalla halutulla koordinaatistolla.



Kuva 21 Koordinaatistoja joiden mukaan robottia kyetään liikjuttamaan

- Maailmakoordinaatisto on ulkopuolinen koordinaatisto jossa robotti työskentelee.
- Peruskoordinaatisto on robotin jalustaan sijoitettu koordinaatisto.
- Työkalukoordinaatisto on työkaluun sijoitettu koordinaatisto.
- Työkohdekoordinaatisto on työkohteelle sovellettu koordinaatisto

4.4 Ohjelmointi

Robottien ohjelmointi alkoi sähkömekaanisista kytkennöistä, joiden avulla saatiin robottien nivelet ajettua haluttuihin asemiin. Tämän jälkeen tuli ”nauhoittamalla” ohjelmoiminen, jossa käytettiin hyväksi robotin paikka-antureita. Nämä kyseiset tekniikat ovat kyllä jo poistuneet markkinoilta, ja nykyisin robotteja ohjelmoidaan pääosin seuraavilla menetelmillä.

- Johdattamalla ohjelmointi
- Opettamalla ohjelmointi
- Etäohjelmointi eli off-line ohjelmointi

Ohjelmoinnin tarkoituksena on saada robotin työkalu haluttuun pisteeseen tekemään haluttua työtä. Tämä työ voi olla hitsausta, kappaleen käsittelyä tai tarkistamista yms. Työtehtävä riippuu täysin robotissa olevasta työkalusta ja itse robotin rakenteesta. Robotti pitää saada ohjelmoitua siten, että robotti solun ympärillä olevat muut järjestelmät toimivat yhteen robotin kanssa. Robottia ohjelmoitaessa on otettava aina seuraavat seikat huomioon:

- Toimintajärjestys ja logiikka robottikäsivarren työkalu käyttöjärjestys.
- Tahdistetaan robotin liikkuminen ympäristön signaaleilla tai välitetään itse robotista signaaleilla muihin ulkopuolisiin elementteihin.
- Määritys robotin toiminnasta virhetilanteessa.

4.4.1. Johdattamalla ohjelmointi

Johdattamalla ohjelmoiminen on nykyään harvinaista sen hankaluuden takia. Johdattamalla ohjelmointi tapahtuu robotin vapautunutta käsivartta liikuttamalla lihasvoimalla, siten että robotille saadaan haluttu liikerata. Nivelanturin tiedot ajetaan instumenttinauhurin kautta robotin toimi-

laitteelle. Tämän ansiosta robotti kykenee pitämään halutun liikeradan. Johdattamalla ohjelmoimisella on paljon huonoja puolia.

- Ohjelmaa on vaikea muuttaa, yleensä ohjelma joudutaan ohjelmoimaan alusta lähtien uudestaan aina kun siihen halutaan tehdä muutos
- Magneettinauhoja on hankala arkistoida ja käsitellä. Materiaali on ohutta rautaoksilla päällystettyä muovinauhaa.
- Ohjelmasta on vaikeaa saada tarkka.

4.4.2. Opettamalla ohjelmointi

Opettamalla ohjelmoiminen ja etäohjelmoiminen on pääsääntöisesti käytössä olevat ohjelmointitavat tämän päivän teollisuudessa. Etäohjelmointi on kokoajan valtaamassa ohjelmointityyliä kokonaan haltuun.

Opettamalla ohjelmoinnin perusteena on opettaa robotille piste pisteeltä haluttu liikerata. Robotin ohjaaminen tapahtuu ohjelmointiyksikön avulla käyttämällä koordinaatistoja ja liikemenetelmiä. Robotille siis opetetaan avaruuteen jokainen haluttu piste missä käyttäjä haluaa robotin kulkevan. Jokaiselle pisteelle opetetaan myös miten robotti saapuu kyseiseen pisteeseen, ja mitä robotin pitää tehdä halutussa pisteessä.

4.4.3 Etäohjelmointi

Etäohjelmointi tarkoittaa robotin ohjelmointia ilman fyysisesti vieressä olevaa robottia. Tätä ohjelmointi tekniikkaa voidaan tehdä vaikka toisella puolella maapalloa robottiin nähden. Tämä ei siis sido ohjelmoijaa robottiin.

Ohjelmointi tapahtuu tietokoneen avulla käyttäen etäohjelmointi ohjelmaa. Tietokoneella luodaan maailma, joka vastaa oikean robotin maailmaa. Tietokone ohjelmissa käytetään 3D-graafista suunnittelua ja erinäisiä simulointimalleja.

Etäohjelmoinnin hyvänä puolena voidaan pitää sen tuotannon nopeutta. Etäohjelmoimalla tuotannon ei tarvitse pysäytettynä kuin hetken, koska sen ohjelma on tehty jo etukäteen tietokoneella. Kun taas opettamalla ohjelmoimalla tuotantolinja pitää pysäyttää.

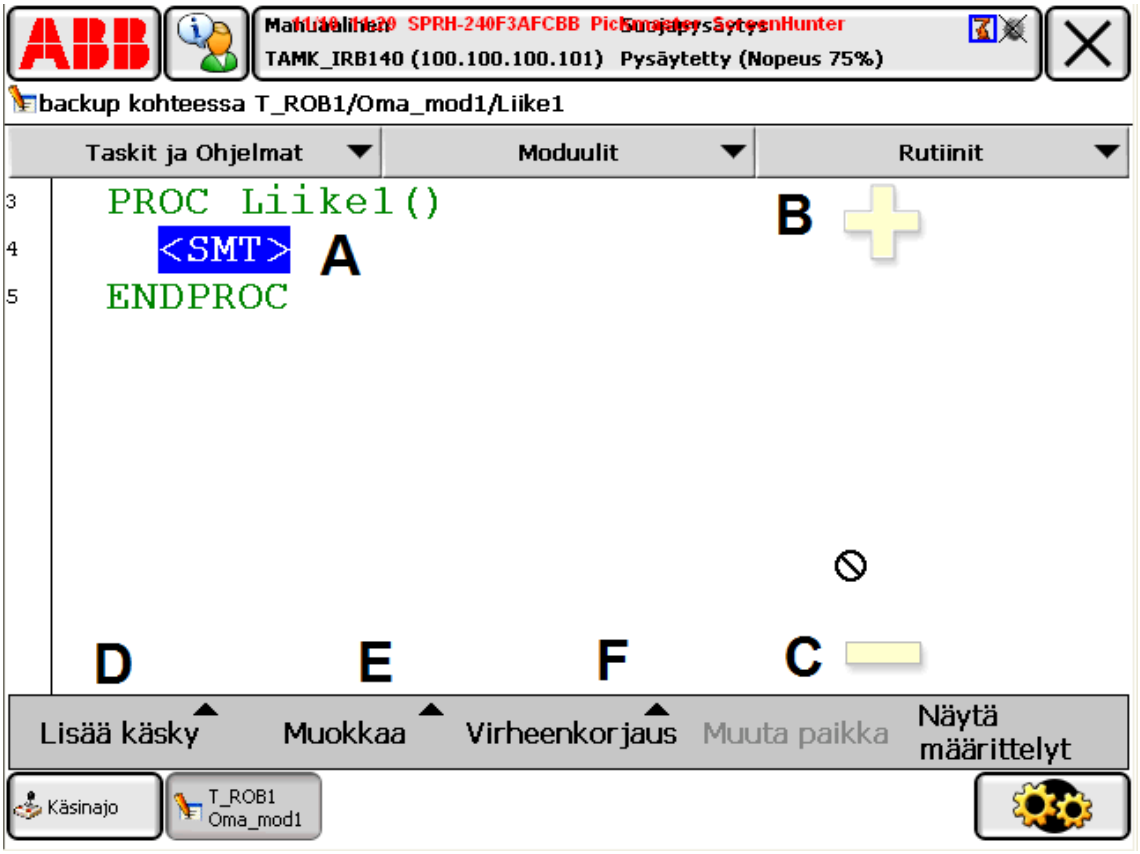
4.4.4. RobotStudio

RobotStudio on tietokoneohjelma, jolla kyetään käyttämään ja hallinnoimaan robottia. Ohjelmalla kyetään tekemään erinäisiä ohjelmia etänä. Ohjelma ei siis sido ohjelmoijaa robottiin. RobotStudiolla sisältää paljon erilaisia järjestelmän osia, joilla voidaan tehdä valmiita kokonaisuuksia. RobotStudio – ohjelma sisältää mm.

- System builder toiminnon, jolla kyetään tekemään järjestelmien luomista ja asentamista.
- Tiedostonhallintaohjelman, jolla kyetään siirtämään dataa PC:n ja ohjainten välillä.
- Tapahtumaloki, johon merkataan kaikki robotin valvontaan liittyvät tapahtumat.
- Kokoonpanoeditorin käytössä olevan järjestelmän järjestelmäparametrien muokkaamista varten.

4.5 Ohjelmaeditori

Ohjelmaeditori on editori, jossa tehdään robotille opettamalla ohjelmoitu ohjelma. Ohjelmalle pitää opettaa kaikki ne pisteet, johon haluamme robotin menevän tai tekevän jotain. Ohjelma-editorivalikkoon avautuu sivu, johon voidaan tehdä ohjelma. SMT-tilaan tehdään haluttu ohjelma korostamalla näytössä se kohta, johon haluamme tehdä ohjelmaa. Lisää käskyvalikosta valitaan aina haluttuja käskyjä ohjelmaan. Oletus luokkana on Common eli yleinen käskyluokka.



Kuva 22 Ohjelmaeditori, johon tehdään ohjelma

A. SMT Liikekäskytila. Tila, johon tulee halutut liikekäskyt haluamaasi järjestykseen.
B. Zoom in
C. Zoom out
D. Lisää käsky -valikosta voidaan lisätä käskyjä ohjelmaan.
E. Valikosta muokataan ohjelmaa
F. Testausvalikko. Valikko josta kyetään testaamaan tehtyä ohjelmaa.

- Taskit ja Ohjelmat -valikko: Valikosta kyetään tekemään robotille uudet ohjelmat, lataamaan vanhoja ohjelmia, tallentamaan luotuja ohjelmia ja ohjelman testausta. Ohjelmat koostuvat usein kolmesta erinäisestä osasta, päärutiinista, alirutiinesta ja ohjelmadatasta.
- Moduulivalikosta kyetään tekemään uusi moduuli ohjelmalle, lataamaan vanhoja moduuleja, tallentamaan moduuleja, nimeämään moduuleja uudelleen ja poistamaan vahoja moduuleja. Moduulit voidaan erotella ohjelmamoduuleihin, joista jo-

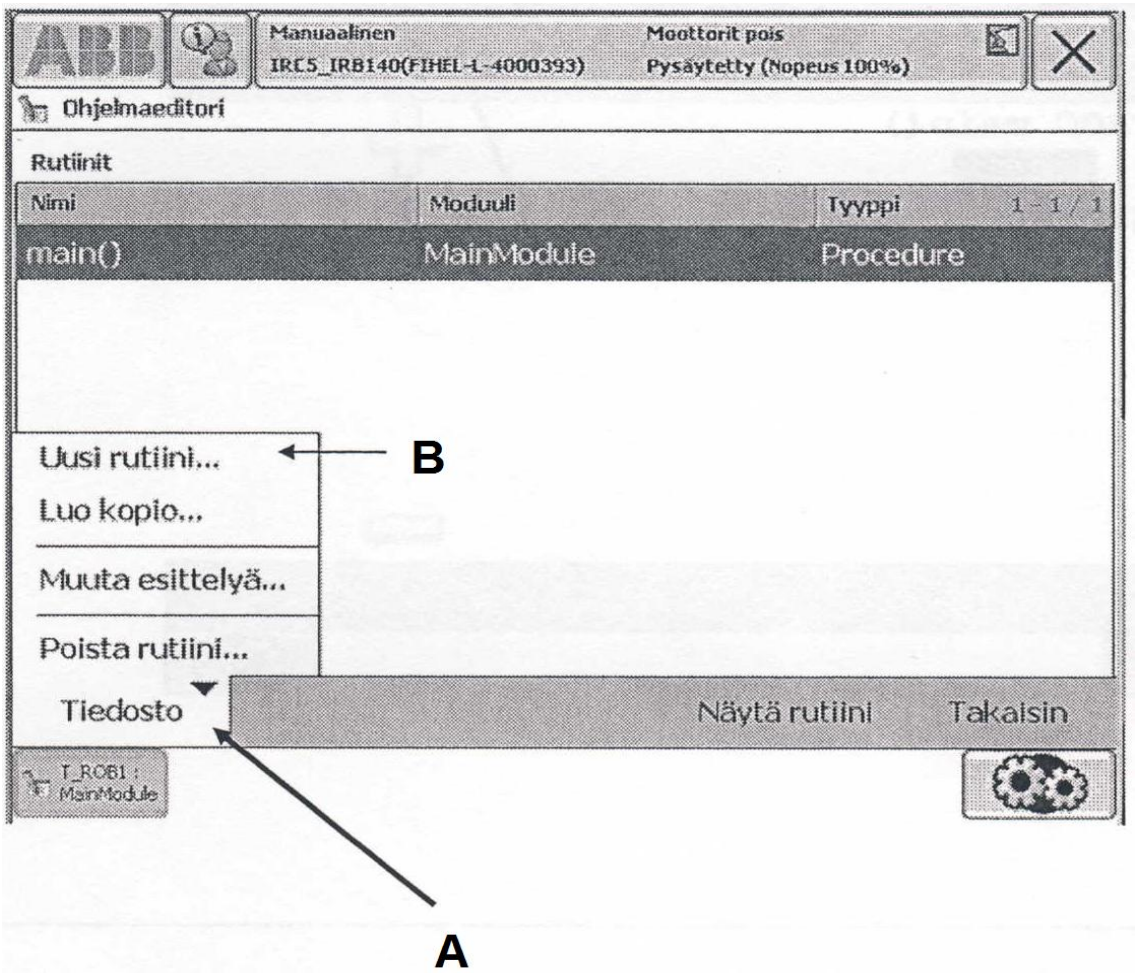
kainen moduuli sisältää joukon rutiineja ja ohjelmadataa. Moduulit voivat myös sisältää CAD:llä valmiiksi tuotettuja sijainteja.

- Rutiinivalikko: Rutiini on usein toistuva liike tai toiminto käskysarjojen jonossa. Rutiinit tulee jakaa alirutiineihin ohjelman selkeyden pitämiseksi. Pääohjelmaa on näin helpompi lukea. Ohjelman rakennetta on hyvä suunnitella jo ennen ohjelmoinnin aloittamista. Esimerkki rutiinina olkoon manipulaattorin tehtävä, josta manipulaattori ottaa linjaston toisesta päästä kappaleen, vie sen esim. cnc-sorviin. Sorvauksen jälkeen manipulaattori vie työstetyn kappaleen uudella linjastolle, josta kappale jatkaa matkaansa. Aliohjelma voisi olla esim. Hae palikka, Vie koneelle, Hae koneelta, Jätä palikka.

4.5.1 Rutiinin luominen ohjelmaan

Rutiinin teko on helppoa, yksinkertaista ja niitä pyritään tekemään ohjelman selkeyttämisen, ja ohjelmoitavuuden nopeuttamiseksi.

Rutiinit luodaan menemällä ABB-valikon kautta ohjelmaeditoriin. Ohjelmaeditorista valitaan yläpalkeesta rutiinit. Uuden rutiinin voi luoda menemällä tiedostot valikkoon, josta kyetään muokkaamaan, luomaan uusia ja poistamaan rutiineja. Kun haluamme luoda uuden menemme näin ollen luo uusi rutiini -kohtaan.

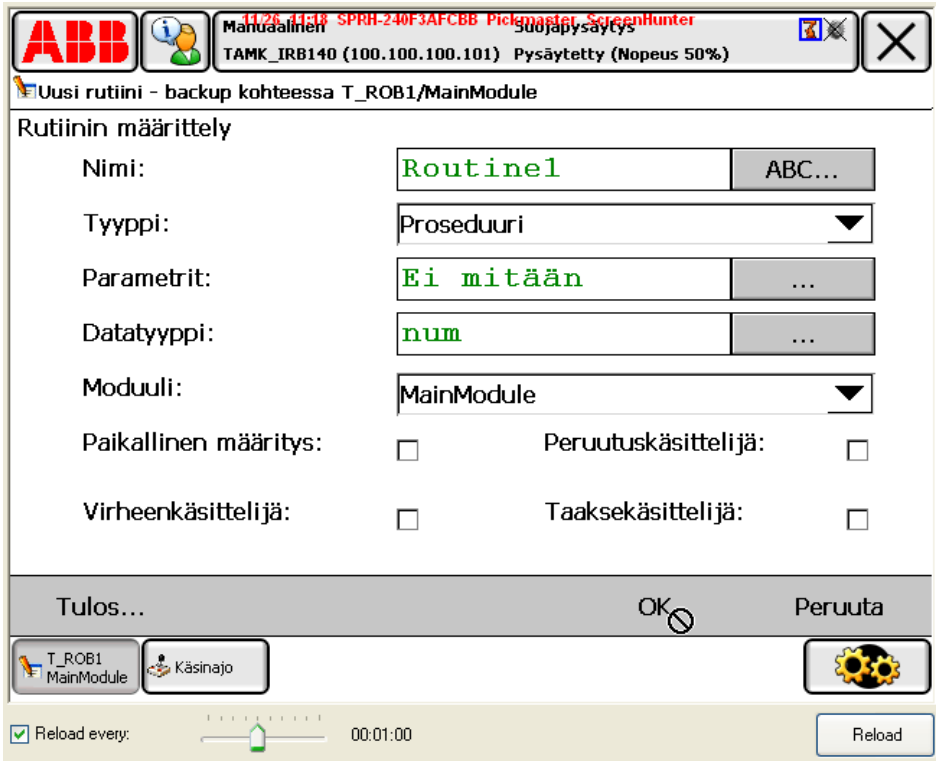


Kuva 23 Rutiinin luominen

Uuden rutiinin määrittelyssä rutiinille pitää antaa nimi, jotta sen löytää muiden rutiinien joukosta. Nimi olisi hyvä olla rutiinin kuvaamista. Esim. Hae kappale tai vie kappale paletille.

Uuden rutiinin tekeminen:

- Määritellään rutiinille kuvaava nimi.
- Määritellään tyyppi. Kuvaa siis minkälainen rutiini on. Proseduuri, toiminto ja keskeytys.
- Määritellään parametrit. Määritellään käskyille tarvittavia tietoja.
- Määritellään datatyyppi. Määritellään datan laatu. Esim num, bool, byte.
- Valitaan moduuli. Valikosta valitaan moduuli johon rutiini sijoittuu.



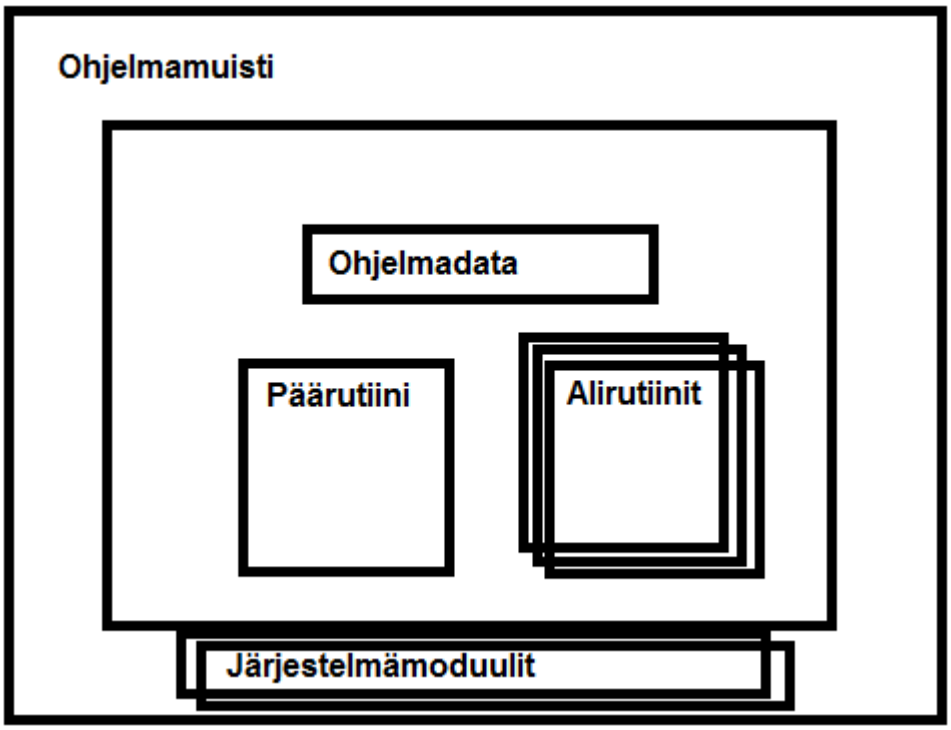
Kuva 24 Rutiinin määrittämisen valikko

4.5.2. Ohjelmanrakenne.

Robotti tarvitsee näin ohjelman toimiakseen. Ohjelma koostuu pääsääntöisesti kolmesta erinäisestä osasta:

- Päärutiini: Ohjelma suorittaminen alkaa päärutiinista. Päärutiini vastaa robotin työkierrosta.
- Useita alirutiineja: Alirutiineja käytetään ohjelman selkeyttämiseksi ja sillä jaetaan ohjelma pienempiin osiin. Alirutiineja kutsuu joko päärutiini tai joku toinen rutiini. Kun rutiini on suorittanut tehtävänsä loppuun, ohjelma siirtyy seuraavan rutiinin ensimmäiseen käskyyn.
- Ohjelmadata: ohjelmadataa käytetään sijaintien, numeeristen arvojen (rekisterit ja laskurit), koordinaatiojärjestelmien yms. määrittämiseen. Datoja voidaan muuttaa itse manuaalisesti, mutta myös ohjelma voi sen tehdä automaattisesti. Datat voi muuttua automaattisesti kun tehdään esim. sijainnin uudelleenmäärityksessä tai kun päivitetään laskuria.

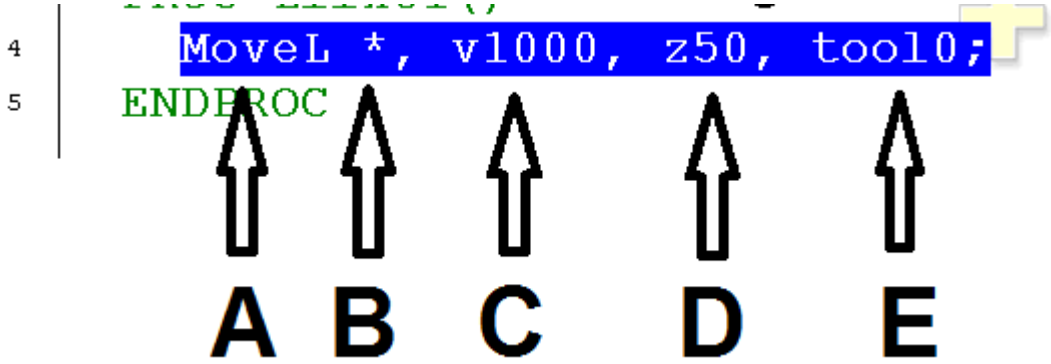
- Järjestelmämoduulit: Järjestelmämoduulissa säilytetään rutiineja ja dataa, jotka liittyvät asennuskokonaisuuteen. Järjestelmämoduulit eivät liity mitenkään ohjelmaan vaan sisältää työkaluihin ja huoltoihin liittyviä rutiineja.



4.5.3. Liikekäskyn lukeminen

Liikekäskyjä eli tyylejä miten robotti liikkuu pisteestä toiseen on kolme erilaista tyyliä.

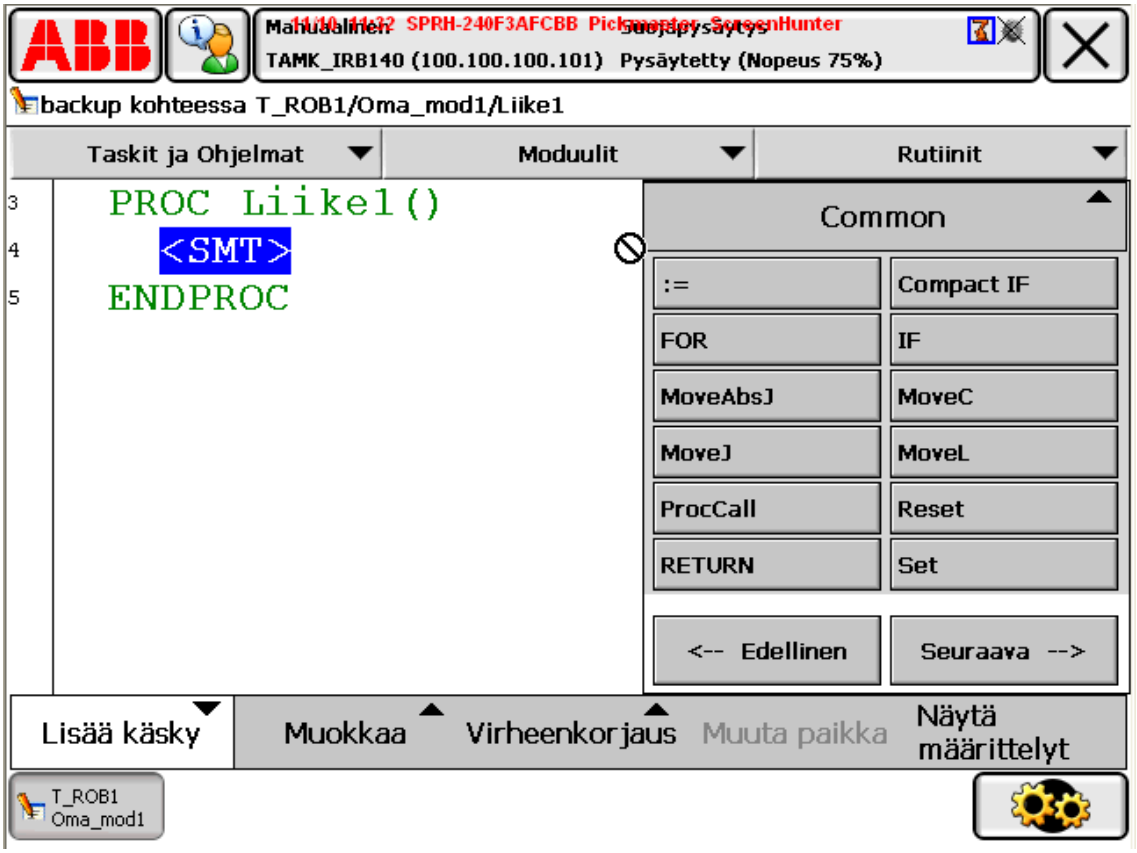
- MoveC = Ympyrämuotoinen liike
- MoveL = Lineaarinen liike
- MoveJ = Yhdistetty liike. Käsiajoliike.



A. Käskyn nimi tai toiminto. Kyseinen käsky käskee robotin liikuttamaan robottia lineaarisesti.
B. Käskypisteen arvot piilotettuina
C. Määrittää käskyn liike nopeuden
D. Määrittää robotin pisteen tarkkuuden
E. Määrittää aktiivisen työkalun

5. Ohjelman teko

Ohjelman teko tapahtuu ohjelmaeditorissa.



Kuva 25 Ohjelman tekoa kuvaava valikko, josta saadaan syötettyä ohjelmalle tietoa.

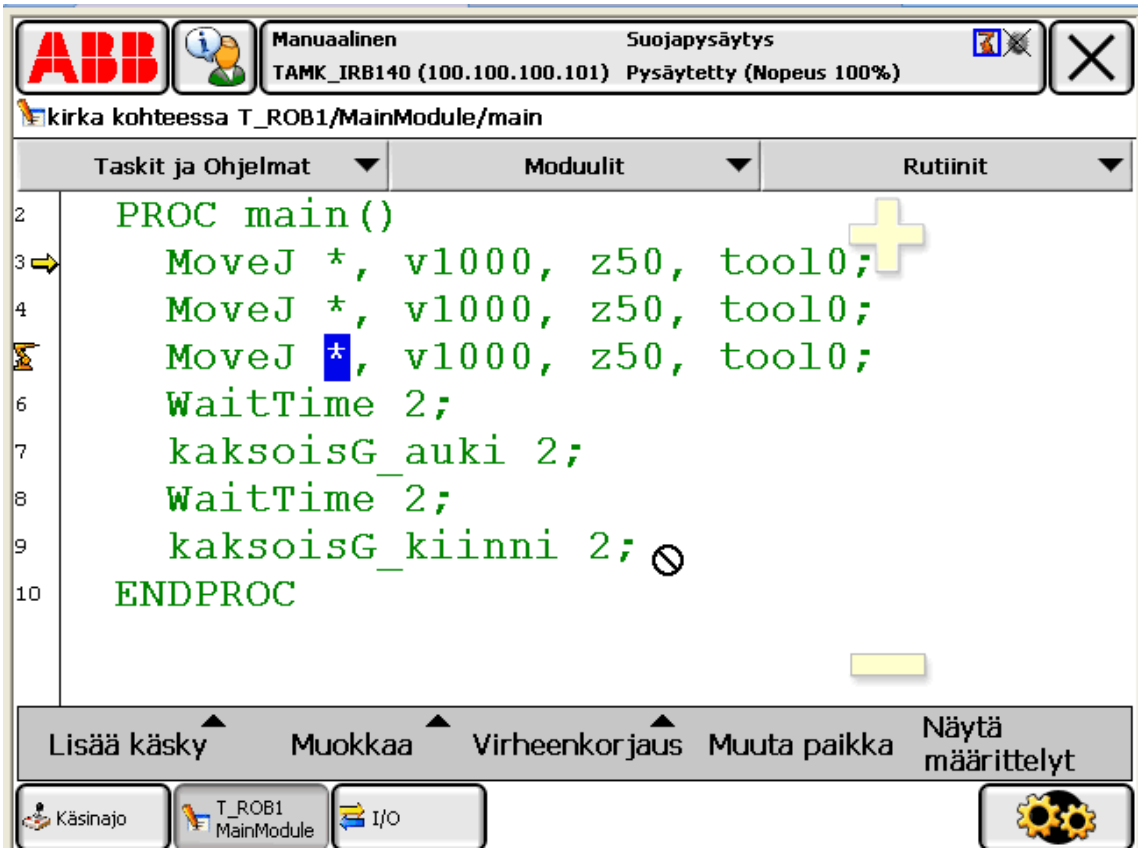
Ohjelmaan aletaan syöttämään pisteitä ajamalla robotti haluttuun pisteeseen ja valikosta haetaan pisteeseen haluttu liikemenetelmä. Käskyt löytyvät Lisää-käskyvalikosta. Käskyjä on robotiikassa valtavasti, joten työssäni käytetään vain yleisimpiä käskyjä eli common-käskyjä. Loput käskyt löytyvät liitteinä.

Common käskyt

:=	Nimitetään arvo datalle
For	Toistaa jakson niin monta kertaa kun ohjelmoija haluaa
MoveJ	Yhdistetty liike
ProcCall	Kutsu toinen rutiini
Return	Palaa alkuperäiseen rutiiniin
Compact IF	Toteuttaa käskyn vain jos käskytila on päällä

IF	Toteuttaa jakson erilaisia käskyjä riippumatta onko tilat päällä.
MoveC	Työkalupiste liikkuu ympyrämuotoisesti. On robotin perusliikekäsky.
MoveL	Työkalupiste liikkuu lineaarisesti. On robotin perusliikekäsky.
Reset	Resetoi digitaalisen ulostulon signaalin nollaan.
Set	Asettaa digitaalisen ulostulon signaalin ykköseksi
Wait DI	Odottaa kunnes digitaalinen input on asettunut
WaitTime	Odottaa ohjelmoijan asettaman ajan tai odottaa robotin pysähtymistä
WHILE	Toistaa jaksoa kunnes tila on vaihtunut.
WaitDo	Odottaa kunnes digitaalinen output on asettunut
WaitUnit	Odottaa kunnes tila on asettunut päälle.

5.1. Käskyjen lisääminen

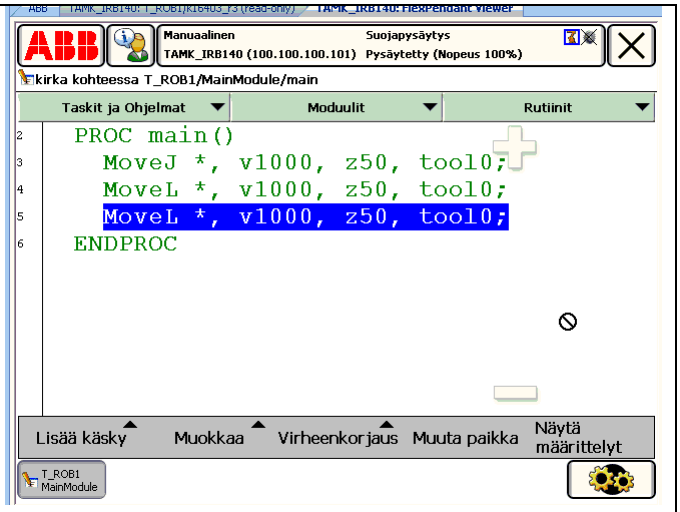
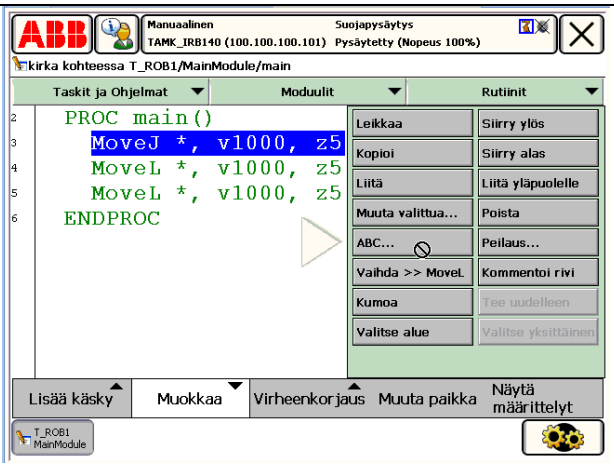
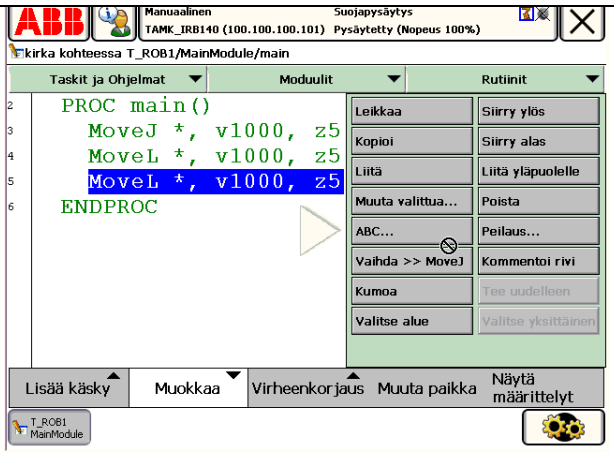


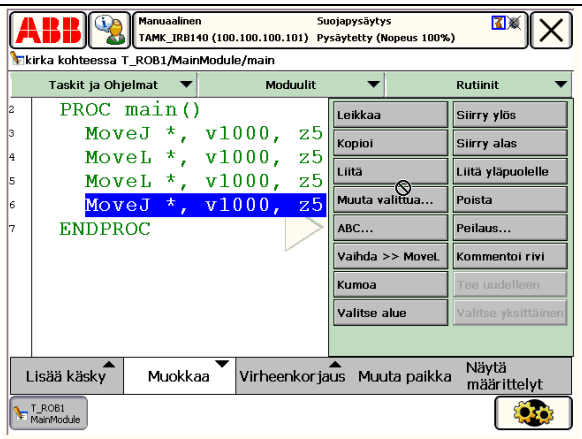
Kuva 26 Valmis ohjelma

Kuvassa oleva ohjelma on yksinkertainen mutta silti siinä on paljon hyviä yleisiä toimivia elementtejä kuten liikekäskyjä, viiveaikaa ja I/O- toimintoja. Nämä elementit ovat yleisiä toimintoja robotiikassa.

5.1.1. Liikekäskyjen kopioiminen

Liikekäskyjen kopioiminen on tärkeä osa robotin ohjelmoimista. On tärkeää että robotti osaa mennä uudestaan samaan pisteeseen, jossa robotti on jo käynyt. Esim. ladonta- ja hitsaustehtävät. Olen tehnyt pienen ja yksinkertaisen liikkeistä kootun ohjelman jossa kopioidaan ohjelman ensimmäinen piste.

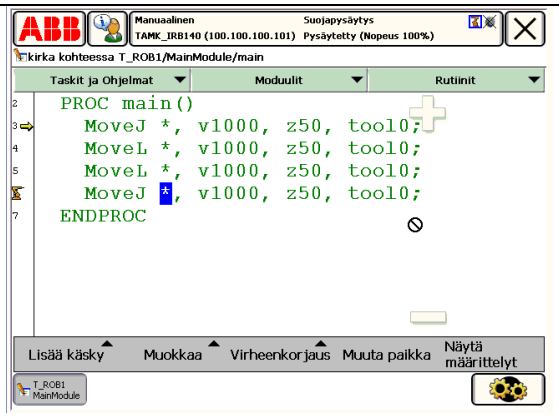
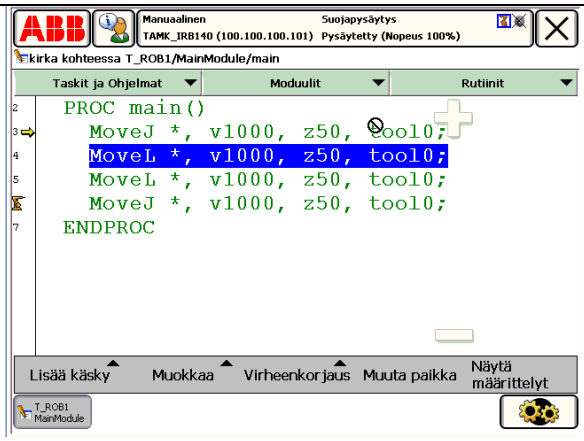
Vaihe1	Tehdään haluttu ohjelma normaalisti.	
Vaihe2	Valitaan haluttu piste, joka halutaan kopioida painamalla pistettä. Mennään muokkaa valikkoon. Kopioidaan piste.	
Vaihe3	Viedään kursori haluttuun pisteeseen mihin kopioitu piste halutaan laittaa.	

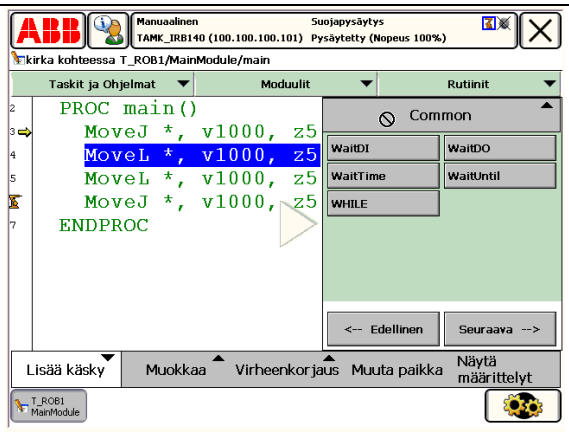
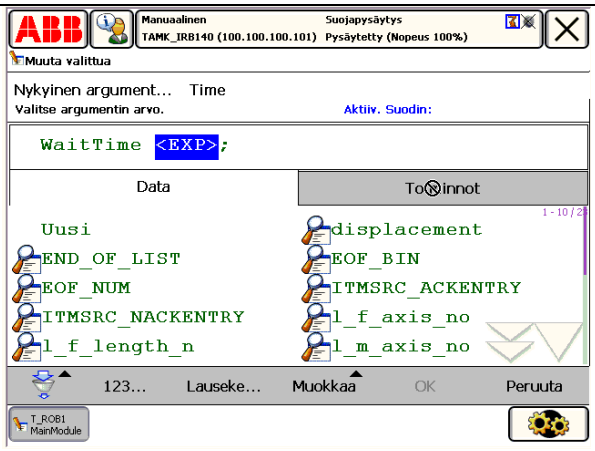
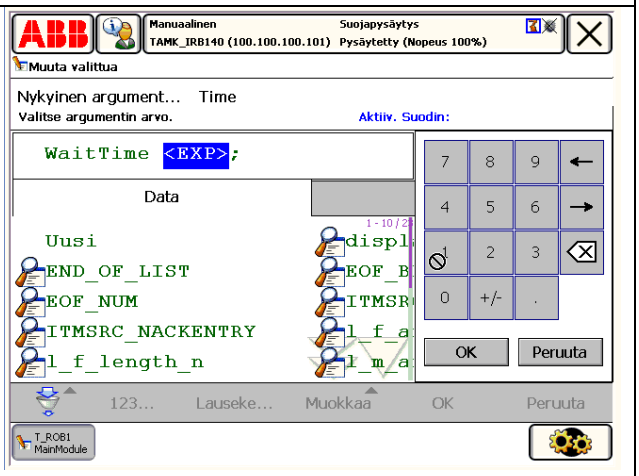
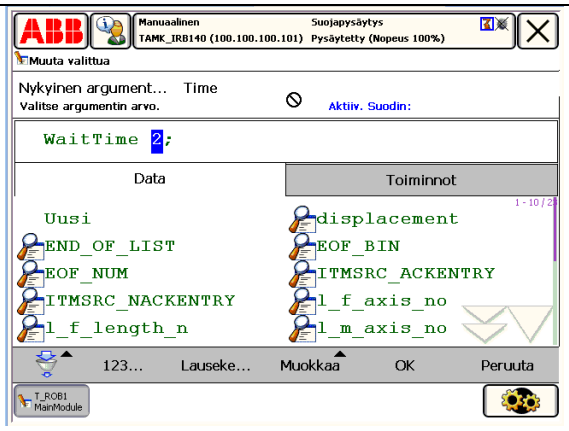
Vaihe4	Halutun pisteen kohdalla painetaan liitä. Liittämisen jälkeen kopioitu piste tulee näkyviin halutun pisteen alapuolelle.	

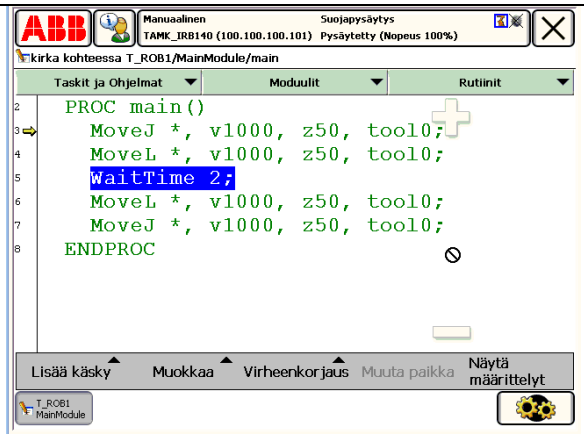
Kopioimalla kyetään muitakin haluttuja elementtejä kopioimaan kuin vain liikekäskyjä ja niiden pisteitä.

5.1.2. Viiveajan ohjelmoiminen

Viiveaikoja käytetään paljon erilaisissa robotiikan toiminnoissa. Robotti voidaan ohjelmoida siten, että se odottaa tietyn ajan tai toimintoa. Ajastimien käyttö toimii useasti käsikädessä I/O-käskyjen kanssa. Käytetään liikekäskyn kopioimisesta tuttua ohjelmaa hyödyksi viiveajan ohjelmoimisessa.

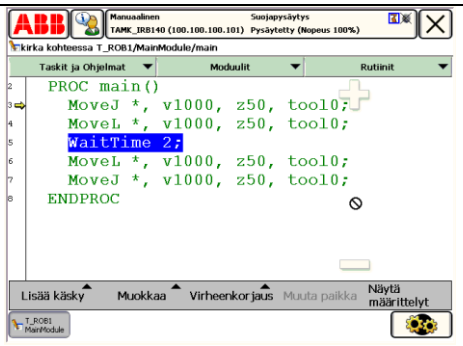
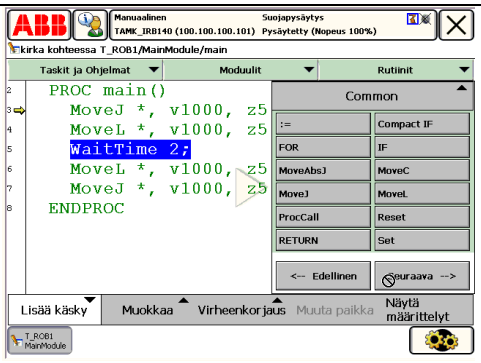
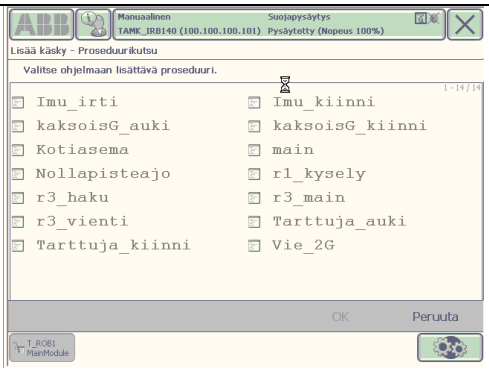
Vaihe1	On tehty haluttu ohjelma.	
Vaihe2	Valitaan kohta, johon haluamme tehdä viiveajan	

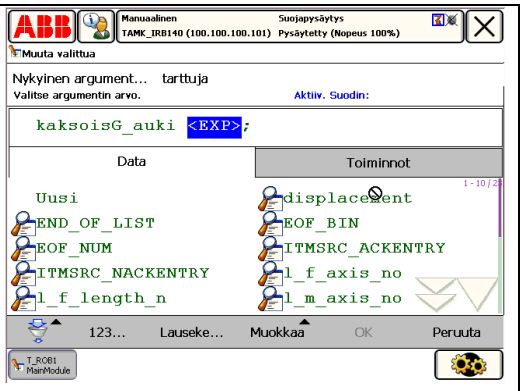
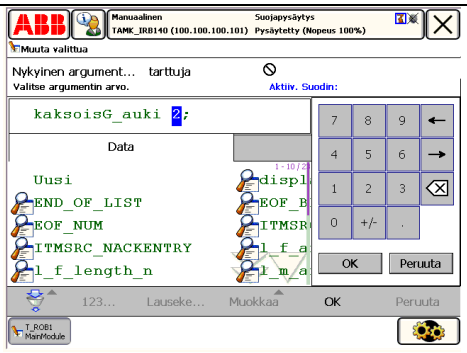
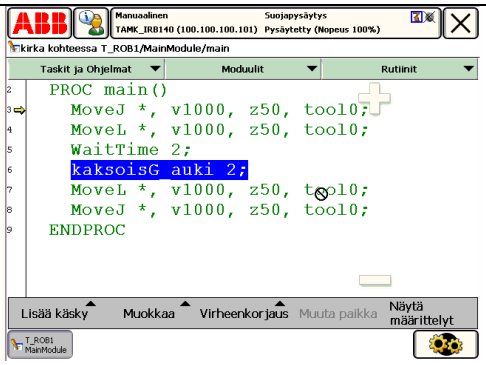
Vaihe3	Kun piste on päätetty otetaan piste aktiiviseksi. Otetaan lisää käsky valikosta common käskyistä WaitTime. Klikataan WaitTime.	
Vaihe4	Tulee valikko, josta voidaan valita valmiiksi tehtyjä datoja. Valitaan kuitenkin ajaksi 2 sekuntia. Klikataan 123…	
Vaihe5	Ilmestyy näytön oikeaan reunaan numeronäppäimistö, josta valitaan aika, jonka robotti odottaa pisteessä.	
Vaihe6	Halusin, että robotti odottaa 2 sekuntia, joten painoin 2.Tämän jälkeen painetaan ok.	

Vaihe7	Hyväksynnän jälkeen ohjelmaan tuli haluttu odotus.	

5.1.3. I/O käskyn ohjelmointi

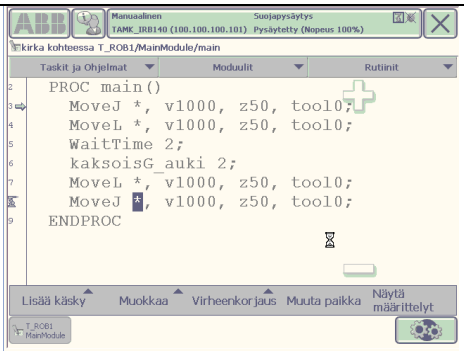
Kappaleessa kerrotaan miten saadaan ohjelmaan I/O-käskyjä. Työssäni I/O-käskyjä otti vastaan tarttuja. Muita vastaavia I/O-käskyjä voisi olla mm. liukuhihna.

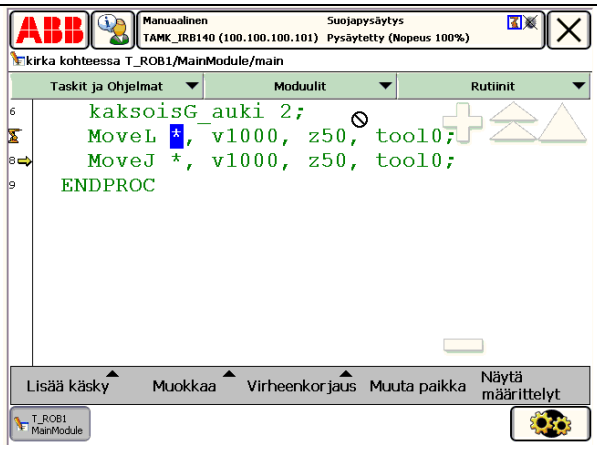
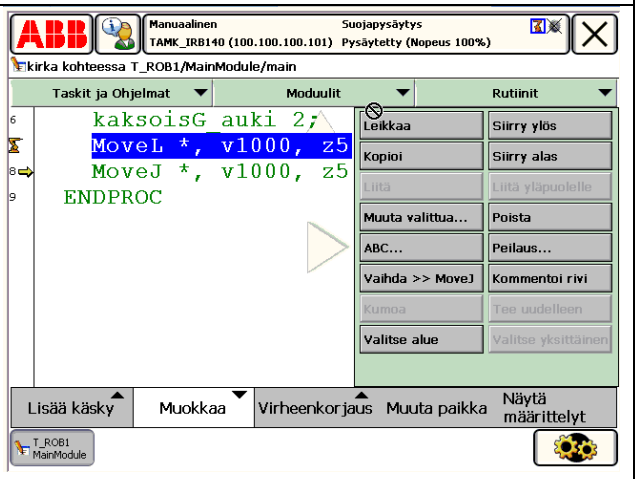
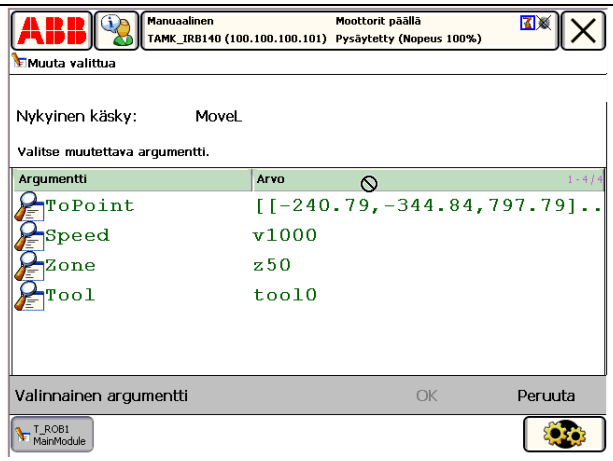
Vaihe1	Valitaan piste johon, halutaan tehdä I/O käsky. Valitaan haluttu piste aktiiviseksi.	
Vaihe2	Lisää käsky valikosta ProcCall-käsky, joka avaa valikon, jossa on robotin kaikki I/O-käskyt.	
Vaihe3	Valitaan listalta minkä I/O-käskyn ohjelmoija haluaa. Työssäni käytin kaksoisG auki -käskyä.	

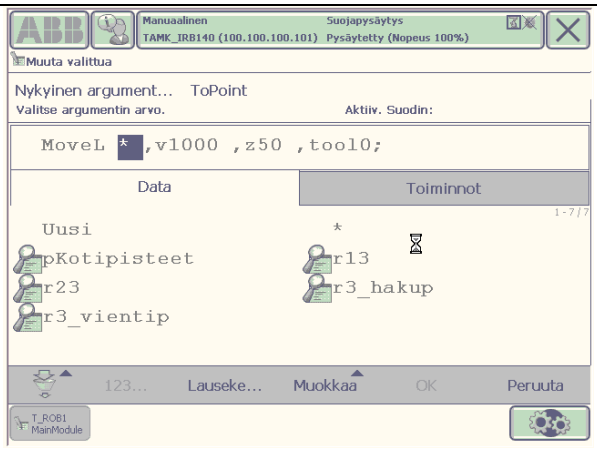
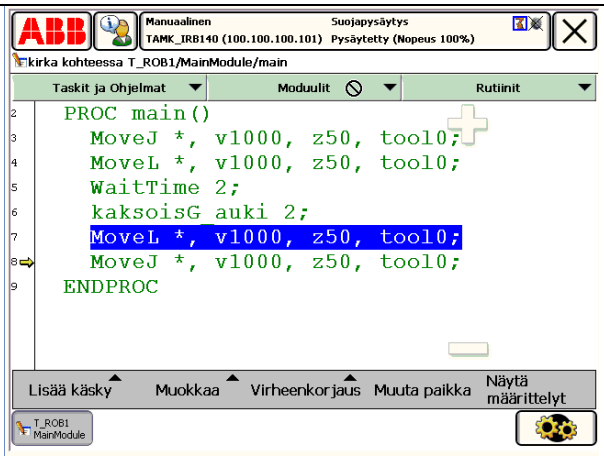
Vaihe4	Valitaan kaksoisG_auki. Koska työkalussa on kaksi tarrainta robotti haluaa kysyä otetaanko tarrain 1 vai 2 käyttöön.	
Vaihe5	Annetaan numerolla viittaus siihen kumpi tarrain otetaan käyttöön. Käytin työssäni tarrainta 2.	
Vaihe6	Ja haluttu käsky ilmestyy ohjelmalistaan.	

5.1.4. Pisteen muuttaminen koordinaatistossa.

Ohjelmassa olevia käskyjä ja koordinaatiopisteitä voidaan muokata uudestaan haluammallaan tavalla. Voidaan vaihtaa yhden tai useamman argumentin paikkaa.

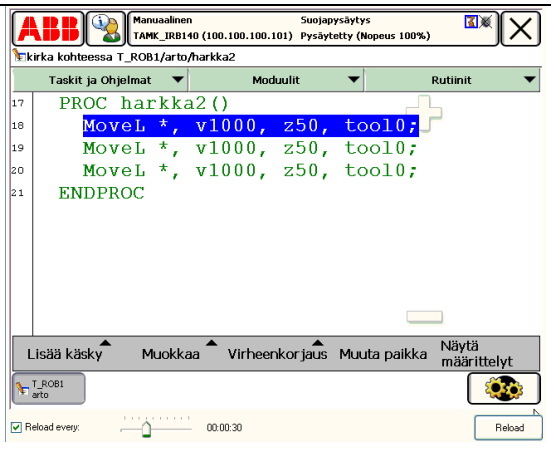
Vaihe1	Ohjelma, josta muutetaan toiseksi viimeistä pistettä.	

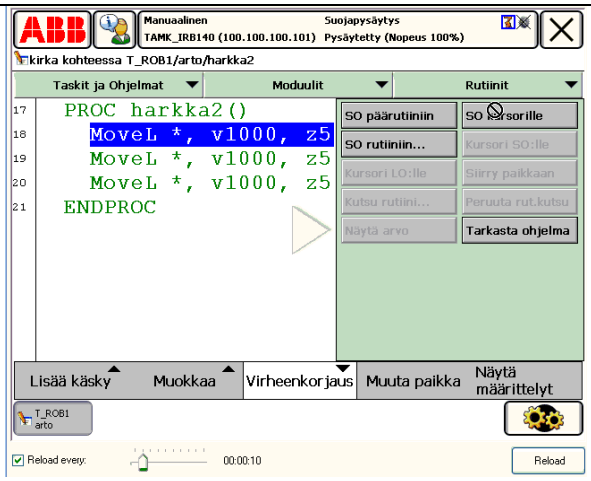
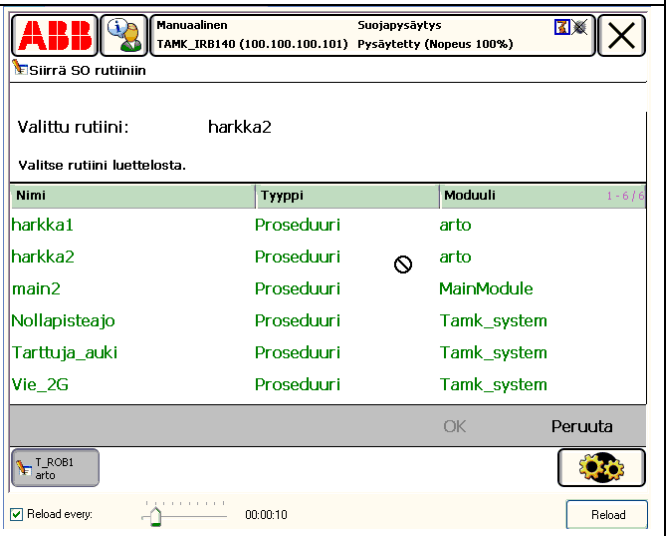
Vaihe2	Ohjelma pitää ajaa vaihe kerrallaan pisteeseen, jota halutaan muokata.	
Vaihe3	Vahvennetaan muutettava piste. Muokkaa valikosta valitaan kohta muuta valittua.	
Vaihe4	Ajetaan uusi haluttu piste.	

Vaihe5	Hyväksytään piste painamalla ok. Valikosta voidaan vaihtaa uuden pisteen tarkuudet ja pisteen nopeudet.	
Vaihe6	Uusi piste asettuu näytölle. Halutessa voidaan testata uusi muokattu ohjelma läpi.	

5.1.5. Ohjelman tarkistus ja testaus

Ohjelma pitää testata ennen kuin sillä aletaan tekemään tuotantoa. Ohjelma on hyvä käydä läpi piste pisteeltä, koska virheiden mahdollisuus on aina olemassa. Osiossa kerrotaan miten kyke-net katsomaan millaisen ohjelman olet tehnyt.

Vaihe1	Haluttu ohjelma on valmis, ja se halutaan testata nyt läpi.	

Vaihe2	Otetaan valikosta virheenkorjaus ja valitaan SO rutiini.	
Vaihe3	Valitaan mihin rutiiniin halutaan viedä tehty ohjelma. Valitaan haluttu rutiini valitaan se aktiiviseksi ja hyväksytään se ok-painikkeella.	

Haluttu ohjelma on nyt valmiina testattavaksi rutiinissa. Kun halutaan testata ohjelma kuolleenmiehenkytkin pitää näin olla aktiivisena. Eli kapula pitää olla kokoajan ohjelmoijan käsissä. Silloin kun kuolleenmiehenkytkin vapauttaa servot aktiiviseksi voidaan painaa kapulasta play-nappia, jolloin robotti alkaa suorittaa tehtyä ohjelmaa.

6. Harjoitustehtävät

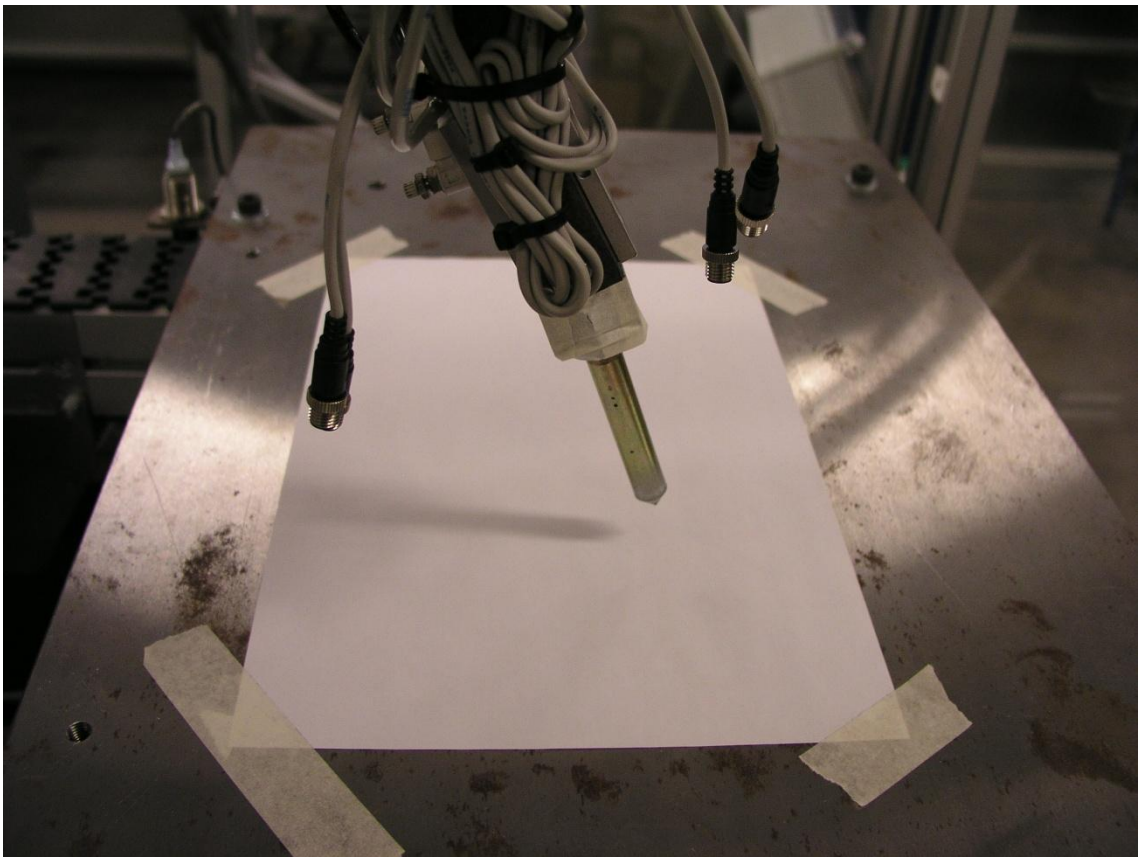
Harjoitustehtävien tarkoituksena on saada robotin käyttäjälle ne perusosaamisen taidot, josta on helppo lähteä kehittämään itseään robotin ohjelmoijana. Harjoitustöissä pyritään kiteyttämään opiskelijalle peruskäskyt ja toiminnot.

6.1 Tehtävä 1

Ensimmäisessä tehtävässä opetellaan robotin käsiajo. Käytetään robotin kynätyökalua hyväksi ja piirretään sillä paperille neliö, kolmio ja ympyrä.

On siis tärkeää tietää miten robotti liikkuu, käyttäytyy ja miten sitä liikutetaan.

Tehtävä alkaa ja loppuu kotiasema 2 paikasta. Kotiasemat ovat jo valmiiksi ohjelmoitu robotille. Kotiasemasta lähdetään ajamaan robottia käsiajolla kohti paperia, joka on teipattu pöydälle. Paperille tehdään halutut piirrokset ja ajetaan robotti takaisin kotiasemaan.



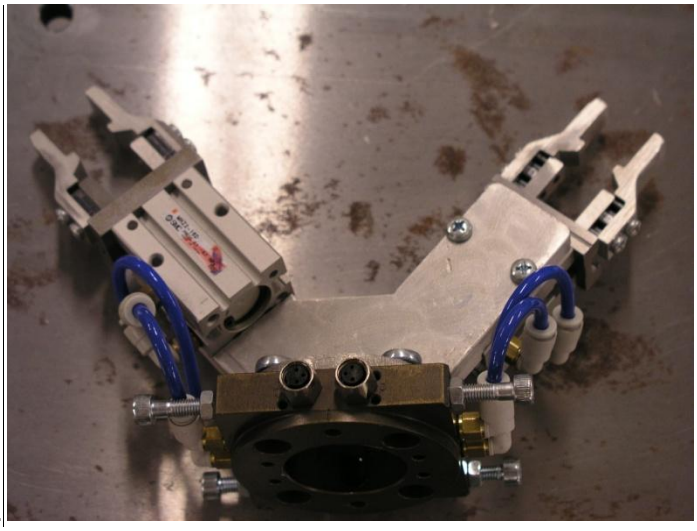
Kuva 27 Esimerkkityökalu, jolla voidaan tehdä halutut piirrokset paperille

6.2 Tehtävä 2

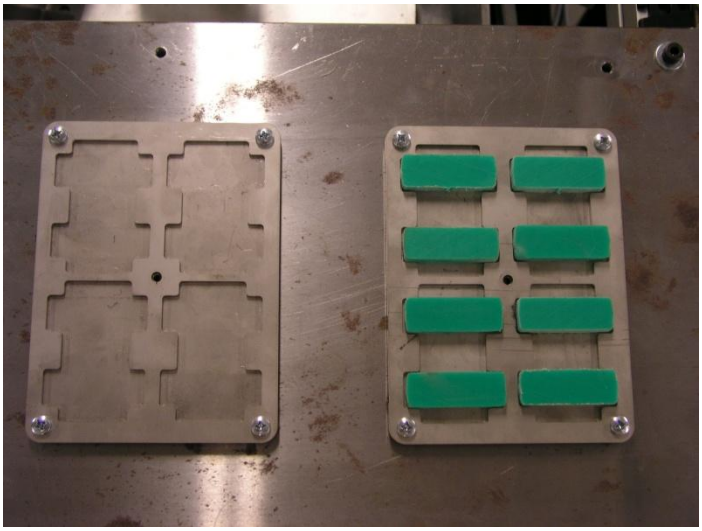
Lasketaan kaksi metallista palettia robotin pöydälle ja toinen paleteista täytetään muovisista kappaleista. Tehtävänä on siirtää täydestä paletista kappaleet tyhjälle paletille yksi kerrallaan, kunnes täysi paletti on tyhjä.

Työssä täytyy käyttää tarttujatyökalua ja I/O-käskyjä.

Tehtävä alkaa ja loppuu kotiasema2-paikkaan



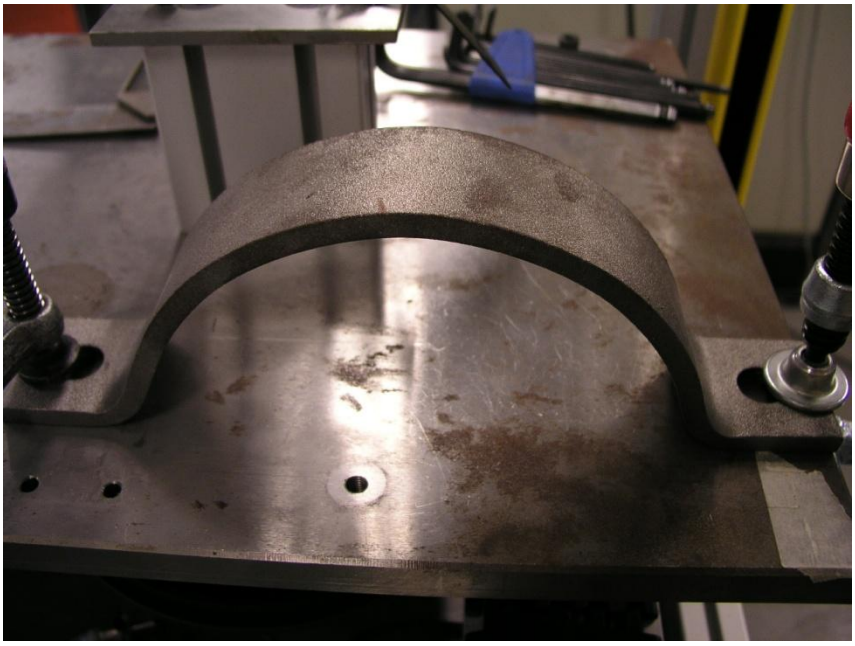
Kuva 28 Työkalu, jolla kyseinen harjoitustyö tehdään



Kuva 29 Paletit

6.3 Tehtävä 3

Tehtävänä on tehdä liikerata, joka myötäilee kuvan kappaleen kaaretta. Työkaluna käytetään tehtävä1-työkalua. Liike ja kynän kulma pitää olla suhteessa oikeat oikeaan hitsaukseen. Tehtävässä pyritään luomaan mielikuva hyvästä esimerkistä teollisuuden hitsauskappaleesta.



Kuva 30 Tehtävän 3 harjoituskappale

7 Lähdeluettelo

Painatetut lähteet.

1. IRC5 Ohjelmoinnin peruskurssi R102 kurssi materiaali
2. User's Guide 3HAC 7793-1 For BaceWare OS 4.0.70
3. RAP Service Specification 3HAC 7697-1 RAP communication OS 4.0
4. Product Manual IRB 140 3HAC 7564-1 / M2000

Sähköiset lähteet

1. ABB- viralliset kotisivut. www.abb.com
2. Suomen automaatioseuran sivut. www.automaatioseura.com
3. Kukan viralliset kotisivut. www.kuka.com
4. Motomanin viralliset kotivivut. www.motoman.fi

7. Liitteet

ABB IRB140 robotin loput käskyt ja niiden tarkoitus.

Prog Flow Käskyt

Break Stop program execution temporarily for debugging purposes.

Compact IF Execute one instruction only if a condition is satisfied.

Exit Cycle Stop the current cycle and move the program pointer to the first instruction in the main routine. When the execution mode *CONT* is selected, execution will continue with the next program cycle.

GOTO Jump to a label

Label Specify a label (line name)

Retun Return to the original routine

CallByVar Call procedures with specific names

EXIT Stop program execution when a program restart is not allowed

FOR Repeat a section of the program a number of times.

IF Execute a sequence of different instructions depending on whether or not a condition is satisfied.

ProcCall Call (jump to) another routine

Stop Stop program execution

SystemStopAction Stop program execution and robot movement

WHILE Repeat a sequence of different instructions as long as a given condition is satisfied

TEST Execute different instructions depending on the value of an expression

Various E.g Käskyt

:= Assign a value to data

EraseModule Erase a module from the program memory.

Save Save a program module.

UnLoad Unload a program module from the program memory

WaitDo Wait until a digital output is set

CancelLoad Cancel the loading of a module that is being or has been loaded with the instruction *StartLoad*

Comment Comment on the program

Load Load a program module into the program memory

StartLoad Load a program module into the program memory during execution

WaitDI Wait until a digital input is set.

WaitLoad Connect the module, if loaded with *StartLoad*, to the program task

WaitTime Wait a given amount of time or to wait until the robot stops moving

WaitUntil Wait a given amount of time or to wait until the robot stops moving

.

Settings Käskyt

AccSet Define the maximum acceleration.

ConfL Configuration check on/off during linear motion DitherDeact

Eoffson Activate an external axis offset

GripLoad The payload of the gripper

MotionSup Deactivates/activates motion supervision

ConfJ Configuration control on/off during joint motion

DitherAct Enables dither functionality for soft servo

EOffsOff Deactivate an external axis offset

EOffsSet Activate an external axis offset by specifying a value

MechUnitlLoad Defines a payload for a mechanical unit

PDispOff Deactivate program displacement

PDispOn Activate program displacement

SingArea The interpolation method through singular points

SoftDeact Deactivate the soft servo

PDispSet Activate program displacement by specifying a value

SoftAct Activate the soft servo for one or more axes

VelSet The maximum velocity and velocity override

Motion&Proc Käskyt

ActUnit Activate an external mechanical unit

MoveAbsJ Absolute joint movement

MoveCDO Moves the robot circularly and sets a digital output in the middle of the corner path.

MoveJ Joint movement

MoveL TCP moves along a linear path

SearchC TCP along a circular path

DeactUnit Deactivate an external mechanical unit

MoveC TCP moves along a circular path

MoveExtJ Moves a linear or rotational external axis without TCP

MoveJDo Moves the robot by joint movement and sets a digital output in the middle of the corner path.

MoveLDo Moves the robot linearly and sets a digital output in the middle of the corner path.

SearchL TCP along a linear path

I/O Käskyt

AliasIO Define a signal with an alias name

IOBusStart Start an I/O bus.

IODisable Disable an I/O module

PulseDo Generate a pulse on a digital output signal

Set	Set a digital output signal (to 1)
SetDo	Change the value of a digital output signal (symbolic value; e.g. *high/low*)
InvertDo	Invert the value of a digital output signal
IOBusState	Get current status of the I/O bus.
IOEnable	Enable an I/O module
Reset	Reset a digital output signal (to 0)
SetAO	Change the value of an analog output signal
SetGo	Change the value of a group of digital output signals
WaitDI	Wait until a digital input is set or reset
WaitDO	Wait until a digital output is set on reset

Communicate käskyt

ClearIOBuff	Clear input buffer of a serial channel
Close	Close the channel/file
CopyRowBytes	Copy from one rawbytes variable to another
Open	Open a serial channel/file for binary transfer of data
ReadAnyBin	Read from any binary serial channel
ReWind	Set the file position to the beginning of the file
ClearRawBytes	Set a rawbytes variable to zero
Copyfile	Copy a file.
ErrWrite	Write text on the FlexPendant display and simultaneously store that message in the progam's error log.
PackRawBytes	Pack the contents of a variable into a "container" of type rawbytes
ReadRawBytes	Read data of type rawbytes from a binary serial channel/file/ field bus
SCWrite	Send a message to the superordinate computer
TPErase	Clear the FlexPendant operator display

TPReadNum Read a numeric value from the FlexPendant

TPWrite Write text on the FlexPendant operator display

UIShow Open an application on the FlexPendant from RAPID

Write Write text to the channel/file

WriteBin Write to a binary serial channel/file

TPReadFK Label the function keys and to read which key is pressed

TPShow Choose a window on the FlexPendant from RAPID

UnPackRawBytes Unpack the contents of a “container” of type rawbytes to a variable

WriteAnyBin Write to any binary serial channel/file

WriteRawBytes Write data of type rawbytes to a binary serial channel/file/fieldbus

WriteStrBin Write a string to a binary serial channel/file

Interrupts

CONNECT Connect a variable (interrupt identity) to a trap routine

IDelete Cancel (delete) an interrupt

IEnable Enable all interrupts

IPers An interrupt when changing a persistent.

ISignalAO An interrupt from an analog output signal

ISignalDO An interrupt from a digital output signal

GettrapData in a trap routine to obtain all information about the interrupt that caused the trap routine to be executed.

IDisable Disable all interrupts

IError Order and enable an interrupt when an error occurs

ISignalAI An interrupt from an analog input signal

ISignalDI An interrupt from a digital input signal

ISignalGI An interrupt from a group of digital input signals

ISignalGO An interrupt from a group of digital output signals

ITimer A timed interrupt

RaiseToUser From a NOSTEPIN routine, the error is raised to the error handler at user level.

ISleep Deactivate an individual interrupt

IWatch Activate an individual interrupt

ReadErrData in a trap routine, to obtain numeric information (domain, type and number) about an error, a state change, or a warning, that caused the trap routine to be executed.

Error Rec. Käskyt

BookErrNo Book a new RAPID system error number.

ErrRaise Create an error in the program and then call the error handler of the routine

ProcerrRecovery Generate process error during robot movement.

ResetRetrycount Reset the number of counted retries.

Return Return to the routine that called the current routine

Trynext Execute the instruction following the instruction that caused the error

ErrLog Display an error message on the teach pendant and write it in the robot message log.

Exit Stop program execution in the event of a fatal error

Raise Call the error handler of the routine that called the current routine

Retry Re-execute the instruction that caused the error

Skipwarn Skip the latest requested warning message.

System&Time Käskyt

ClkReset Reset a clock used for timing

ClkStop Stop a clock used for timing

MakeDir Create a new directory.

ReadCfgData Read one attribute of a named system parameter.

RemoveFile Remove a file.

WriteCfgData Write one attribute of a named system parameter.

ClkStart Start a clock used for timing

CloseDir Close a directory in balance with *OpenDir*.

OpenDir Open a directory for further investigation.

RemoveDir Remove a directory.

RenameFile Rename a file.

Matematics Käskyt

:= Perform calculations on any type of data

BitClear Clear a specified bit in a defined *byte* data.

Clear Clear the value

Incr Increment by 1

Add Add or subtract a value

BitSet Set a specified bit to 1 in a defined *byte* data.

Decr Decrement by 1

TryInt ’ Test if data object is a valid integer

MotionSetAdv Käskyt

CirPathMode Choose the way the tool reorientates during circular interpolation

PathResol Adjust the geometric path resolution

TuneReset Reset tuning to normal

WZBoxDef Define a box-shaped global zone

WZDisable Deactivate supervision of a temporary global zone

PathAccLim Set or reset limitations on TCP acceleration and/or TCP deceleration along the movement path.

SpeedRefresh Update speed override for ongoing movement

TuneServo Adjust the robot tuning values

WorldAccLim Limiting the acceleration/deceleration of the tool (and griplоad) in the world coordinate system.

WZClyDef Define a cylindrical global zone

WZDoSet Activate global zone to set digital outputs

WZEnable Activate supervision of a temporary global zone

WZHomeJointDef Define a global zone in joints coordinates

WZLimSup Activate limit supervision for a global zone

WZFree Erase supervision of a temporary global zone

WZLimJointDef Define a global zone in joints coordinates for limitation of working area.

WZSphDef Define a spherical global zone

MotionAdv Käskyt

ClearPath Clear the whole motion path on the current motion path level.

MoveJSync Moves the robot by joint movement and executes a RAPID procedure

RestoPath Regenerate a path stored earlier

StartMoveRetry Restart the robot movements and make a retry in one indivisible sequence

StopMove Stop the robot movements

StorePath Store the last path generated

MoveCSync Moves the robot circularly and executes a RAPID procedure

MoveLSync Moves the robot linearly and executes a RAPID procedure

StartMove Restart the robot movements

StepBwdPath Move backwards on its path in a RESTART event routine

StopMoveReset Reset the stop move status, but don’t start the robot movements

TriggC Run the robot (TCP) circularly with an activated trigg condition

TriggCheckIO Define an IO check at a given position

TriggInt Define a trigg condition to execute a trap routine at a given position

TriggJ Run the robot axis-by-axis with an activated trigg condition

TriggSpeed Define conditions and actions for control of an analog output signal with output value proportional to the actual TCP speed.

TriggEquip Define a trigg condition to set an output at a given position with the possibility to include time compensation for the lag in the external equipment

TriggIO Define a trigg condition to set an output at a given position

TriggL Run the robot (TCP) linearly with an activated trigg condition

TriggRampAO Define a trigg condition to ramp up or down analog output signal at a given position with the possibility to include time compensation for the lag in the external equipment

TriggStopProc Create an internal supervision process in the system for zero setting of specified process signals and the generation of restart data in a specified persistent variable at every program stop (STOP) or emergency stop (QSTOP) in the system

MultiTaskin&WaitTestAndSet Käskyt

SyncMoveUndo Reset synchronized movements

WaitTestAndSet Retrieve exclusive right to specific RAPID code areas or system resources (type interrupt control)

RAPID Support Käskyt

GatDataVal Get a value from a data object that is specified with a string variable

SetDataSearch Together with *GetNextSym* data objects can be retrieved from the system.

SetSysData Activate a specified system data name for a specified data type.

WarmStart Restart the controller e.g. when you have changed system parameters from RAPID.

GetSysData Fetch data and name of current active Tool or Work Object.

SetAllDataVal Set a new value to all data objects of a certain type that match a given grammar.

SetDataVal	Set a value for a data object that is specified with a string variable
TextTabInstall	Install a text table in the system.

Calib&Service Käskyt

MToolRotCalib	Calibrate the rotation of a moving tool.
SPyStart	Start the recording of instruction and time data during execution.
SToolRotCalib	Calibrate the TCP and rotation of a stationary tool.
TestSignDefine	Define a test signal
MToolTCPCalib	Calibrate Tool Centre Point - TCP for a moving tool.
SPyStop	Stop the recording of time data during execution
SToolTCPCalib	Calibrate Tool Centre Point - TCP for a stationary tool
TestSignReset	Reset all test signals definitions

M.C.1Käskyt

MoveJ	Joint movement
MoveC	TCP moves along a circular path
Set	Set a digital output signal (to 1)
WaitTime	Wait a given amount of time or to wait until the robot stops moving
:=	Assingn a value to data
MoveL	TCP moves along a linear path
ProcCall	Call (jump to) another routine
Reset	Reset a digital output signal (to 0)
WaitDI	Wait until a digital input is set or reset
CompactIF	Execute one instruction only if a condition is satisfied
Incr	Increment by 1
Add	Add or subtract a value

Decr	Decrement by 1