Jeremias Walkeajärvi

Kobotin integrointi pakkaustyöhön

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Sähkö- ja automaatiotekniikka

Insinöörityö

1.5.2023

Tiivistelmä

Tekijä:	Jeremias Walkeajärvi
Otsikko:	Kobotin integrointi pakkaustyöhön
Sivumäärä:	27 sivua + 2 liitettä
Aika:	1.5.2023
Tutkinto:	Insinööri (AMK)
Tutkinto-ohjelma:	Sähkö- ja automaatiotekniikka
Ammatillinen pääaine:	Automaatiotekniikka
Ohjaajat:	Lehtori Timo Tuominen Kehitysjohtaja Anssi Tura

Insinöörityössä tutkittiin yhteistyörobotin eli kobotin integroinnin toteuttamiskelpoisuutta logistiikkakeskuksen pakkaussoluun. Kobotin integroinnin tavoite on vapauttaa toinen operaattori kahden operaattorin pakkaussolusta. Insinöörityössä tavoitteena oli purkaa pakkaustyö vaiheisiin ja selvittää, mitkä vaiheet pakkaustyöstä oli mahdollista suorittaa kobotilla. Insinöörityön tavoitteena oli myös luoda ohjeistus käytetystä kobotista ja pohtia sen kannattavuutta automatisoinnin selvitystyön perusteella.

Automatisoinnin selvitystyö tehtiin Metropolia Ammattikorkeakoulun omistamalla ABB YuMi IRB 14000 -kobotilla. Kobotille luotiin työasema ja sille ohjelmoitiin demo pakkaustyöstä. Selvitystyön perusteella luotiin pakkaussoluehdotus, jossa pakkausvaiheet on jaettu operaattorin ja kobotin kesken. Pakkaussoluehdotuksen kannattavuutta ja suorituskykyä tutkittiin. Insinöörityön aikana kertyneen osaamisen pohjalta luotiin ohjeistus asiakasyrityksen ja Metropolia Ammattikorkeakoulun käyttöön.

Automatisoinnin selvitystyössä kävi ilmi, että YuMi-kobotti on soveltuva suorittamaan vain pakkaustyön yksinkertaisempia vaiheita. Kobotti oli myös pakkaustyössä huomattavasti ihmistä hitaampi, jolloin paketin kokonaispaketointiaika kasvoi huomattavasti. Yleisesti ottaen kobotit ovat kuitenkin soveltuvia tutkittuun paketointityöhön, vaikka YuMi-kobotti saattaa ollakin liian kevyt valinta jatkuvaan tuotantoon. Kobotin ohjelman nopeutta olisi myös mahdollista nopeuttaa huomattavasti niin, ettei se olisi este kobotin kannattavuudelle. Loppupäätelmänä kuitenkin todettiin, että kobotin integrointi olisi kannattavaa vain, jos operaattorin työtaakkaa pystyttäisiin laskemaan enemmän kuin tämän selvitystyön perusteella pystytään.

Avainsanat: Kobotti, YuMi, Pakkaustyö

Abstract

Author:	Jeremias Walkeajärvi
Title:	Integrating a Cobot into Packaging Work
Number of Pages:	27 pages + 2 appendices
Date:	1 May 2023
Degree:	Bachelor of Engineering
Degree Programme:	Electrical and Automation Engineering
Professional Major:	Automation Technology
Supervisors:	Timo Tuominen, Senior Lecturer
	Anssi Tura, Development Director

The thesis work investigated viability of integrating a collaborative robot, or cobot, into a packaging cell of a logistics center. The purpose of integrating a cobot was to free one operator from the two-operator packaging cell. The objective of the thesis work was to break the packaging work down into phases and investigate which phases could be carried out by a cobot. Aim was also to create a guide for the cobot and examine its profitability based on the investigation.

The investigation of automating the phases was carried out using Metropolia UAS own ABB YuMi IRB 14000 -cobot. A workstation for the cobot was created, and a demo of packaging work was programmed for it. Based on the investigation, a packaging cell proposal was created, where packaging phases were shared between the operator and the cobot. The profitability and performance of the packaging cell were examined. With the know-how accumulated during the thesis work, a guide for the cobot was created for the use of commissioning company and Metropolia UAS.

The automation investigation concluded that YuMi-cobot was only capable of performing the simpler packaging phases. Cobot was also considerably slower than a human while performing the same packaging work. This resulted in a considerably longer total time to complete one package. However, it was agreed that cobots are suited to do packaging work in question, although the YuMi-cobot might be too lightweight choice when considering continuous production. It is also possible to speed up the cobot's program by such a margin that it would no longer pose an obstacle to profitability. However, the final conclusion is that integrating a cobot would only be profitable if the operator's workload could be reduced further than what this investigation was able to.

Keywords: Cobot, YuMi, Packaging Work

Sisällys

1 Johdanto 1

2 HUB logistics Finland Oy 2

3 Pakkaustyö 3

3.1 Pakkaustyöhön kuuluvat artikkelit 3
3.2 Laatikon ja paketin osat 5
3.3 Pakkaustyön vaiheet 6
3.4 Havaitut haasteet 7

4 Teoria 9

4.1 Mikä on Kobotti? 9
4.1.1 Ero perinteiseen robottiin 9
4.1.2 Lähestyttävyyden edellytykset 11
4.2 YuMi-kobotti 12
4.2.1 Rakenne 12
4.2.2 Pendantti 14
4.2.3 RobotStudio 15

5 Pakkaussoluehdotus 16

5.1 Pakkaussolu 16
5.2 Kobotin ohjelman kulku 17
5.3 Suorituskyky 19
5.4 Kannattavuus 20

6 Yhteenveto 22

Lähteet 23

Liitteet
Liite 1: Demo-ohjelman koodi
Liite 2: ABB YUMI IRB 14000 -pikakäyttöopas

1 Johdanto

Tässä insinöörityössä selvitetään ABB YuMi kobotin soveltuvuutta HUB logistic-sillä tehtävään pakkaustyöhön. Kobotin käyttöä pakkaustyöhön alettiin tutkimaan, sillä tällä hetkellä ihmisen tekemä pakkaustyö on monotonista ja puuduttavaa. Yhteistyörobotteja eli kobotteja käytetään juuri tämänkaltaisten ja toistuvien tehtävien automatisointiin. Kobotin integrointi pakkaustyöhön voisi tahdittaa työtä ja tehdä työstä dynaamisemman.

Insinöörityön tavoitteena on purkaa pakkaustyö vaiheisiin, joiden automatisointimahdollisuutta selvitetään. Tämä tehdään luomalla kobotille työasema, johon se ohjelmoidaan suorittamaan pakkaustyötä. Tuloksena on tarkoitus esitellä operaattorin sisältämä pakkaussoluehdotus, johon kobotti on integroitu. Pakkaussoluehdotus luodaan sen perusteella, mitkä vaiheet pakkaustyöstä määritellään kobotin mahdolliseksi suorittaa ja mitkä vaiheet lasketaan ihmisen suorittamiksi.

Insinöörityössä tutustutaan kobotteihin ja tutkitaan niiden eroja perinteisiin robotteihin. Pakkaustyön automatisoinnin selvitystyö tehtiin Metropolia Ammattikorkeakoulun omistamalla ABB YuMi IRB 14000 -kobotilla. Insinöörityössä tutustutaan ABB YuMi -kobottiin ja luodaan sille ohjeistus, joka käsittelee kyseisen laitteen ominaisuuksia ja operointia. Ohjeistuksen tarkoitus on antaa lukijalle kuva kobotista ja auttaa perustavanlaatuisen operoinnin hallitsemisessa.

Valmistuneen työn kokonaisuus luodaan helpottamaan toimeksiantajayrityksen päätöstä kobotin hankinnasta. Insinöörityössä käsitellään myös pakkaussoluehdotuksen suorituskykyä ja kannattavuutta verrattuna lähtötilanteeseen, jossa kaksi operaattoria pakkaa käsin paketteja.

2 HUB logistics Finland Oy

HUB logistics perustettiin vuonna 1992, ja sen liiketoiminta oli logistiikan kehittäminen tarjoamalla konsultointipalveluja suomalaisille logistiikka-alan yrityksille. Kertyneen kokemuksen avulla siirtyi HUB logistics lopulta itsekin tarjoamaan logistiikkapalveluita, ja pystyi siten tarjoamaan kokonaisvaltaisen sisälogistiikan hoidon asiakasyrityksilleen. 2000-luvulla HUB Logistics on ollut ulkoistettujen logistiikkapalveluiden erikoisosaaja, joka on tuonut logistiikan toimintamalleja maailmalta Suomeen. Nimellä HUB viitataan informaation, materiaalin ja pääoman keskittämistä yhteen keskittymään. Keskittämällä nämä osa-alueet voi HUB logistics hoitaa koko logistiikkaketjun alusta loppuun. Koko HUB logisticsin historian ajan on aktiivinen kehittäminen ollut tärkeää, ja tätä jatketaan tällä insinöörityölläkin. (1; 2.)

Koska HUB logistics tarjoaa kokonaisvaltaisia logistiikan ratkaisuja, on palvelulähtöisyys hyvin asiakaslähtöinen. LEAN-filosofian hallitsevat asiantuntijat työskentelevät läheisesti asiakkaiden kanssa, ja jo vuosien kokemuksella voivat ehdottaa parannuksia materiaalivirtojen tehostamiseen. HUB logistics onkin luonut maineen luotettavana ja arvostettuna kumppanina ja on nykyään johtava toimija omalla toimialallaan. Tämä asema on pystytty saavuttamaan, sillä HUB logisticsilla työskentelee motivoitunut ja vankan ammattitaidon omaava henkilökunta. Hyvä työilmapiiri ja asianmukaiset edellytykset työn tekemiseen ovat HUB logisticsin takaamia periaatteita. HUB logistics on sitoutunut vastuulliseen toimintaan ja haluaa tukea korkealaatuisia sekä kotimaisia palveluita. Riihimäellä ja Tampereella rakennetaan mittavia määriä puupakkauksia logistiikan tarpeisiin. Kaikki raakapuu on PEFC- tai FSC-merkittyä sekä on EUTR-asetusten mukaista. HUB logistics haluaa toimia myös ympäristövastuullisesti, ja Hakkilan logistiikkakeskus toimiikin jo 100-prosenttisesti aurinkoenergialla. (3; 4; 5.)

HUB logistics tarjoaa perinteisten varastopalveluiden lisäksi myös sisälogistiikan palveluita. Sisälogistiikalla tarkoitetaan asiakkaan omissa tiloissa tapahtuvaa logistiikkaa. Tämä on usein vastaanottamista, lähettämistä, varastointia tai kuljetuksia. Se voi olla myös jokin lisäarvopalvelu tai käsittää kaikki ne resurssit, joita logistiikkakeskuksen pyörittäminen ylipäätään vaatii. Yhdessä ulko- ja sisälogistiikan ulkoistamisen kautta voi asiakasyritys saada huomattavat vuosisäästöt ja useita muita ulkoistamisen hyötyjä. (6.)

HUB logistics tarjoaa myös laajan kirjon lisäarvopalveluita, joihin tämä insinöörityökin liittyy. HUB logisticsin palveluvarastoissa tehdään asiakasyrityksien tuotteille lisäpalveluja. Nämä voivat olla esimerkiksi pakkausta, testausta tai vaikkapa toimituksia asiakkaille ja palautusten käsittelyä. Tämän insinöörityön tapauksessa asiakasyrityksen eri toimitusketjuja tulevat tuotteet pakataan logistiikkakeskuksessa ja lähetetään yhtenä valmiina tuotteena eteenpäin. Näin toimitusketjut voidaan optimoida ilman, että asiakasyrityksellä tarvitsisi olla oma varasto pakkaustoimintaan. (7.)

3 Pakkaustyö

Pakkaustyössä lisätään Euroopassa valmistettu kaukosäädin Aasiassa valmistetun digiboksin laatikkoon ja nämä jatkopakataan asiakkaan brändiin kuuluvalla tavalla. Digiboksilaatikot saapuvat ruskeissa pahvilaatikoissa mukanaan ID-tarra, joka tulee poistaa laatikosta ja liimata ulkopaketin alakulmaan. Kaukosäätimet saapuvat muovisuojuksissa, joissa on mukana paristot. Paristot tulee taittaa pakkausvaiheessa kaukosäätimen alle ja asettaa kaukosäädin laatikossa olevaan upotukseen. Jatkopakkaustyöhön tulee hiha digiboksilaatikon päälle sekä pyöreä tarra, joka liimataan digiboksilaatikon kanteen. Valmis digiboksilaatikko asetetaan ulkopakettiin ja turvatäpän päälle liimataan sinetti.

3.1 Pakkaustyöhön kuuluvat artikkelit

Pakkaustyöhön tulee eri toimittajilta yhteensä seitsemän artikkelia. Kuvassa 1 näkyy pakkaustyöhön kuuluvat artikkelit. Ne ovat seuraavat:

1. Digiboksilaatikko. Ruskean digiboksilaatikon kannessa ja kannen sulkuläpässä ovat laskosläpät, jotka upottautuvat laatikossa oleviin rakoihin.

2. Kaukosäädin. Kaukosäädin on pakattu muoviin, jonka päädyssä on oma osa paristoille.

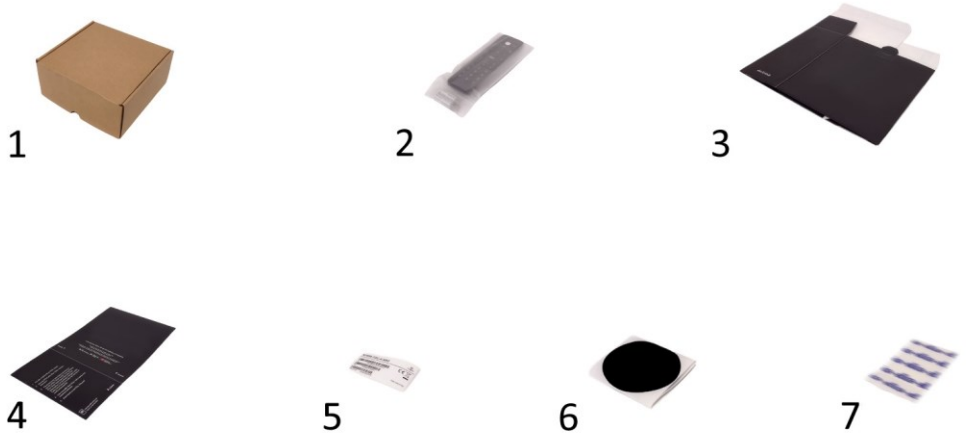
3. Ulkopaketti. Mustan ulkopaketin kannessa on sulkuläppä sekä paketin reunoilta laskeutuvat pölysuojaläpät sen alle. Kansi lukitaan kielityyppisellä turvaläpällä. Tulee kokoontaitettuna.

4. Hiha. Hiha on auki taitettava paperinen suojus, joka pujotetaan digiboksilaatikon päälle.

5. ID-tarra. ID-tarra on pienehkö tarra, joka sijaitsee kaukosäätimen paikalla digiboksilaatikossa.

6. Pyöreä tarra. Pyöreät tarrat tulevat tarranauhassa.

7. Sinettitarra. Sinettitarrat tulevat tarranauhassa.

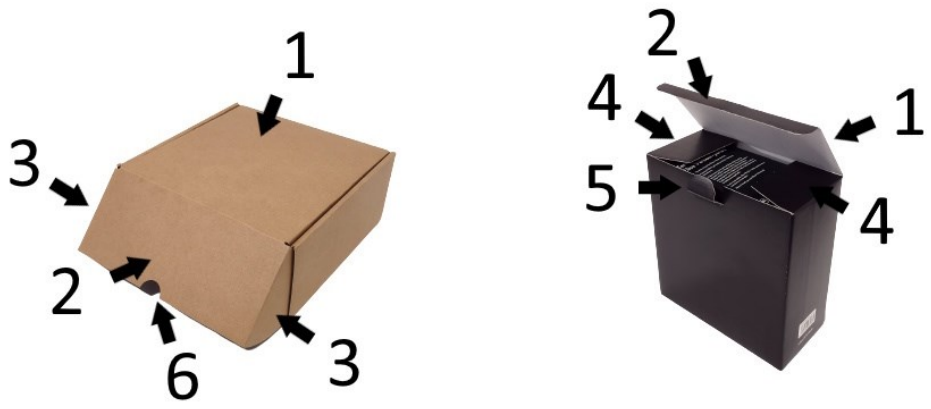


Kuva 1. Kuvassa seitsemän pakkaustyön artikkelia numeroituna.

3.2 Laatikon ja paketin osat

Digiboksilaatikon ja ulkopaketin pakkaustyötä kuvatessa käytetään siihen liittyvää termistöä. Termistö on listattu alta löytyvään listaan ja sen numeroita vastaavat kohdat on merkattu nuolilla kuvaan 2. Käyttäjän pitää huomioida, että digiboksilaatikon kannessa ovat myös laskosläpät, vaikka ne eivät näy kansi kiinni painettuna.

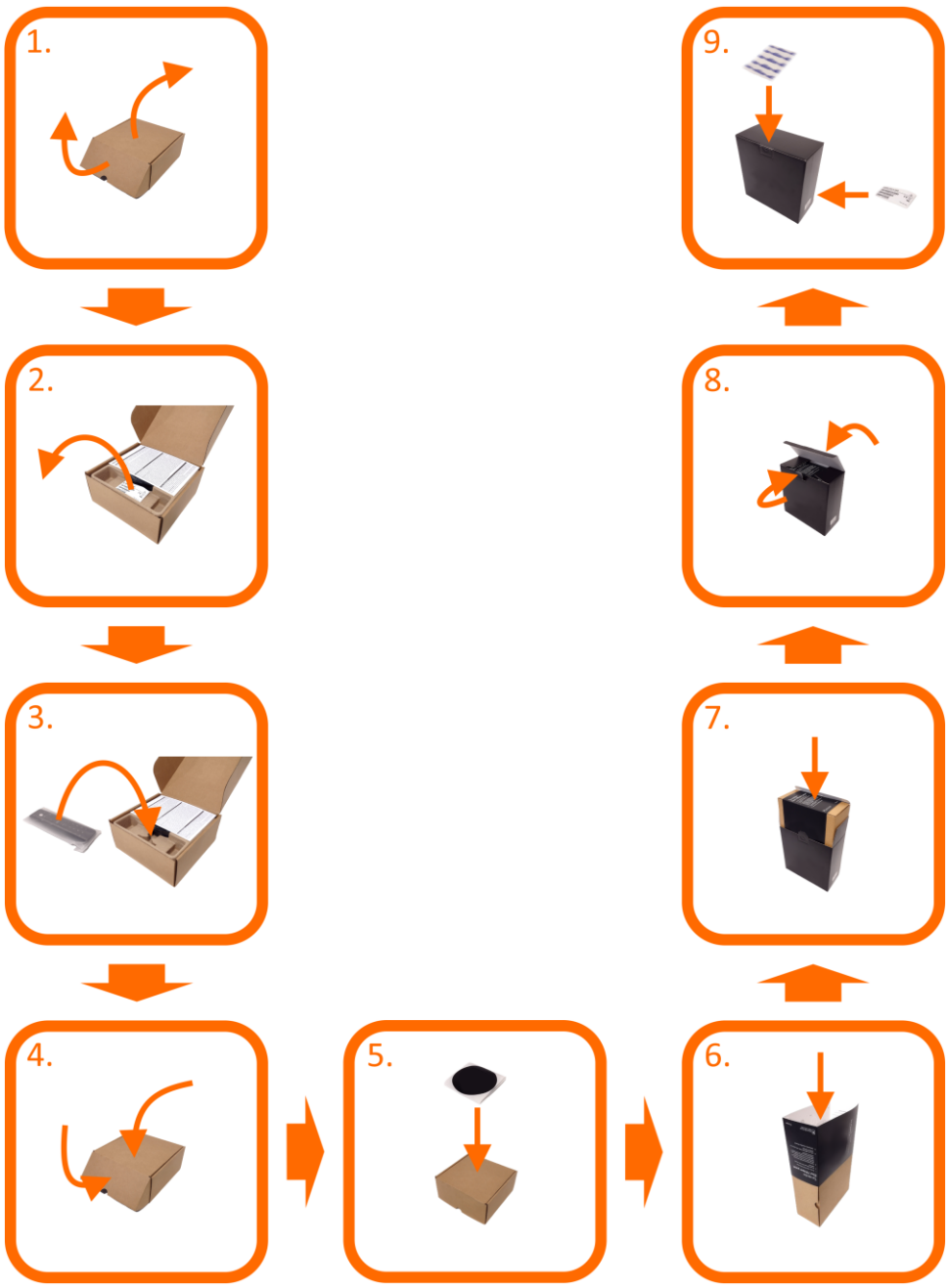
1. kansi
2. sulkuläppä
3. laskosläppä
4. pölyläppä
5. turvatäppä
6. avaamisen sormileikkaus.



Kuva 2. Kuvassa numeroitu laatikon ja paketin läppien ja kansien osat.

3.3 Pakkaustyön vaiheet

Kuvassa 3 on kuvattu pakkaustyön vaiheet. Lista vaiheista on koottu sivulle 7.



Kuva 3. Kuvassa on vuokaavio pakkaustyöstä, johon on numeroitu pakkausvaiheet ja kuvattu nuolilla suoritettavat liikkeet pakkausvaiheissa.

Pakkaustyönvaiheet 1 – 9:

1. Avataan digiboksilaatikon kansi.

2. Poimitaan ID-tarra digiboksilaatikosta ja asetetaan se sivuun.

3. Asetetaan kaukosäädin ja kaukosäätimen paristot alaspäin asetettuna digiboksilaatikossa olevaan upotukseen.

4. Suljetaan digiboksilaatikon kansi.

5. Liimataan pyöreä tarra kanteen.

6. Käännetään digiboksilaatikko sivulleen ja pujotetaan hiha laatikon päälle.

7. Asetetaan digiboksilaatikko ulkopaketin sisälle.

8. Suljetaan ulkopaketin kansi ja laitetaan turvatäppä kiinni.

9. Liimataan sinettitarra turvatäpän päälle ja ID-tarra oikeaan alakulmaan.

3.4 Havaitut haasteet

Digiboksilaatikon laskosläpät pitävät kannen tiukasti kiinni. Tästä seuraa, että laatikkoa on pidettävä kiinni samalla, kun kantta avataan. Sulkuläpän avaamisen sormileikkaus sijaitsee kobotin tarttujalle haastavassa paikassa, aivan pöytätasoa vasten. Kantta suljettaessa on laskosläpät aseteltava tarkkuutta käyttäen niille tarkoitettuihin rakoihin. Kun kansi suljetaan, sen sulkuläppä pyrkii sulkeutumaan itsestään, ja laskosläpät levittäytyvät paketin sivuille.

Kaukosäätimen muovisuojus on löysähkö. Näin ollen kaukosäätimen mitat muovisuojuksineen ovat suuremmat kuin pelkän kaukosäätimen. Tästä seuraa, että kaukosäätimen paikka muovisuojuksen sisällä ei ole vakio, joten se on asetettava operaattorin toimesta standardoidulla tavalla. Aseteltaessa kaukosäädintä

laatikon upotukseen on paristot piilotettava ensin sen alle. Muutoin paristot eivät asetu kunnolla niille tarkoitettuun upotukseen eivätkä salli kaukosäätimen asettua tasaisesti omaan upotukseensa. Kaukosäätimen upotus digiboksilaatikossa on tiukka, joten kaukosäätimen laskettua on se vielä painettava upotukseen, jotta se istuu samassa tasossa laatikon kanssa ja kansi pääsee sulkeutumaan täysin kiinni.

Hiha istuu erittäin tiiviisti digiboksilaatikon päälle. Laskettaessa hihaa digiboksilaatikon päälle on sitä pidettävä pingotuksessa, koska kaikkien sivujen on tultava digiboksilaatikon päälle lähes samaan aikaan. Tämä työvaihe vaatii erityistä tarkkuutta ihmiseltäkin.

Ulkopaketin kansi pyrkii sulkeutumaan itsestään, joten sitä on pidettävä auki asetettaessa digiboksilaatikkoa ulkopakettiin. Kantta on myös pidettävä aktiivisesti auki, jotta molemmat pölyläpät voidaan sulkea. Pölyläpät pyrkivät myös aukeamaan, joten niitä on pidettävä aktiivisesti suljettuina, jotta kansi voidaan sulkea. Kielityyppisen lukon sulkeminen vaatii huomattavaa sorminäppäryyttä.

ID-tarra on melkein kaukosäätimen upotuksen levyinen, joten tarttujan sormet eivät mahdu poimimaan sitä poikittaissuunnassa. Se lepää pitkittäissuunnassa ja kuperasti upotuksen pohjaa vasten, jolloin siihen on haastavaa tarttua pitkittäissuunnassa. ID-tarraa koskeva vaihe olisi todennäköisesti helppo automatisoida käyttämällä vakuumi-imukuppitoimintoa, mutta tämän toiminnon soveltuvuutta ei päästä selvittämään Metropolian YuMi-kobotilla.

Pyöreä tarra on suurehko, ja sen liimaaminen onnistuneesti on haastavaa tartuttaessa sitä reunasta.

Sinetti tarra asetetaan ulkopaketin kannen sulkevaan kielisulkijaan, jolloin liimapinta jakaantuu kahdelle sivulle. Tarraa liimattaessa on liimattava yksi sivu kerrallaan, koska tarrasta ei voida irrottautua ennen sen liimapinnan painamista laatikkoa vasten erillisellä työvaiheella.

4 Teoria

4.1 Mikä on Kobotti?

Ensimmäinen yhteistyörobotti eli kobotti tuli markkinoille vuonna 2008 tanskalaisen Universal Robots -yrityksen lanseeraamana. Universal Robots onkin yhä johtava kobottivalmistaja. Kobotit vastasivat tarpeeseen madaltaa kynnystä automatisointiin niin kannattavuuden kuin lähestyttävyydenkin kannalta. Yhä nouseva automaatioaste työpaikoilla tarkoittaa tulevaisuudessa sitä, että ihmiset ja robotit tulevat väistämättä toimimaan tiiviimmin yhdessä. Tähän ihmisen ja robotin yhteistyöhön kobotit on suunniteltu, ja vain kobottien tarjoamien ominaisuuksien avulla ihmisen ja robotien saumaton yhteistyö voidaan saavuttaa. (8.)

4.1.1 Ero perinteiseen robottiin

Eroja perinteisiin robotteihin on useita ja niitä on verrattu taulukossa 1. Perinteiset robotit ovat ensinnäkin painavia ja ne pultataan lattiaan kiinni. Näiden robottien turvallinen käyttö edellyttää turvalaitteita kuten valoverhoja ja turvahäkkejä. Mitä läheisemmin ihmiset työskentelevät näiden perinteisten robottien kanssa, sitä monimutkaisemmaksi turvalaitteiden verkko myös muodostuu. Kobotit ovat sen sijaan kevyitä, ja ne voidaan siirtää uuteen työpisteeseen ilman, että työympäristö pitäisi aina rakentaa niiden ympärille. Kobottien ohjelmointi on myös suunniteltu helpoksi, jotta työtehtävää voidaan vaihtaa tarpeen mukaan. Koska ohjelmointi on kallista, perinteiset robotit suorittavat tyypillisesti samaa tehtävää vuosikausia. (9, s. 13–14; 10, s. 2–5; 11, s. 17.)

Vaikka pelkät kobotit ovat usein kalliimpia verrattaessa vastaavaan perinteiseen robottiin ovat niiden hankinnan kokonaiskustannukset huomattavasti halvemmat. Tämä selittyy perinteisten robottien ohjelmoinnin haastavuudella, joka vaatii aina koulutetun ammattilaisen. Perinteisten robottien hankintahintaa nostavat myös turvalaitteet, joiden pitää täyttää standardit, ja ne on hyväksytettävä viranomaisilla. Kobottien edullisempi hankintakustannus ja joustavuus työtehtävien suhteen tarjoaa niille erinomaisen kannattavuuden. Tähän kannattavuuteen

pystyvät tarttumaan myös pienet ja keskisuuret yritykset, joille perinteinen robotisointi olisi liian kallista. Kannattavuuden ja ohjelmoinnin lähestyttävyyden ansiosta ne soveltuvatkin hyvin ensimmäiseksi askeleeksi yrityksen toiminnan automatisoinnissa. (9, s. 15–16; 10, s. 5; 11, s. 17.)

Koboteille tyypillisiä tehtäviä ovat teollisuuden laitteen käyttö tehtävät ja pick-and-place-tyyppiset tehtävät. Konenäköä hyödyntäen kobotti soveltuu myös erilaisiin tarkastus- ja laadunvalvontatehtäviin. Kobottien liikkeet ovat tyypillisesti hitaampia, ja kantokyky on pienempi kuin perinteisillä roboteilla. Tämä johtuu kobottien kevyemmästä rakenteesta, joka juontuu siitä, että kobotit on suunniteltu lähtökohtaisesti turvalliseksi. Kevyemmästä rakenteesta ja hitaammista liikkeista johtuenkin kobotit eivät välttämättä olekaan järkevä vaihtoehto korkean volyymin työtehtävään, jossa robotin elinikä ja huoltovarmuus ovat tärkeitä tekijöitä. Kobotit eivät myöskään sovellu erityisen fyysisiin tehtäviin tai tehtäviin, joissa saattaa kohdistua kovia iskuja robotin rakenteeseen. (9, s. 16; 10, s. 5.)

Taulukko 1. Taulukossa verrataan kobotin ja perinteisen robotin ominaisuuksia.

Attribuutti	**Kobotti**	**Perinteinen robotti**
Hankintakustannus	Edullisempi	Kalliimpi
Lähestyttävyys	Helppo ohjelmoida	Haastava ohjelmoida
Turvallisuus	Luonnostaan turvallinen	Vaatii erilliset turvalaitteet
Työympäristö	Voi toimia jaetussa työtilassa	Vaatii oman työtilan
Liikuteltavuus	Voidaan siirtää työpisteestä toiseen	Pultattu lattiaan
Nopeus	Hitaammat liikkeet	Nopeat liikkeet
Paino	Kevyempi rakenne	Raskaampi rakenne
Kantokyky	Pienempi kantokyky	Suurempi kantokyky
Elinikä	Mahdollisesti lyhyempi	Kestävämpi rakenne

4.1.2 Lähestyttävyyden edellytykset

Jotta robotti on lähestyttävä käyttäjälle, on sen ennen kaikkea oltava turvallinen. Kobotit on suunniteltu siten että törmätessäänkin niiden ei tule aiheuttaa vakavaa vahinkoa itselle tai ihmiselle. Tämän takia kobottien rakenne on kevyt ja ulkokuori muovia. Ulkokuoreen on myös lisätty pehmustepintoja ja sen reunat pyöristetty. Kobotin pääyksikkö tarkkailee jatkuvasti nivelissään olevien moottorien käyttämiä voimia. Jos voima nousee yli sallitun arvon, tulkitsee pääyksikkö tämän törmäykseksi ja pysäyttää liikkeen. Kobotit voidaan myös varustaa sensoreilla, joilla ne havaitsevat, jos määritellylle turva-alueelle astuu ihminen. Tällöin ne voivat pysäyttää liikkeen, ja ihminen voi joustavasti käydä suorittamassa tehtävänsä. Turva-alue voi käsittää myös varoalueen, jolloin esimerkiksi ahtaassa tehtaassa ihminen voi oleskella varoalueella niin, että kobotti voi jatkaa tehtäviään alennetuilla nopeuksilla. Kobotin turvallisuutta käsiteltäessä on muistettava työtoimenpiteen luontainen vaarallisuus. Jos kobotilla on terävä tai muuten vaarallinen työkalu, ei kobotin törmäysturvallisuudesta ole hyötyä, ja kobotti voidaankin joutua sulkemaan turvahäkkiin. (9; 10, s. 5; 12, s. 20.)

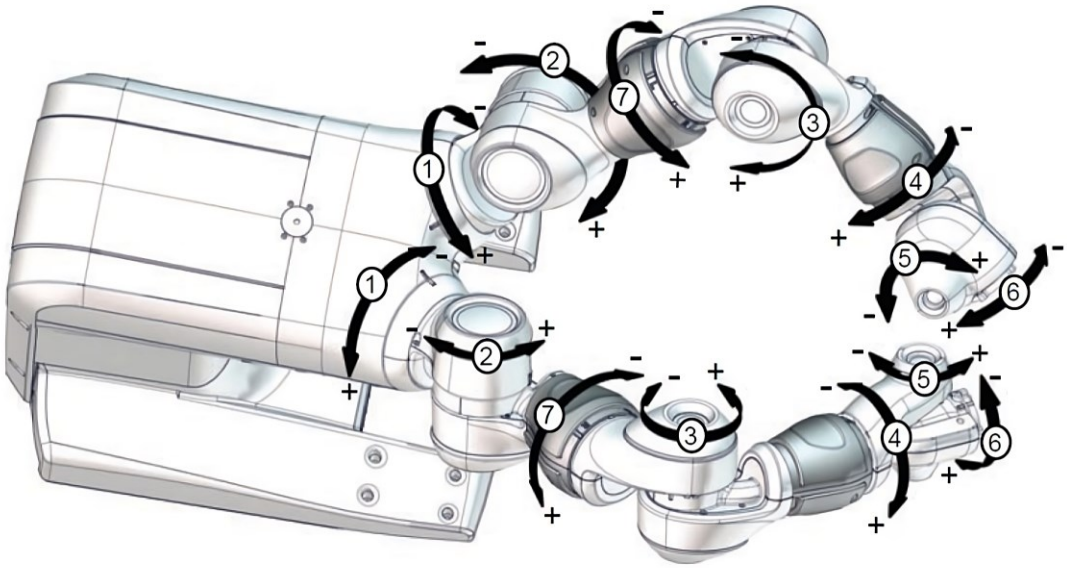
Kobottien ohjelmointi on suunniteltu helpoksi ja sen pitäisi olla mahdollista ilman aikaisempaa ohjelmointikokemusta. Ohjelmointi tapahtuu kobottien johdolla kiinnitettävän operointipäätteen kautta, jota kutsutaan pendantiksi. Pendantin kautta käytetään laajahkoa käyttöliittymää, joka sisältää muun muassa tiedostonhallintaohjelman ja ohjelmointisovelluksen. Tämän käyttöliittymän avulla voidaan kobottia käyttää ilman sen liittämistä koskaan erilliseen tietokoneeseen. Usein kobotit ohjelmoidaan ja niin tässä insinöörityössäkin käytetyllä käsi opastus -menetelmällä. Kyseisessä menetelmässä vapautetaan kobotin varret vapaasti liikuteltavaksi, jolloin kobotti voidaan liikuttaa käsin haluttuun paikkaan ja asentoon. Tällä tavalla voidaan tallentaa reitti ja työkalun toimenpiteet, jotka kobotti osaa suorittaa automaattisesti. Pendantti ei välttämättä aina ole paras ohjelmointityökalu ja tämäntyyppisellä ohjelmoinnilla onkin katto tehtävän monimutkaisuudessa. (8; 9, s. 15.)

4.2 YuMi-kobotti

Tähän insinöörityöhön liittyen käsittelen ABB:n kahdella 7-akselisella kädellä varustettua YuMi IRB 14000 -kobottia. ABB suunnitteli YuMi-kobotit erityisesti Aasian 3C-teollisuuden valmistustöihin (13). 3C tulee sanoista tietokoneet, kommunikaatio ja kuluttajaelektroniikka. Alaa leimaavat monotoniset ja toistuvat toimenpiteet sekä tiuha vaihtelu tuotteiden välillä (14). Juuri kyseiseen ja moneen muuhun kevyeen tuotantoon ABB:n kobotit soveltuvat erinomaisesti.

4.2.1 Rakenne

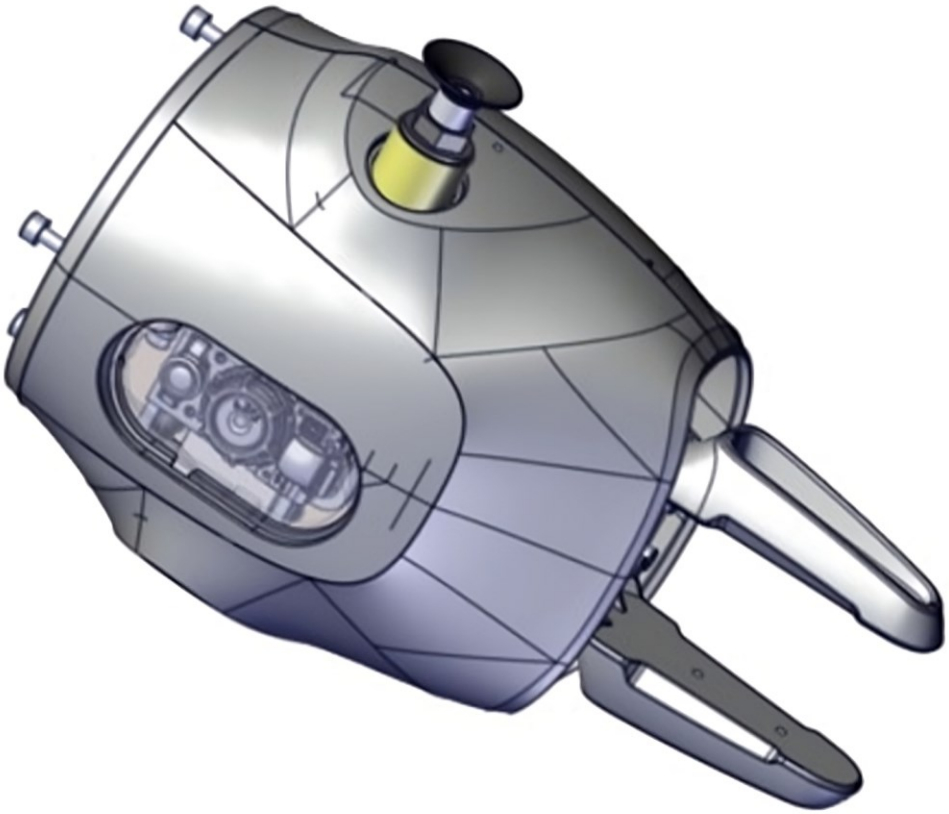
Kuvassa 4 näemme insinöörityössä käytetyn kobotin, johon on merkattu numeroilla käsien seitsemän akselia. Akselit 1 ja 2 ovat käsivarren juuren pyörimis- ja taivutusakselit. Akselit 3 ja 7 ovat kyynärvarren vastaavat pyörimis- ja taivutusakselit. Akselit 4 ja 5 ovat ranteen pyörimis- ja taivutusakselit. Akseli 6 on työkalun, eli YuMin tapauksessa tarttujan pyörimisakseli. Seitsemättä akselia ei ole juuresta laskien nimetty järjestyksessä, koska se puuttuu perinteisistä 6-akselisista koboteista. (15, s. 46.)



Kuva 4. Kuvassa numeroitu kobotin käsien seitsemän liikkuvaa akselia (16, s. 17).

Kobotin varret on pehmustettu ja koko laitteisto on IP30-suojattu. Kädet kiinnittyvät integroituun ohjausyksikköön, joka sisältää kobottia ohjaavan tietokoneen. Ohjausyksikössä sijaitsevat myös kobotin digitaaliset ja analogiset rajapinnat. Ohjausyksikkö käsineen painaa 38 kg, ja se asennetaan pöydän reunaan tai tasaiselle pinnalle. Pelkän ohjausyksikön pinta-ala on 339 x 497 mm ja kädet yltävät uloimmillaan 559 mm:n etäisyyteen. (17, s. 9–17.)

Kuvassa 5 on YuMin työkalu, eli SmartGripper. SmartGripper on servoilla ohjattu kaksihampainen tarttuja. Tarttuja voidaan myös konfiguroida vakuumiominaisuudella ja konenäöllä. Vakuumi-imukupit sijaitsevat tarttujan sivuilla. Konenäkömoduuli sijaitsee tarttujan alapuolella tarttujan runkoon integroituna. (15, s. 55–56.)



Kuva 5. Kuvassa tarttuja vakuumi-imukupilla ja konenäöllä varustettuna (15, s. 61).

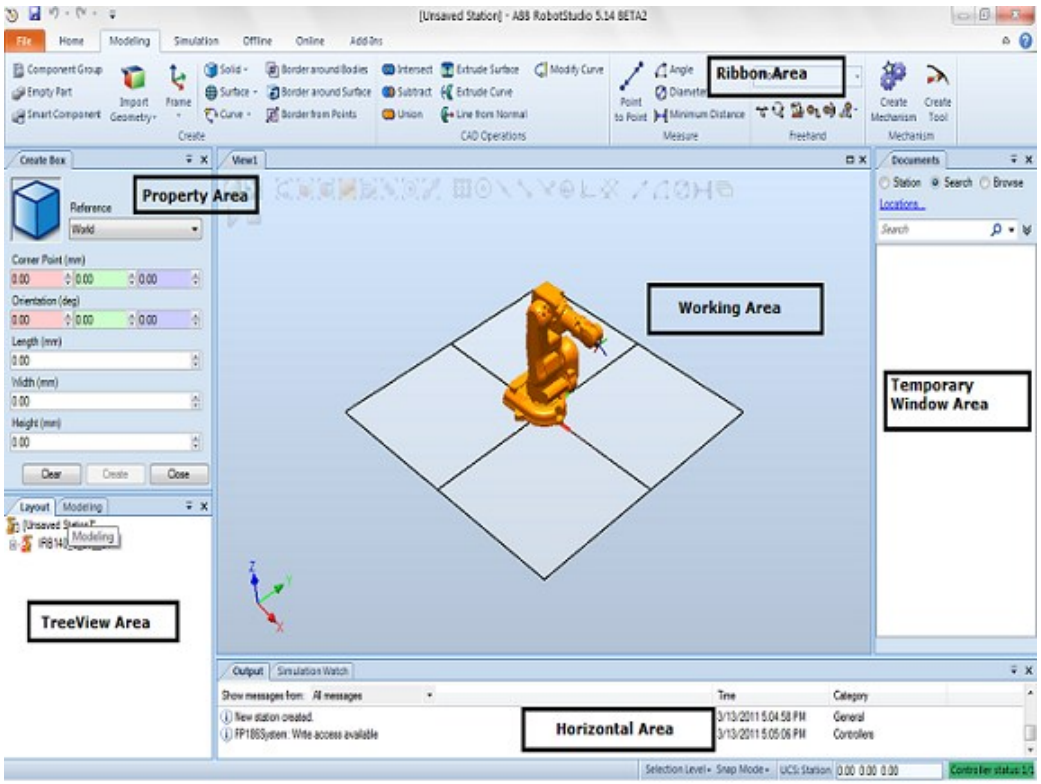
4.2.2 Pendantti

FlexPendant on ABB:n kannettava operointipääte, joka liitetään ohjausyksikköön johdolla. Kuvassa 6 voimme nähdä pendantin, joka on tietokone itsessään. Sen kautta saadaan kokonaisvaltainen kontrolli kobotista. FlexPendanttiin on asennettu RobotWare-käyttöliittymä, johon asennetaan ohjelmia tarpeen mukaan. Pendantista löytyvät painikkeet ohjelman suorittamiseen ja neljä ohjelmoitavaa painiketta. Manuaalitilassa pendantin painikkeilla voidaan myös valita haluttu akseli tai liikkumistapa, jolloin sitä voidaan ohjata joystickillä. Ohjelmointi tapahtuu ajamalla kobotti haluttuun paikkaan ja lisäämällä ohjelmaan liikkumiskomento. Tällöin ohjelmaa suoritettaessa kobotti löytää itse reitin ohjelmoituihin paikkoihin. On myös lähestyttävämpi lead-through-ohjelmointitoiminto. Kuvassa 6 näkyvää Enable Lead-through -näppäintä painettaessa YuMin käsi vapautuu, ja käsin liikuttamalla voidaan viedä se haluttuun paikkaan. Näin ohjelmointi nopeutuu, kun käytetään apuna luojan luomaa kättä kobotin nopeaan ja tarkkaan liikuttamiseen. (18, s. 37–39; 16, s. 24.)

Kuva 6. Kuvassa pendantti, jossa on ohjaamiseen tarkoitettu ohjelma auki.

4.2.3 RobotStudio

RobotStudio on ABB:n robottien ohjelmointi- ja konfigurointityökalu. RobotStudio-ohjelmalla on mahdollista ottaa yhteys kobotin ohjausyksikköön. Yhteyden muodostaminen tapahtuu liittämällä ohjausyksikkö tietokoneeseen Ethernet-kaapelilla. Kuvassa 7 on ruudunkaappaus RobotStudion käyttöliittymästä, jonka avulla on mahdollista ohjelmoida robotti virtuaalisesti ja ladata simuloitu ohjelma robottiin. RobotStudion tärkeimmät ominaisuudet on pääsy kobotin tiedostonhallintaohjelmaan ja RAPID Editor -ohjelmointityökalu, joka on oivallinen työkalu kehittyneessä ohjelmoinnissa pendantin ohella. RobotStudio mahdollistaa myös ohjelmien lisäämisen ja päivittämisen pendantissa. FlexPendant SDK -ohjelmistokehityspaketin lataamalla on myös mahdollista luoda omia ohjelmia pendanttiin. ABB tarjoaa myös Windows 10 -tableteille suunnatun RobotStudio Online -ohjelmiston, jolla on mahdollista kontrolloida robotteja langattomasti, esimerkiksi tehtaassa. (18, s. 45–47; 19, s. 1.)

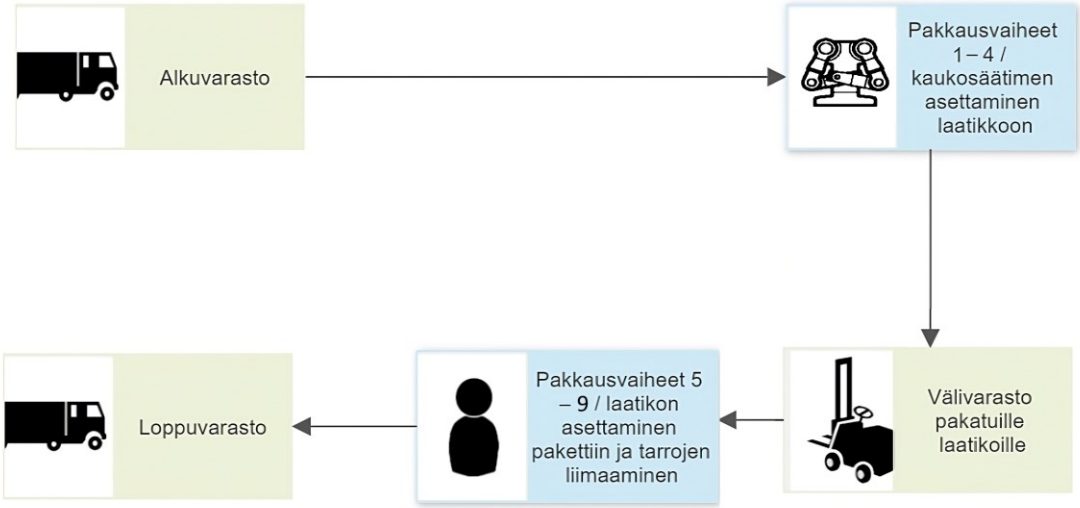


Kuva 7. Kuvassa RobotStudion käyttöliittymä, johon on nimetty englannin kielellä käyttöliittymän rakenteen osia (20).

5 Pakkaussoluehdotus

5.1 Pakkaussolu

Pakkaustyönvaiheista vaiheet 1 – 4 määriteltiin automatisoitavaksi kobotilla ja loput 5 – 9 vaiheet määriteltiin operaattorin suorittamiksi. Kuvassa 8 voimme nähdä pakkaussolun, joka sisältää kaksi työasemaa. Näiden välissä sijaitsee välivarasto. Ensimmäisessä työasemassa kobotti avaa laatikon, poistaa ID-tarran ja asettaa kaukosäätimen laatikkoon. Toisessa työasemassa operaattori asettaa laatikon ulkopakettiin ja liimaa tarrat niille kuuluville paikoille.



Kuva 8. Kuvassa pakkaussolun vuokaavio, jossa sinisillä lohkoilla suorittavat toimenpiteet ja vihreällä varastolohkot.

Koska pakkausvaihe kaksi (ID-tarran poistaminen laatikosta) suorittaminen kobotilla on olennaista pakkaussolun tehokkuuden kannalta, on tässä pakkaussolu ehdotuksessa pakkausvaihe kaksi automatisoitu. On perusteltua olettaa, että pakkausvaiheen kaksi automatisointi onnistuisi kobotin vakuumi-imukuppitoiminnolla, vaikka sitä ei ole päästy käytännössä selvittämään eikä sille löydy aliohjelmaa demo-ohjelman koodissa (liite 1).

5.2 Kobotin ohjelman kulku

Seuraavana on ehdotusohjelman main-ohjelman koodi, jossa käydään läpi aliohjelmia:

```
PROC main()
     WaitSyncTask vaihe1, tasklist;
     laatikontuominen;
     WaitSyncTask vaihe2, tasklist;
     läpänavaaminen;
     WaitSyncTask vaihe3, tasklist;
     kallistaminen;
     WaitSyncTask vaihe4, tasklist;
     asettelu;
     WaitSyncTask vaihe5, tasklist;
     kannenavaaminen;
     WaitSyncTask vaihe6, tasklist;
     idtarra;
     WaitSyncTask vaihe7, tasklist;
     kaukosäädin;
     WaitSyncTask vaihe8, tasklist;
     kannensulkeminen;
     WaitSyncTask vaihe9, tasklist;
     laatikonvieminen;
ENDPROC
```

Main-ohjelma on identtinen molemmilla käsillä, ja niiden eteneminen synkronoidaan jokaisen vaiheen alussa. Ohjelman yhdeksän vaihetta ovat seuraavat:

1. Laatikon noutaminen varastosta. Kobotti siirtää pystyasennossa olevan laatikon varastosta työskentelyalueelle. Siirtoliikkeen lopuksi kobotti tarkistaa, että laatikko sijaitsee tarkasti halutussa paikassa läpän avaamista varten.

2. Läpän avaaminen. Laatikon läpän avaaminen tapahtuu asettamalla tarttujan sormi läpän alle ja nostamalla se ylöspäin. Oikea käsi tukee laatikkoa takaa, jottei se kallistu taaksepäin läppää avattaessa.

3. Kallistaminen. Laatikko on kallistettava vaaka-asentoon, jotta kansi voidaan avata. Kallistaminen tapahtuu tarttumalla läppään vasemmalla kädellä ja tuomalla sitä sisään ja ylöspäin. Samalla oikea käsi painaa laatikon alareunaa vastakkaiseen suuntaan, jolloin laatikon painopiste siirtyy vaakatasoon.

4. Asettelu työskentelyalueeseen. Myöhemmät vaiheet vaativat enemmän työskentely tilaa, joten on tarpeen siirtää laatikko kauemmas. Kuitenkin vaiheen aluksi raotetaan kantta seuraavaa vaihetta varten. Tämä tapahtuu nostamalla vaakasuorassa olevan laatikon kantta samalla kun oikea käsi tukee sitä reunasta. Itse siirtoliike tapahtuu työntämällä laatikkoa sisäsivulta ja tarkistamalla siirtoliikkeen lopuksi molemmilta sivuilta, että laatikko sijaitsee tarkasti halutussa paikassa.

5. Kannen avaaminen. Kannen avaaminen tapahtuu sijoittamalla suljettu tarttuja kannen ja laatikon välissä olevaan rakoon ja avaamalla tarttuja. Tämän jälkeen tarttuja siirtyy vaakatasoon kantta nähden ja työntää sen kokonaan auki. Oikea käsi tukee samalla laatikkoa, jottei se nousisi ilmaan. Lopuksi vasen käsi tarkistaa, että laatikko sijaitsee yhä tarkasti työskentelyalueella.

6. ID-tarran poistaminen. ID-tarraan tartutaan vakuumi-imukupilla ja se asetetaan sivuun.

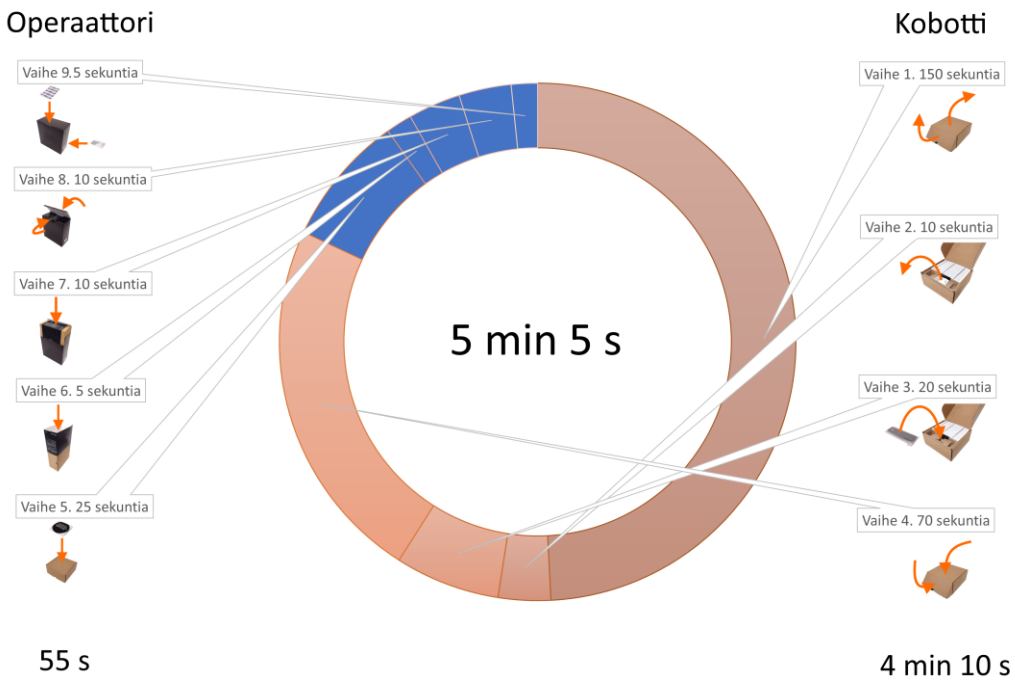
7. Kaukosäätimen asettaminen laatikkoon. Kobotti tarttuu varastossa olevaan kaukosäätimeen, niin että paristot roikkuvat alapuolella ja siirtää sen laatikon upotukseen. Lopuksi kobotti painaa kaukosäädintä upotukseen, jotta kansi menee hyvin kiinni.

8. Kannen sulkeminen. Kannen sulkeminen tapahtuu painamalla molempia kannen laskosläppiä sisäänpäin, jolloin ne uppoutuvat rakoihinsa. Tämän jälkeen vasen käsi painaa kannen tiukasti kiinni. Sulkuläpän laskosläppien sijoittaminen rakoihinsa tapahtuu siirtämällä ne samaan tasoon laatikon reunan kanssa ja painamalla tämän jälkeen niitä sisäänpäin. Sulkuläpän laskosläpät uppoutuvat samalla tavalla rakoihinsa kuten kannenkin. Lopuksi vasen käsi painaa sulkuläpän tiukasti kiinni.

9. Laatikon vieminen varastoon. Vasen käsi nostaa sivussa olevan ID-tarran laatikon päälle. Oikea käsi siirtää tämän jälkeen laatikon varastoon.

5.3 Suorituskyky

Kuvassa 9 näemme pakkaussolun vaiheiden viemän ajan kokonaisajan ollessa 5 minuuttia ja 5 sekuntia. Tämä tarkoittaa sitä, että pakkaussolu pystyy tuottamaan 11 valmista pakettia tunnissa. Jos kobotilla on riittävä varasto laatikoita ja kaukosäätimiä, voi se tuottaa 14 paketoitua laatikkoa tunnissa. Operaattori voi suorittaa vaiheet 5 – 9 55 sekunnissa, joka vastaa 65 paketin tuntitehokkuutta. Näistä luvuista voimme päätellä kobotin olevan selkeä pullonkaula pakkaussolussa. Operaattorille on töitä vain noin 18 % ajasta. Jos operaattori paketoisi vain kobotin pakkaamia laatikoita valmistuisi seitsemän tunnin työnteon aikana 100 pakettia. Jos operaattori paketoi jäljelle jäävän 82 % ajasta omia paketteja, paketoituisi seitsemän tunnin työn aikana 381 pakettia.



Kuva 9. Kuvassa ympyräkaavio pakkaussolun työtä suorittavista vaiheista, johon on merkattu kunkin vaiheen viemä aika sekunteina. Kobotin suorittamat vaiheet on maalattu oranssilla ja operaattorin sinisellä.

Suorituskykyä voidaan siis kasvattaa huomattavasti, kun käytetään välivarastoa, joka mahdollistaisi kobotin käymisen ympäri vuorokauden. Jos kobotin halutaan pakkaavan maksimaalisen määrän paketteja eli ympäri vuorokauden,

pitäisi sillä olla 237 paketin varasto operaattorin työpäivän päätyttyä. 237 laatikon jatkopaketoiminen vie yhdeltä operaattorilta 3 tuntia ja 37 minuuttia. Kun mukaan lasketaan kobotin työpäivän aikana käsittelemät paketit, tulee operaattorin töiden pituudeksi 5 tuntia ja 17 minuuttia. Yhden päivän aikana kobotti voi siis pakata yhteensä 345 pakettia, joka vastaa operaattorin näkökulmasta 65 paketin tuntitehokkuutta.

Jos kobotti valmistaa paketteja ympäri vuorokauden, ja operaattori paketoi paketteja ylimääräisen ajan, pystytään kasvattamaan vuorokaudessa paketoitujen pakettien määrää. Olettaen operaattorille 7 tuntia tehokasta työaikaa, pystyy paketointisolu yhteensä tuottamaan 432 pakettia päivässä. Tämä vastaa 61 paketin tuntitehokkuutta.

Suorituskykyä voidaan kuitenkin kasvattaa optimoimalla ohjelmaa. Ehdotusohjelma toimii 15 %:lla YuMin maksiminopeudesta. Jos ehdotusohjelmaa voidaan nopeuttaa edes kolmanneksella, pakkaa kobotti vuorokaudessa niin paljon paketteja, että operaattorin työajasta menee koko seitsemän tuntia jatkopaketoimiseen.

5.4 Kannattavuus

Ilman kobotin integrointia pakkaussoluun vie pakkaustyön suorittaminen operaattorilta 1 minuutin ja 10 sekuntia. Tämä vastaa 51 paketin tuntitehokkuutta. Kahden operaattorin pakkaussolu pystyi teoriassa paketoimaan 714 pakettia päivässä. Kannettavuuden laskemiseen käytetään 251:tä työpäivää vuodessa ja operaattoreilta 7 tuntia tehokasta työaikaa. Kobotin hankintakustannukseksi oletetaan 50 000 € ja vapautunut työntekijä säästää noin 50 000 € vuodessa.

Jos operaattori käy vain jatkopakkaamassa kobotin pakkaamat laatikot, nostaa se pakkaussolun tehokkuutta 27 %. Ihmistyötunteina tämä vastaa 4 tunnin ja 10 minuutin säästöä jokaista 1000 pakettia kohden. Jos operaattori jatkopaketoi ylinmääräisen ajan, nousee pakkaussolun tehokkuus 19 %. Tällöin jokaista 1000 pakettia kohden säästyy tunti ja 28 minuuttia.

Jos kobotti paketoi 7 tuntia päivässä, saavuttaisi pakkaussolu 381 paketin päivätehokkuuden. Tällöin kobotti maksaa itsensä takaisin vuodessa, ja pakkaussolun päivätehokkuus putoaa 67 % alkutilanteeseen verrattuna.

Jos kobotti paketoi 24 tuntia päivässä, saavuttaisi pakkaussolu 432 paketin päivätehokkuuden. Tällöin kobotti maksaa itsensä takaisin vuodessa, ja pakkaussolun päivätehokkuus putoaa 40 % alkutilanteeseen verrattuna.

Jos kobotti paketoi 7 tuntia päivässä ja sen nopeus optimoidaan kaksi kertaa nopeammaksi, saavuttaisi pakkaussolu 405 paketin päivätehokkuuden. Tällöin Kobotti maksaisi itsensä takaisin vuodessa, ja pakkaussolun päivätehokkuus putoaisi noin 44 % verrattuna alkutilanteeseen.

Jos kobotti paketoi 24 tuntia päivässä ja sen nopeus optimoidaan kolmanneksen nopeammaksi, saavuttaisi pakkaussolu 458 paketin päivätehokkuuden. Tällöin Kobotti maksaisi itsensä takaisin vuodessa, ja pakkaussolun päivätehokkuus putoaisi noin 36 % verrattuna alkutilanteeseen.

Jos kobotti paketoi 24 tuntia päivässä ja sen nopeus optimoidaan kaksi kertaa nopeammaksi, saavuttaisi pakkaussolu 691 paketin päivätehokkuuden. Tämä vaatisi kahden operaattorin käyttöä jatkopaketoimiseen, jotka paketoisivat yhteensä kymmenen ja puoli tuntia päivässä. Jos operaattorit pystyisivät tekemään tuottavaa työtä ylimääräisen kolme ja puoli tuntia, maksaisi kobotti itsensä takaisin kahdessa vuodessa, ja 36 työpäivässä. Pakkaussolun päivätehokkuus putoaisi vain noin 3 % verrattuna alkutilanteeseen.

6 Yhteenveto

Kobotin soveltuvuus pakkaustyöhön osoittautui mahdolliseksi. Kuitenkin pakkaustyön vaiheista vain ensimmäiset olivat kobotille mahdollisia. Viimeisien pakkausvaiheiden monimutkaisuus tarkoitti väistämättä näiden asettamista operaattorin suoritettaviksi. Pakkausvaiheet onnistuttiin kuitenkin jakamaan niin, että operaattorin ei tarvitse avata digiboksilaatikkoa, joka selkeyttää kobotin ja operaattorin työkuvia. Vaikka kobotille onnistuttiin ohjelmoimaan paketteja paketoiva demo, ei käytetty YuMi-kobotti välttämättä ole tarpeeksi järeä toistuvassa käytössä.

Insinöörityössä tutustuttiin kobotteihin ja opittiin operoimaan YuMi-kobottia. YuMi-kobotti tulikin erittäin tutuksi ja siitä luotiin ohjeistus asiakasyrityksen ja koulun käyttöön. Luotu ohjeistus (liite 2) antaa maallikolle yleiskuvan YuMi-kobotista. Ohjeistuksessa ei perehdytä syvälliseen ohjelmointiin, mutta se antaa lukijalle vankan pohjan kobotin ohjelmista ja operoinnista. Ohjeistuksen avulla lukija kykenee luomaan ensimmäiset ohjelmansa ja oppii käyttämään RobotWare-käyttöliittymää.

Vaikka pakkaussolu ehdotuksessa kobotille esitetty neljä pakkausvaihetta on lähes puolet, kuvaa se silti lyhyttä aikaa operaattorin suorittaessa nämä. Kobotti myös suorittaisi nämä huomattavasti hitaammin kuin ihminen, joka heikentää kobotin päivätehokkuutta. Vaikka ihmisten päivätehokkuuden arviointi on haastavaa, on silti selvää, että yhden operaattorin korvaaminen kobotilla on mahdollista vain, jos kobotti toimii ympäri vuorokauden. Tämä vaatisi myös huomattavan välivaraston, joka olisi oma investointinsa. Jos matalampi päivätehokkuus kuitenkin riittää, olisi kobotin integrointi jo nyt erittäin kannattavaa. Insinöörityön johtopäätöksenä voidaan todeta, että kobotin integroiminen pakkaussoluun olisi kannattavaa vain, jos operaattorin työkuormaa saataisiin kevennettyä enemmän kuin tämän selvitystyön perusteella voidaan olettaa mahdolliseksi. Voidaan myös kuitenkin todeta, että kobotit soveltuvat pakkaustyöhön logistiikassa ja niille soveltuvien töiden selvittäminen voi olla erittäin kannattavaa, kun sopiva työ löytyy.

Lähteet

1 Tietoa meistä. Verkkoaineisto. HUB logistics. <https://hub.fi/hub-logistics-yritys/hub-logistics-tietoa-meista/>. Luettu 21.3.2023.

2 Varastointipalvelut yrityksille. Verkkoaineisto. HUB logistics. <https://hub.fi/hub-logistics-palvelut/varastointipalvelut/>. Luettu 21.3.2023.

3 Toimitusketjukonsultointi. Verkkoaineisto. HUB logistics. <https://hub.fi/hub-logistics-palvelut/toimitusketjukonsultointi-hub-logistics-avataan-logistiikkasi-solmut/>. Luettu 21.3.2023.

4 Töihin HUBille, töissä HUBilla. Verkkoaineisto. HUB logistics. <https://hub.fi/tyo-hubilla/>. Luettu 21.3.2023.

5 Vastuullisuustyömme. Verkkoaineisto. HUB logistics. <https://hub.fi/hub-logistics-yritys/hub-logistics-vastuullisuus/>. Luettu 21.3.2023.

6 Sisälogistiikkapalvelut. Verkkoaineisto. HUB logistics. <https://hub.fi/hub-logistics-palvelut/sisalogistiikka-hub-logistics/>. Luettu 21.3.2023.

7 Logistiikan, varastoinnin ja sisälogistiikan lisäarvopalvelut. Verkkoaineisto. HUB logistics. <https://hub.fi/hub-logistics-palvelut/logistiikan-lisaarvopalvelut/>. Luettu 21.3.2023.

8 How Universal Robots Sold the First Cobot. Verkkoaineisto. Universal Robots. <https://www.universal-robots.com/about-universal-robots/news-centre/the-history-behind-collaborative-robots-cobots/>. Luettu 29.3.2023.

9 Murtonen, Jukka. 2019. Yhteistyökäyttörobottien käyttö teollisuudessa. Kandidaatintyö. Tampereen yliopisto, Tekniikan ja luonnontieteiden tiedekunta. <https://urn.fi/URN:NBN:fi:tuni-201912267122>. Luettu 29.3.2023.

10 Miriam Matúšová, Marcela Bučányová, Erika Hrušková. The future of industry with collaborative robots. Verkkoaineisto. Slovak University of Technology. <https://www.researchgate.net/publication/337710834_The_future_of_industry_with_collaborative_robots/fulltext/5de66371299bf10bc33bcf13/The-future-of-industry-with-collaborative-robots.pdf>. Luettu 29.3.2023.

11 Lehtonen, Sami 2018. Konenäkö ohjaa yhteistyörobotteja. Sickinsight sick-asiakaslehti 3/2018, s. 16–17. Luettu 29.3.2023.

12 Lehtonen, Sami 2018. Yhteistyörobotit Claus & Clara työskentelevät vuorotta. Sickinsight sick-asiakaslehti 3/2018, s. 18–20. Luettu 29.3.2023.

13 ABB: human-robot collaboration drives future development of robotics industry. 2019. Verkkoaineisto. ABB. <https://new.abb.com/news/detail/19640/abb-human-robot-collaboration-drives-future-development-of-robotics-industry-en/>. 19.4.2019. Luettu 25.4.2023.

14 Cobots: a Tool Changing the Chinese “3C Industry”. 2018. Verkkoaineisto. Universal Robots. <https://www.universal-robots.com/blog/cobots-a-tool-changing-the-chinese-3c-industry/>. 8.6.2018. Luettu 30.3.2023.

15 Product specification - IRB 14000. 2023. ABB. Verkkoaineisto. <https://search.abb.com/library/Download.aspx?DocumentID=3HAC052982-001&LanguageCode=en&DocumentPartId=&Action=Launch/>. Luettu 30.3.2023

16 Operating manual - IRB 14000. 2022. ABB. Verkkoaineisto. <https://search.abb.com/library/Download.aspx?DocumentID=3HAC052986-001&LanguageCode=en&DocumentPartId=&Action=Launch/>. Luettu 30.3.2023.

17 YuMi® – IRB 14000 Overview. 2018. ABB. Verkkoaineisto <http://search.abb.com/library/Download.aspx?DocumentID=9AKK106354A3256&LanguageCode=en&DocumentPartId=&Action=Launch/>. Luettu 30.3.2023.

18 Operating manual – IRC5 with FlexPendant. 2019. ABB. Verkkoaineisto. <https://us.v-cdn.net/5020483/uploads/editor/x7/0kqcav7a5yfx.pdf/>. Luettu 3.4.2023.

19 RobotStudio® – The world’s most used offline programming & simulation tool for robotics. 2022. ABB. Verkkoaineisto. <https://search.abb.com/library/Download.aspx?DocumentID=9AKK107045A3920&LanguageCode=en&DocumentPartId=&Action=Launch/>. Luettu 25.4.2023.

20 Overview - RobotStudio User Interface. 2021. ABB. Verkkoaineisto. <https://developercenter.robotstudio.com/api/robotstudio/articles/Concepts/Appearances/Appearances-Topic.html/>. Luettu 30.3.2023.

Demo-ohjelman koodi

Vasemman käden koodi alkaa sivulta 2 ja oikean käden alkaa sivulta 11.

```
MODULE MainModule
     PERS tasks tasklist{2}:=[["T_ROB_L"],["T_ROB_R"]];
     VAR syncident vaihe1;
     VAR syncident vaihe2;
     VAR syncident vaihe3;
     VAR syncident vaihe4;
     VAR syncident vaihe5;
     VAR syncident vaihe6;
     VAR syncident vaihe7;
     VAR syncident vaihe8;
     CONST robtarget a10:=[[343.16,232.98,343.38],[0.04744,-
0.206907,0.925981,-0.312248],[0,1,0,4],[-
168.145,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a20:=[[343.16,502.20,256.12],[0.0224374,0.206912,-
0.977755,0.0260607],[0,1,0,4],[-
168.142,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a30:=[[403.16,502.20,190.41],[0.0224314,0.206909,-
0.977756,0.0260584],[-1,1,0,4],[-
168.141,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a40:=[[401.22,502.20,165.70],[0.0224194,0.206909,-
0.977756,0.0260728],[-1,1,0,4],[-
168.139,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget a50:=[[399.96,154.44,163.62],[0.0270717,0.207072,-
0.97795,-0.00169378],[-1,1,-1,4],[-
168.157,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a60:=[[399.96,154.44,163.62],[0.0270758,0.207071,-
0.97795,-0.0016882],[-1,1,-1,4],[-
168.157,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a70:=[[417.00,268.09,162.85],[0.0306168,0.206988,-
0.977725,-0.0164893],[-1,1,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a80:=[[417.00,268.09,334.40],[0.0306126,0.206988,-
0.977726,-0.0164838],[0,1,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a90:=[[276.35,268.09,162.85],[0.0306138,0.206987,-
0.977726,-0.0164869],[-1,2,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a100:=[[214.30,-1.04,54.95],[0.0306224,0.206988,-
0.977725,-0.0164766],[-1,2,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a110:=[[282.23,-2.84,58.37],[0.246325,-
0.120584,0.96151,-0.0168011],[-1,2,-1,4],[-
168.634,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a120:=[[236.56,-1.04,162.85],[0.030624,0.206988,-
0.977725,-0.0164774],[-1,2,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a130:=[[284.77,-1.04,258.41],[0.0306225,0.206986,-
0.977726,-0.0164757],[-1,1,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a140:=[[537.67,-1.04,266.86],[0.0306302,0.206991,-
0.977724,-0.0164721],[-1,1,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a150:=[[566.43,34.87,114.87],[0.0306303,0.206997,-
0.977723,-0.0164658],[-1,1,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
     CONST robtarget a160:=[[493.85,34.87,114.87],[0.0306348,0.206999,-
0.977723,-0.0164668],[-1,1,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST robtarget a170:=[[588.25,34.87,114.87],[0.0306344,0.206999,-
0.977723,-0.016469],[-2,1,-1,4],[-
168.195,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
a180:=[[462.04,210.02,328.87],[0.0306308,0.206997,-0.977724,-
0.0164494],[-1,1,-1,4],[-168.194,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
a190:=[[520.63,258.30,276.12],[0.0306262,0.206998,-0.977724,-
0.0164488],[-1,1,-1,4],[-168.194,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget a200:=[[286.45,262.79,230.21],[0.0306174,0.20699,-
0.977726,-0.0164467],[0,1,-1,4],[-
168.192,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
a210:=[[417.04,154.44,163.62],[0.0270697,0.207071,-0.97795,-
0.00169841],[-1,1,-1,4],[-168.157,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
a220:=[[417.04,154.44,163.62],[0.0270691,0.207069,-0.97795,-
0.00169833],[-1,1,-1,4],[-168.157,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c10:=[[254.63,76.38,237.13],[0.283233,-
0.0278161,0.958408,0.0214498],[-1,2,-1,4],[-
175.545,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c20:=[[315.54,11.38,209.72],[0.251326,-
0.0397099,0.966888,0.0196199],[-1,2,-
1,4],[175.862,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c30:=[[286.56,11.65,194.82],[0.379815,-
0.0450565,0.923959,0.00303966],[-1,2,-1,4],[-
177.397,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c40:=[[271.56,16.26,181.06],[0.453614,-
0.0352566,0.890495,-0.00307815],[-1,2,-1,4],[-
173.423,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c50:=[[250.65,9.70,158.89],[0.549902,-
0.0312913,0.834389,0.0205581],[-1,3,-1,4],[-
165.754,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
c60:=[[246.26,10.66,130.97],[0.647576,0.00979268,0.760942,0.0389385],[
-1,3,-1,4],[-160.183,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c70:=[[253.77,62.54,272.42],[0.138007,-
0.00427287,0.990421,-0.00114519],[-1,2,-1,4],[-
167.67,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c80:=[[251.92,65.92,265.85],[0.010278,-
0.0198651,0.998711,0.0455693],[-1,2,-1,4],[-
167.472,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c90:=[[251.92,65.92,174.83],[0.0110455,-
0.036688,0.998234,0.0453925],[-1,2,-1,4],[-
167.472,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c100:=[[251.92,65.92,231.09],[0.0110479,-
0.0366867,0.998234,0.0453935],[-1,2,-1,4],[-
167.472,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c110:=[[262.37,0.34,151.97],[0.641348,-
0.0307237,0.765674,0.0383629],[-1,3,-1,4],[-
163.704,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c120:=[[325.64,21.99,247.00],[0.437531,-
0.0365263,0.898406,-0.00993721],[-1,2,-
1,4],[173.836,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget c130:=[[386.95,55.50,273.30],[0.22283,-
0.0106715,0.973788,-0.0443908],[-1,2,-
1,4],[163.753,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget e10:=[[440.25,18.28,291.49],[0.0199716,-
0.0273505,0.999342,-0.0130137],[-1,1,-1,4],[-
161.178,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST robtarget e20:=[[440.25,-13.72,269.89],[0.01997,-
0.0273641,0.999341,-0.0130235],[-1,1,-1,4],[-
161.177,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget e30:=[[330.67,-
25.74,284.33],[0.00158158,0.016616,-0.99984,-0.00644359],[-1,1,-
1,4],[-155.373,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget e40:=[[310.88,-
25.25,236.74],[0.0511917,0.00546405,-0.998134,-0.0328397],[-1,1,-
1,4],[-145.601,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget e50:=[[261.72,-14.07,185.34],[0.00718019,-
0.0274948,0.99948,0.015265],[-1,2,-1,4],[-
146.203,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget e60:=[[261.95,-13.58,163.57],[0.0104613,-
0.0274788,0.999331,0.0217656],[-1,2,-1,4],[-
146.191,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k10:=[[225.18,200.17,275.14],[0.0408707,-
0.0344352,0.998566,-0.00325729],[0,2,-
1,4],[163.379,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
k20:=[[233.19,361.32,27.89],[0.0061364,0.00151135,-0.999857,-
0.0157013],[-1,1,0,4],[170.591,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
k30:=[[233.19,361.32,5.42],[0.00613739,0.00151118,-0.999857,-
0.0156848],[-1,1,0,4],[170.591,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
k40:=[[355.05,120.75,263.49],[0.00818304,0.0248004,0.998569,0.0466603]
,[-1,1,-1,4],[159.733,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k50:=[[487.53,-
16.92,222.58],[0.00817848,0.0247986,0.998569,0.0466593],[-1,1,-
1,4],[159.733,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k60:=[[489.63,-
12.33,123.29],[0.00908394,0.00534164,0.998863,0.0464934],[-1,2,-
1,4],[159.733,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k70:=[[489.63,-
12.33,101.02],[0.00908344,0.0053392,0.998863,0.0464954],[-1,2,-
1,4],[159.733,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k80:=[[486.63,-
7.37,150.63],[0.0081811,0.0248003,0.998569,0.0466604],[-1,2,-
1,4],[159.732,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k90:=[[477.77,3.43,150.63],[0.0108627,-
0.719025,0.694877,-0.00565633],[-1,2,-
2,4],[159.732,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k100:=[[474.65,-7.33,94.63],[0.00975246,-
0.718807,0.69503,-0.0123972],[-1,2,-
2,4],[159.717,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k110:=[[492.21,-2.72,136.99],[0.0153742,-
0.70049,0.711991,0.0463405],[-1,2,-
2,4],[153.808,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget k120:=[[392.80,172.32,165.07],[0.0153694,-
0.700489,0.711992,0.046335],[-1,2,-
1,4],[153.808,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget g5:=[[265.31,-13.09,195.16],[0.0159962,0.0278342,-
0.999232,-0.0224822],[-1,2,-1,4],[-
144.429,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget g10:=[[279.74,-13.54,178.76],[0.0634363,0.021658,-
0.997097,-0.0361112],[-1,2,-1,4],[-
167.994,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget g20:=[[204.63,-
1.87,191.76],[0.282143,0.318325,0.904643,0.0261877],[-1,2,-1,4],[-
152.773,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST robtarget
g30:=[[168.94,10.95,166.56],[0.433888,0.426437,0.789524,0.0809003],[-
1,2,-1,4],[-139.952,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g40:=[[133.59,24.45,108.88],[0.544593,0.59069,0.547739,0.233423],[-
1,3,0,4],[-138.667,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g50:=[[174.57,7.15,123.39],[0.158547,0.659312,0.723456,0.129543],[-
1,2,0,4],[-145.724,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g60:=[[182.41,19.53,68.88],[0.185801,0.663259,0.706126,0.164171],[-
1,2,0,4],[-147.394,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g70:=[[219.13,21.84,55.95],[0.156961,0.669894,0.712318,0.138593],[-
1,2,0,4],[-159.371,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g80:=[[326.85,7.14,40.12],[0.177252,0.689571,0.672318,0.202637],[-
1,2,0,4],[-173.427,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g90:=[[352.29,4.16,33.27],[0.227637,0.671596,0.653309,0.265192],[-
1,2,0,4],[-174.848,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g100:=[[361.45,6.80,27.55],[0.220473,0.670841,0.647854,0.285745],[-
1,2,0,4],[-174.559,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g110:=[[210.05,22.35,72.83],[0.263809,0.633796,0.661309,0.302287],[-
1,2,0,4],[-146.463,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g120:=[[269.42,229.32,108.36],[0.263808,0.633798,0.661308,0.302287],[-
1,2,-1,4],[-146.463,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g130:=[[473.43,229.32,17.56],[0.293959,0.62041,0.646017,0.333678],[-
1,2,-1,4],[-146.463,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g140:=[[476.31,153.38,17.56],[0.293971,0.62041,0.646011,0.333679],[-
1,2,-1,4],[-146.465,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
g150:=[[303.92,236.77,48.18],[0.29399,0.620417,0.646,0.333672],[-1,2,-
1,4],[-146.465,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget j10:=[[195.07,4.81,111.16],[0.171716,-
0.0597783,0.983068,0.022749],[-1,2,-1,4],[-
141.036,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget j20:=[[236.02,-5.87,-
39.10],[0.600711,0.0111671,0.799056,0.0230461],[-1,3,-1,4],[-
156.274,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget j30:=[[262.57,4.15,-73.43],[0.590581,-
0.038068,0.806073,0.00340352],[-1,3,-2,4],[-
154.714,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget j40:=[[302.42,2.94,-64.45],[0.659979,-
0.0022257,0.7512,0.0110282],[-1,3,-2,4],[-
162.304,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget j50:=[[325.12,4.71,-
64.76],[0.804648,0.0491069,0.591661,0.00816799],[-1,3,-2,4],[-
163.784,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget j60:=[[360.14,-4.13,-20.73],[0.779058,-
0.0587826,0.622406,-0.0471603],[-1,3,-2,4],[-
163.616,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
j70:=[[361.87,25.28,21.14],[0.489813,0.481726,0.568149,0.453023],[-
1,3,-1,4],[-162.469,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST robtarget
j80:=[[498.08,25.28,153.83],[0.489818,0.481734,0.568135,0.453026],[-
1,2,0,4],[-162.47,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
j90:=[[358.65,25.28,153.83],[0.489817,0.481736,0.568134,0.453028],[-
1,3,0,4],[-162.47,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
j100:=[[273.76,246.70,51.62],[0.429306,0.756839,0.447199,0.207134],[-
1,3,-1,4],[-152.296,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
j110:=[[379.82,161.44,12.60],[0.405255,0.769994,0.453494,0.192927],[-
1,3,-1,4],[-152.297,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
j120:=[[368.33,180.72,12.60],[0.405255,0.769995,0.453493,0.192927],[-
1,3,-1,4],[-152.297,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m10:=[[505.87,259.51,97.76],[0.143571,-
0.542121,0.710077,-0.425774],[-
1,1,0,4],[153.795,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m20:=[[505.87,205.86,97.76],[0.143568,-
0.542118,0.710078,-0.425777],[-1,1,-
1,4],[153.795,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m30:=[[505.87,280.90,118.29],[0.143572,-
0.542121,0.710075,-0.425777],[-
1,1,0,4],[153.795,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m40:=[[505.87,234.38,284.51],[0.143574,-
0.542107,0.710078,-0.425788],[-1,1,-
1,4],[153.796,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m50:=[[560.09,16.97,321.03],[0.00334848,0.709854,-
0.70423,-0.0125424],[-1,1,-
2,4],[149.445,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m60:=[[608.24,12.68,196.15],[0.148727,0.722226,-
0.655315,-0.163804],[-1,1,-
1,4],[153.881,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m70:=[[558.02,17.59,144.44],[0.115019,0.719351,-
0.676545,-0.107666],[-1,1,-
1,4],[170.288,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m80:=[[518.93,17.59,144.44],[0.115027,0.71935,-
0.676544,-0.107664],[-1,1,-
2,4],[170.288,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m90:=[[492.58,17.62,104.21],[0.020493,0.728191,-
0.68476,-0.020559],[-1,1,-
2,4],[170.289,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m100:=[[530.75,23.27,167.04],[0.0738356,0.723639,-
0.68275,-0.0688927],[-1,1,-
2,4],[173.148,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m110:=[[501.76,313.27,200.36],[0.245319,0.83029,-
0.48918,0.105544],[-1,1,-
1,4],[175.588,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
m120:=[[476.30,249.27,23.53],[0.400851,0.872428,0.0539706,0.274363],[-
1,0,-1,4],[160.993,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
m130:=[[469.44,223.05,9.45],[0.400842,0.872428,0.0539625,0.274378],[-
1,0,-1,4],[160.993,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m140:=[[407.78,216.52,-
6.81],[0.397631,0.877649,0.158836,0.215391],[-1,0,-
1,4],[165.965,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m150:=[[407.78,216.52,-
6.81],[0.397632,0.877649,0.158836,0.21539],[-1,0,-
1,4],[165.965,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST robtarget
m160:=[[338.78,279.83,13.16],[0.442153,0.864795,0.146432,0.187586],[-
1,0,-1,4],[167.865,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
m170:=[[309.32,197.86,224.45],[0.0998969,0.665837,0.730537,0.114011],[
-1,2,0,4],[164.628,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
m180:=[[411.00,6.69,122.98],[0.0296767,0.665242,0.744835,0.042354],[-
1,2,0,4],[159.848,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
m190:=[[352.75,0.97,34.54],[0.187859,0.661305,0.68983,0.226978],[-
1,2,0,4],[178.666,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m200:=[[300.97,4.03,-
53.68],[0.446935,0.546428,0.537349,0.461434],[-1,3,-1,4],[-
172.566,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m210:=[[319.53,13.48,-
63.27],[0.4905,0.531322,0.51157,0.464115],[-1,3,-1,4],[-
171.617,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m220:=[[322.57,13.55,-
81.15],[0.485273,0.503842,0.536684,0.47183],[-1,3,-1,4],[-
172.47,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
m230:=[[289.38,76.79,35.11],[0.460749,0.648151,0.605459,0.0321055],[-
1,3,-1,4],[-171.172,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget m240:=[[242.46,341.47,83.66],[0.241476,-
0.058629,0.861368,-0.443053],[-1,2,-
1,5],[146.729,9E+09,9E+09,9E+09,9E+09,9E+09]];
    PROC main()
        WaitSyncTask vaihe1, tasklist;
        laatikontuominen;
        WaitSyncTask vaihe2, tasklist;
        läpänavaaminen;
        WaitSyncTask vaihe3, tasklist;
        kallistaminen;
        WaitSyncTask vaihe4, tasklist;
        asettelu;
        WaitSyncTask vaihe5, tasklist;
        kannenavaaminen;
        WaitSyncTask vaihe6, tasklist;
        kaukosäädin;
        WaitSyncTask vaihe7, tasklist;
        kannensulkeminen;
        WaitSyncTask vaihe8, tasklist;
        laatikonvieminen;
    ENDPROC
    PROC laatikontuominen()
        g_GripIn;
        MoveJ a10, v1000, z50, tool0;
        MoveJ a20, v1000, z50, tool0;
        MoveJ a30, v1000, z50, tool0;
        MoveJ a40, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ a50, v1000, z50, tool0;
        MoveJ a60, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ a70, v1000, z50, tool0;
        MoveJ a80, v1000, z50, tool0;
        MoveJ a90, v1000, z50, tool0;
        MoveJ a100, v1000, z50, tool0;
        MoveJ a110, v1000, z50, tool0;
```

```
    WaitRob\InPos;
    MoveJ a120, v1000, z50, tool0;
    MoveJ a130, v1000, z50, tool0;
    MoveJ a140, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ a150, v1000, z50, tool0;
    MoveJ a160, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ a170, v1000, z50, tool0;
    MoveJ a180, v1000, z50, tool0;
    MoveJ a190, v1000, z50, tool0;
    MoveJ a200, v1000, z50, tool0;
    g_GripOut;
ENDPROC
PROC läpänavaaminen()
    WaitTime 15;
    MoveJ c10, v1000, z50, tool0;
    MoveJ c20, v1000, z50, tool0;
    MoveJ c30, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ c40, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ c50, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ c60, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ c110, v1000, z50, tool0;
    MoveJ c120, v1000, z50, tool0;
    MoveJ c130, v1000, z50, tool0;
ENDPROC
PROC kallistaminen()
    WaitTime 21;
    MoveJ e10, v1000, z50, tool0;
    MoveJ e20, v1000, z50, tool0;
    WaitRob\InPos;
    g_GripIn;
    WaitRob\InPos;
    MoveJ e30, v1000, z50, tool0;
    WaitTime 5;
    MoveJ e40, v1000, z50, tool0;
    MoveJ e50, v1000, z50, tool0;
    MoveJ e60, v1000, z50, tool0;
ENDPROC
PROC asettelu()
    WaitTime 5;
    MoveJ g10, v1000, z50, tool0;
    WaitRob\InPos;
    g_GripOut;
    WaitRob\InPos;
    MoveJ g20, v1000, z50, tool0;
    MoveJ g30, v1000, z50, tool0;
    MoveJ g40, v1000, z50, tool0;
    MoveJ g50, v1000, z50, tool0;
    MoveJ g60, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ g70, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ g80, v1000, z50, tool0;
    MoveJ g90, v1000, z50, tool0;
    MoveJ g100, v1000, z50, tool0;
```

```
    WaitRob\InPos;
    MoveJ g110, v1000, z50, tool0;
    MoveJ g120, v1000, z50, tool0;
    MoveJ g130, v1000, z50, tool0;
    MoveJ g140, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ g150, v1000, z50, tool0;
ENDPROC
PROC kannenavaaminen()
    WaitTime 10;
    g_GripIn;
    MoveJ j10, v1000, z50, tool0;
    MoveJ j20, v1000, z50, tool0;
    MoveJ j30, v1000, z50, tool0;
    MoveJ j40, v1000, z50, tool0;
    MoveJ j50, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ j60, v1000, z50, tool0;
    WaitRob\InPos;
    g_GripOut;
    WaitRob\InPos;
    MoveJ j70, v1000, z50, tool0;
    MoveJ j80, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ j90, v1000, z50, tool0;
    MoveJ j100, v1000, z50, tool0;
    MoveJ j110, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ j120, v1000, z50, tool0;
ENDPROC
PROC kaukosäädin()
    MoveJ k10, v1000, z50, tool0;
    MoveJ k20, v1000, z50, tool0;
    MoveJ k30, v1000, z50, tool0;
    WaitRob\InPos;
    g_GripIn;
    WaitRob\InPos;
    MoveJ k40, v1000, z50, tool0;
    MoveJ k50, v1000, z50, tool0;
    MoveJ k60, v1000, z50, tool0;
    MoveJ k70, v1000, z50, tool0;
    WaitRob\InPos;
    g_GripOut;
    MoveJ k80, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ k90, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ k100, v1000, z50, tool0;
    WaitRob\InPos;
    MoveJ k110, v1000, z50, tool0;
    MoveJ k120, v1000, z50, tool0;
    WaitRob\InPos;
ENDPROC
PROC kannensulkeminen()
    MoveJ m10, v1000, z50, tool0;
    MoveJ m20, v1000, z50, tool0;
    WaitTime 5;
    MoveJ m30, v1000, z50, tool0;
    MoveJ m40, v1000, z50, tool0;
    MoveJ m50, v1000, z50, tool0;
```

```
        MoveJ m60, v1000, z50, tool0;
        MoveJ m70, v1000, z50, tool0;
        MoveJ m80, v1000, z50, tool0;
        MoveJ m90, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ m100, v1000, z50, tool0;
        MoveJ m110, v1000, z50, tool0;
        MoveJ m120, v1000, z50, tool0;
        MoveJ m130, v1000, z50, tool0;
        MoveJ m140, v1000, z50, tool0;
        MoveJ m150, v1000, z50, tool0;
        WaitTime 5;
        MoveJ m160, v1000, z50, tool0;
        MoveJ m170, v1000, z50, tool0;
        MoveJ m180, v1000, z50, tool0;
        MoveJ m190, v1000, z50, tool0;
        MoveJ m200, v1000, z50, tool0;
        MoveJ m210, v1000, z50, tool0;
        MoveJ m220, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ m230, v1000, z50, tool0;
        MoveJ m240, v1000, z50, tool0;
    ENDPROC
    PROC laatikonvieminen()
        WaitRob\InPos;
    ENDPROC
ENDMODULE
```

```
MODULE MainModule
    VAR syncident vaihe1;
    VAR syncident vaihe2;
    VAR syncident vaihe3;
    VAR syncident vaihe4;
    VAR syncident vaihe5;
    VAR syncident vaihe6;
    VAR syncident vaihe7;
    VAR syncident vaihe8;
    PERS tasks tasklist{2}:=[["T_ROB_L"],["T_ROB_R"]];
    CONST robtarget b10:=[[264.65,-294.25,216.37],[0.406984,-
0.86143,0.241613,-
0.184189],[0,1,1,4],[170.767,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget b20:=[[294.14,-242.26,104.46],[0.406965,-
0.861454,0.241588,-
0.184154],[1,1,1,4],[170.764,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget b30:=[[311.52,-182.35,49.52],[0.406978,-
0.861456,0.241571,-
0.184138],[1,1,1,4],[170.762,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget d10:=[[311.53,-251.81,49.52],[0.40696,-
0.86146,0.241577,-
0.184151],[1,1,1,4],[170.761,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget d20:=[[311.54,-251.81,261.02],[0.407006,-
0.861461,0.241529,-
0.184105],[0,1,1,4],[170.759,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget d30:=[[553.27,-165.65,140.77],[0.38213,-0.687092,-
0.411438,-0.461086],[1,0,1,4],[-
172.352,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget d40:=[[545.33,-
64.81,104.60],[0.216489,0.177847,0.86306,0.420273],[1,0,-
1,4],[168.471,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget d50:=[[526.37,-2.64,41.96],[0.0930614,-
0.392181,0.890868,0.209494],[2,0,-
1,4],[144.428,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget d60:=[[509.12,-2.68,41.95],[0.0930001,-
0.392149,0.89089,0.209488],[2,0,0,4],[144.426,9E+09,9E+09,9E+09,9E+09,
9E+09]];
    CONST robtarget d70:=[[509.12,-2.68,41.95],[0.0929971,-
0.392147,0.890892,0.209485],[2,0,0,4],[144.426,9E+09,9E+09,9E+09,9E+09
,9E+09]];
    CONST robtarget f10:=[[548.56,-217.12,146.78],[0.00910453,-
0.730203,0.683057,-0.0124365],[1,3,-
1,4],[173.607,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget f20:=[[377.29,-247.51,146.79],[0.00911355,-
0.7302,0.683059,-
0.0124468],[1,3,0,4],[173.606,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget f30:=[[230.87,-247.51,146.79],[0.654961,-
0.754888,0.0100603,0.0326891],[1,0,1,4],[173.601,9E+09,9E+09,9E+09,9E+
09,9E+09]];
    CONST robtarget f40:=[[273.59,-91.81,-82.39],[0.60624,-
0.766351,0.169383,-
0.12841],[1,0,1,4],[173.37,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget f50:=[[331.16,-150.06,-82.39],[0.606233,-
0.766356,0.169384,-
0.128411],[1,0,1,4],[173.37,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget f60:=[[308.02,-196.17,-52.25],[0.606248,-
0.766349,0.169345,-
0.128432],[1,0,1,4],[173.37,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST robtarget f70:=[[251.93,-272.33,32.80],[0.606261,-
0.766346,0.169291,-
0.128457],[1,0,1,4],[173.37,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget i10:=[[244.74,-228.94,10.94],[0.557116,-
0.711672,0.304191,-
0.301019],[1,1,1,4],[173.338,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget i20:=[[465.66,-228.94,10.94],[0.55712,-
0.71167,0.304193,-
0.301014],[1,1,1,4],[173.338,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget i30:=[[419.59,-177.74,-2.05],[0.468744,-
0.772778,0.33768,-
0.262804],[1,1,1,4],[173.339,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget i40:=[[419.59,-198.73,-2.05],[0.468742,-
0.77278,0.337676,-
0.262807],[1,1,1,4],[173.339,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget i50:=[[419.61,-198.73,44.67],[0.468747,-
0.77278,0.337665,-
0.262812],[1,1,1,4],[173.339,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget i60:=[[419.61,-170.38,44.67],[0.468745,-
0.772782,0.337663,-
0.262812],[1,1,1,4],[173.338,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget i70:=[[420.22,-170.08,28.37],[0.479609,-
0.762749,0.337061,-
0.27309],[1,1,1,4],[173.268,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget l10:=[[392.55,-266.79,46.54],[0.499755,-
0.744375,0.332821,-
0.292199],[1,1,1,4],[173.215,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget h10:=[[276.52,-208.40,-25.95],[0.606247,-
0.766353,0.1693,-
0.128468],[1,0,1,4],[173.369,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget h20:=[[276.53,-237.02,40.41],[0.606252,-
0.76635,0.169295,-
0.128471],[1,0,1,4],[173.368,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget h30:=[[276.52,-204.25,40.41],[0.606247,-
0.766352,0.169301,-
0.128478],[1,0,1,4],[173.368,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget h40:=[[276.50,-204.23,1.88],[0.61198,-
0.761678,0.168647,-
0.129942],[1,0,1,4],[173.34,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget h50:=[[357.53,-253.58,29.69],[0.611971,-
0.761685,0.168639,-
0.129952],[1,0,1,4],[173.337,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget n10:=[[542.25,-237.02,81.00],[0.46101,-
0.830661,0.257745,-
0.17618],[1,0,2,4],[173.309,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget n20:=[[542.24,-185.23,81.00],[0.46101,-
0.830659,0.257756,-
0.176175],[1,0,2,4],[173.309,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget n30:=[[481.11,-247.75,81.01],[0.461014,-
0.830655,0.257729,-
0.176223],[1,0,1,4],[173.308,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget n40:=[[454.04,-186.20,27.67],[0.348579,-
0.911849,0.186441,-
0.11074],[1,0,1,4],[151.437,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget n50:=[[427.45,-185.65,27.67],[0.348568,-
0.911853,0.186438,-
0.110748],[1,1,1,4],[151.437,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget n60:=[[401.19,-183.29,27.67],[0.348572,-
0.911851,0.18644,-
0.110746],[1,1,1,4],[151.437,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST robtarget n70:=[[391.80,-183.29,27.67],[0.348567,-
0.911854,0.186436,-
0.110749],[1,1,1,4],[151.437,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget n80:=[[401.45,-288.97,23.03],[0.480118,-
0.861136,0.143351,-
0.0859195],[1,1,1,4],[127.557,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget o10:=[[302.98,-3.61,226.75],[0.352739,-0.897433,-
0.0521695,-
0.259746],[1,0,1,5],[176.681,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget o20:=[[486.22,176.27,110.41],[0.314927,-
0.675805,0.666011,0.0231936],[2,1,2,5],[164.09,9E+09,9E+09,9E+09,9E+09
,9E+09]];
    CONST robtarget o30:=[[508.20,-6.65,70.74],[0.215285,-
0.743883,0.632536,-
0.0137125],[2,1,2,5],[157.361,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget o40:=[[505.40,-228.90,81.13],[0.168924,-
0.799496,0.554854,-
0.156232],[1,1,1,5],[146.316,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget o50:=[[244.88,-190.10,219.90],[0.516448,-
0.798029,0.264946,-
0.161969],[1,1,1,4],[151.918,9E+09,9E+09,9E+09,9E+09,9E+09]];
    PROC main()
        WaitSyncTask vaihe1, tasklist;
        laatikontuominen;
        WaitSyncTask vaihe2, tasklist;
        läpänavaaminen;
        WaitSyncTask vaihe3, tasklist;
        kallistaminen;
        WaitSyncTask vaihe4, tasklist;
        asettelu;
        WaitSyncTask vaihe5, tasklist;
        kannenavaaminen;
        WaitSyncTask vaihe6, tasklist;
        kaukosäädin;
        WaitSyncTask vaihe7, tasklist;
        kannensulkeminen;
        WaitSyncTask vaihe8, tasklist;
        laatikonvieminen;
    ENDPROC
    PROC laatikontuominen()
        g_GripIn;
        MoveJ b10, v1000, z50, tool0;
        MoveJ b20, v1000, z50, tool0;
        MoveJ b30, v1000, z50, tool0;
    ENDPROC
    PROC läpänavaaminen()
        MoveJ d10, v1000, z50, tool0;
        MoveJ d20, v1000, z50, tool0;
        MoveJ d30, v1000, z50, tool0;
        MoveJ d40, v1000, z50, tool0;
        g_GripOut;
        MoveJ d50, v1000, z50, tool0;
        MoveJ d60, v1000, z50, tool0;
        MoveJ d70, v1000, z50, tool0;
    ENDPROC
    PROC kallistaminen()
        MoveJ f10, v1000, z50, tool0;
        g_GripIn;
        MoveJ f30, v1000, z50, tool0;
        MoveJ f40, v1000, z50, tool0;
```

```
        WaitTime 1;
        MoveJ f50, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ f60, v1000, z50, tool0;
        MoveJ f70, v1000, z50, tool0;
    ENDPROC
    PROC asettelu()
        WaitTime 2;
       MoveJ h10, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ h20, v1000, z50, tool0;
        MoveJ h30, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ h40, v1000, z50, tool0;
        WaitTime 10;
        MoveJ h50, v1000, z50, tool0;
        WaitTime 10;
    ENDPROC
    PROC kannenavaaminen()
        MoveJ i10, v1000, z50, tool0;
        MoveJ i20, v1000, z50, tool0;
        MoveJ i30, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ i40, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ i50, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ i60, v1000, z50, tool0;
        WaitRob\InPos;
        MoveJ i70, v1000, z50, tool0;
        WaitRob\InPos;
    ENDPROC
    PROC kaukosäädin()
        MoveJ l10, v1000, z50, tool0;
        WaitRob\InPos;
    ENDPROC
    PROC kannensulkeminen()
        MoveJ n10, v1000, z50, tool0;
        MoveJ n20, v1000, z50, tool0;
        WaitTime 5;
        MoveJ n30, v1000, z50, tool0;
        WaitTime 12;
        MoveJ n40, v1000, z50, tool0;
        MoveJ n50, v1000, z50, tool0;
        MoveJ n60, v1000, z50, tool0;
        MoveJ n70, v1000, z50, tool0;
        WaitTime 5;
        MoveJ n80, v1000, z50, tool0;
    ENDPROC
    PROC laatikonvieminen()
        MoveJ o10, v1000, z50, tool0;
        MoveJ o20, v1000, z50, tool0;
        MoveJ o30, v1000, z50, tool0;
        MoveJ o40, v1000, z50, tool0;
        MoveJ o50, v1000, z50, tool0;
    ENDPROC
ENDMODULE
```

ABB YuMi IRB 14000 -pikakäyttöopas

ABB YUMI IRB 14000 PIKAKÄYTTÖOPAS

Metropolia Ammattikorkeakoulu

Sisällys

1 Yleiskuva 4
1.1 Pendantin pikanäppäimet 4
1.2 Käyttöliittymän pikavalikot 5
1.3 Käyttöliittymän ohjelmat 6
1.4 Akselit 7
1.5 Koordinaatistot 8
1.6 Liikkumistavat 9
1.7 Askeltilat 10
1.8 Ohjelma rakenne 11
1.9 Molempien käsien käyttäminen 12
1.10 Liikkeen törmäysvalvonta 14
1.11 Ohjelmoitavat pikanäppäimet 15
2 Ohjelmat 16
2.1 SmartGripper 16
2.2 Integrated Vision 17
2.3 HotEdit 18
2.4 Inputs and Outputs 19
2.5 Jogging 20
2.6 Program Editor 21
2.7 Program Data 28
2.8 Production Window 29
2.9 Backup and Restore 30
2.10 Calibration 31
2.11 Control Panel 33
2.12 FlexPendant Explorer 34
2.13 Lock Screen 35
2.14 System Info 36
2.15 Event Log 37
3 Nopeat visuaaliset esimerkit 38
3.1 Ohjelman luominen ja sen ajaminen 38
3.2 Ohjelman lataaminen/tallentaminen 40

Johdanto

Tämän oppaan tarkoitus on antaa alustavan laatuinen käsitys ABB YuMi 14000 IRB -kobotin ominaisuuksista ja valmius yksinkertaisten ohjelmien tekemiseen. Luethan yleiskuva osion saadaksesi parhaan käsityksen kobotista. Ohjelmat osio sisältää selityksen kustakin ohjelmasta ja tärkeimmistä toiminnoista. Nopeiden visuaalisten esimerkkien tarkoitus on antaa käyttäjälle käsitys vain FlexPendantin peruskäytöstä, ohjelmointi harjoittelussa on käännyttävä muiden oppaiden puoleen.

Jeremias Walkeajärvi 2023
Metropolia Ammattikorkeakoulu

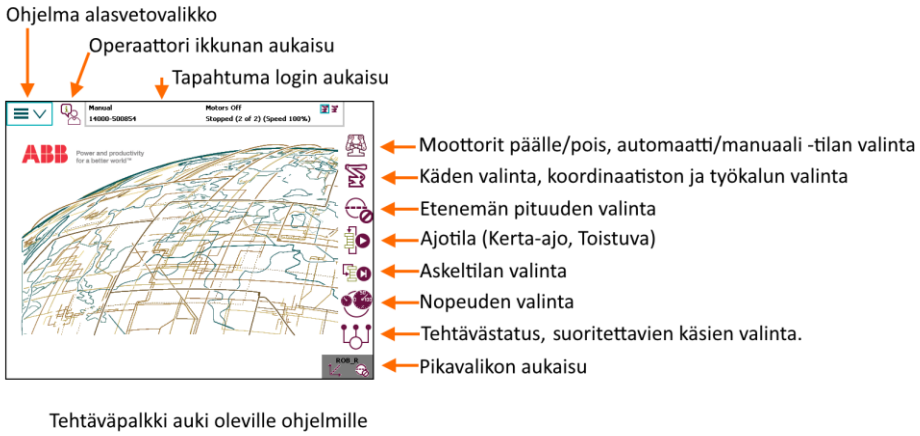
1 Yleiskuva

1.1 Pendantin pikanäppäimet

Kuva 1. Kuvassa FlexPendant, kuvaan merkattu nuolilla pikanäppäinten toiminnot.

Pendantin pikanäppäimet ovat hyödyllisiä ”jogging” -toimintoa käytettäessä, näppäimillä voi valita nopeasti akseliohjauksen tai koordinaatisto ohjauksen. Etenemän pituudella voidaan valita kuinka pitkän matkan kobotti liikkuu yhdellä joystick ohjauskomennolla. Alimpia harmaalla pohjalla olevia pikanäppäimiä käytetään ohjelmien ajamisessa. Ylimpien ohjelmoitavien pikanäppäiten käyttö käydään sivulla 15. Kuvassa näkyvä joystick-sauvaohjain liikkuu 8 suuntaan ja sitä voidaan pyörittää myötä- ja vastapäivään. oikealla näkyvä tummanharmaa painike on 3-asentoinen kuolleenmiehen kytkin, joka on tarkoitettu perinteisille roboteille. Pendantin yläpuolelta löytyy hätäseis painike, joka resetoidaan pyörittämällä sitä. Pendantin oikealla alaosassa olevan kumiläpän takaa löytyy usb portti, jolla pendant voidaan kiinnittää tietokoneeseen. Pendantin takapuolelta löytyvät pidike stylus-kynälle ja reset-painike.

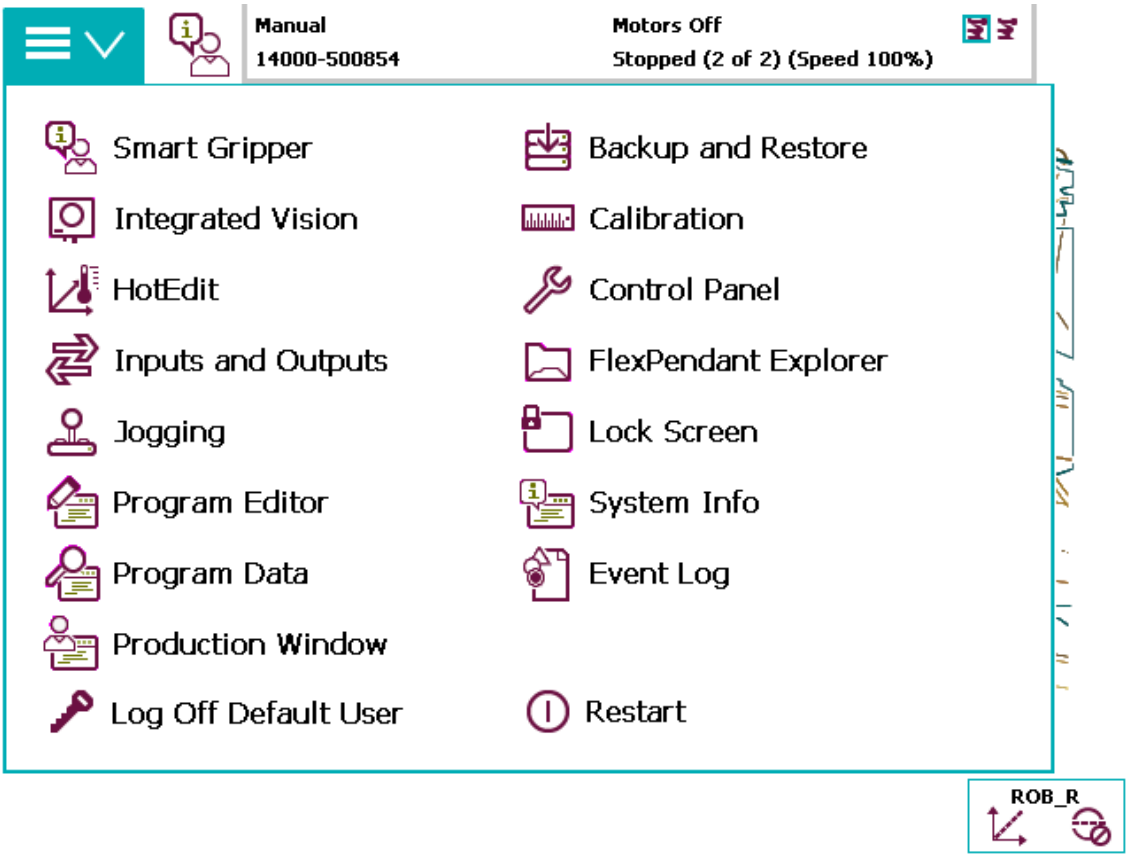
1.2 Käyttöliittymän pikavalikot



Kuva 2. Kuvassa RobotWare-käyttöliittymän pikavalikko, johon on merkattu nuolilla pikavalikosta löytyvät toiminnot sekä muuta käyttöliittymän rakennetta.

Pikavalikosta löytyvät kaikki olennaiset toiminnot kobotin hallintaan ja ajamiseen. Loput toiminnot löytyvät ohjelma alasvetovalikosta. Operaattori ikkunasta voi katsoa pyörivän ohjelman lähettämät viestit. Tapahtuma logi aukaistaan painamalla yläpalkkia. Siellä näkyvät jokainen poikkeava tapahtuma aikaleimoineen. Ylimpänä pikavalikosta löytyy operaattori paneeli, josta kytketään moottorit päälle ja valintaan manuaali- ja automaattitilan välillä. Operaattori tila on valittava manuaaliksi, kun kobottia halutaan ohjelmoida. Pikavalikosta löytyy pendantissakin oleva toiminto käden valitsemiselle, sieltä löytyy myös liikkumistavan ja koordinaatiston valinnat. Samasta alavalikosta voidaan määritellä kädelle myös työkalu ja työobjekti. Etenemän pituuden valinta viittaa kobotin suorittaman liikkeen pituuteen yhdellä ohjausnäpäytyksellä. Oletuksena tämä on pois päältä, jolloin kobotti liikkuu tasaisesti, niin pitkään kuin joystick on ohjattuna. Etenemä ajo on tarkoitettu työkalun tarkkaan sijoittamiseen. Ajotila voi olla kerran ohjelman suorittava tai ohjelmaa toistuvasti suorittava. Askeltilalla viitataan siihen, kuinka ohjelman askeleet suoritetaan. Askeltilat käydään läpi sivulla 10. Nopeuden valinta vaikuttaa vain manuaalitilaan, jos se määritellään manuaalitilassa. Automaattitilassa määriteltäessä se vaikuttaa myös manuaalitilaan. Tehtävä status valikosta voidaan valita suoritettavat kädet ja tarkistaa suorituksen tilanne.

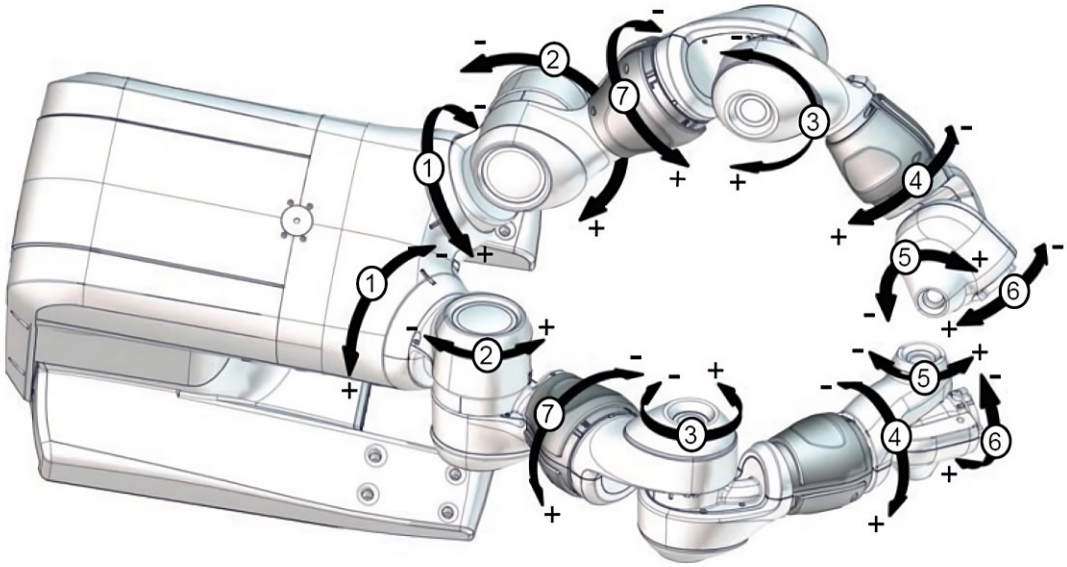
1.3 Käyttöliittymän ohjelmat



Kuva 3. Kuvassa RobotWare-käyttöliittymän ohjelma-alasvetovalikko.

Ohjelma-alasvetovalikosta voidaan aukaista ohjelmia ja käynnistää uudelleen FlexPendant. Log Off Default User painikkeesta voidaan kirjautua ulos käyttäjältä. Kobotin voi sammuttaa IRC5 pääyksikön virtakatkaisimesta. Kobotissa on kondensaattorit, joiden avulla päätietokone tekee hallitun sammuttamisen. Sammutettuasi ohjausyksikön ethän kuitenkaan kytke virtoja takaisin 15 sekunttiin, jotta pääyksikkö ehtii suorittamaan sammutamisen loppuun. Ohjelmien välillä voidaan siirtyä käyttämällä tehtäväpalkkia ja niitä voidaan sulkea oikeassa yläkulmassa olevalla x -painikkeella. Itse ohjelmien käyttämiseen perehdytään oppaan ohjelmat osiossa alkaen sivulta 16.

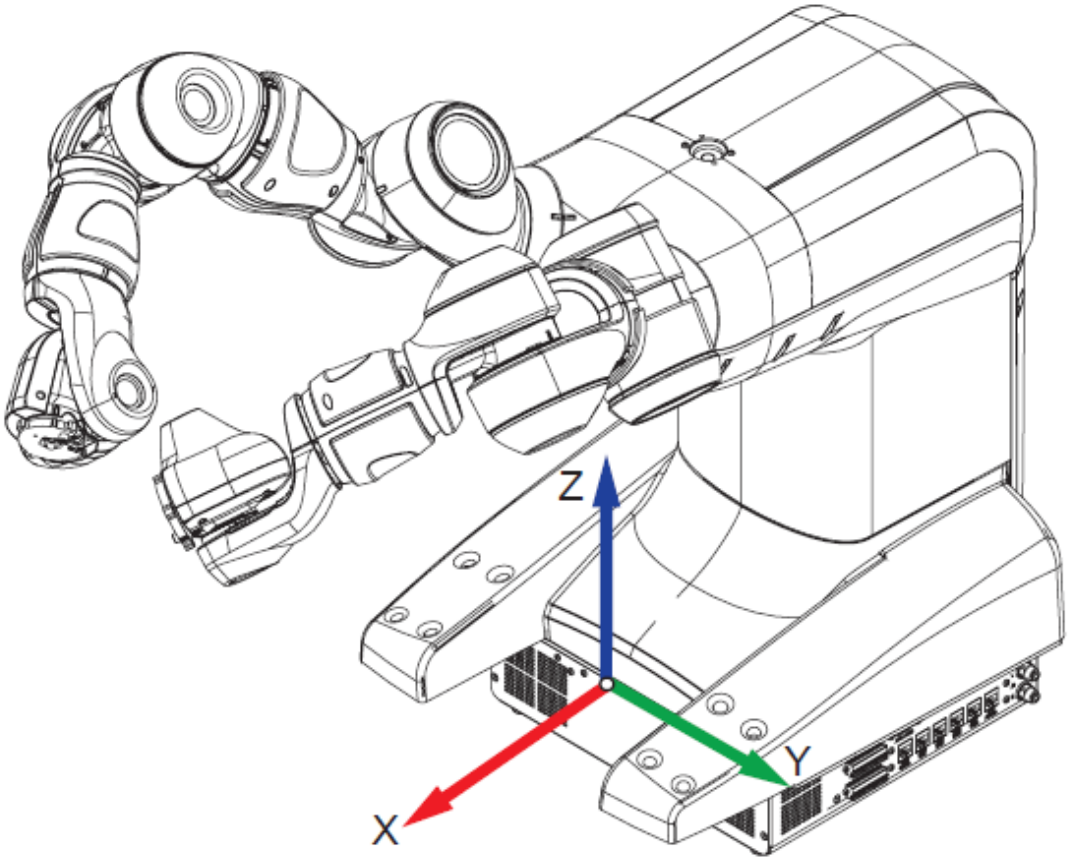
1.4 Akselit



Kuva 4. Kuvassa numeroitu kobotin käsien seitsemän liikkuvaa akselia.

Akselit 1 ja 2 ovat käsivarren juuren pyörimis ja taivutus akselit. Akselit 3 ja 7 ovat kyynärvarren vastaavat pyörimis ja taivutus akselit. Akselit 4 ja 5 ovat ranteen pyörimis ja taivutus akselit. Akseli 6 on työkalun, eli YuMi:n tapauksessa tarttujan pyörimisakseli. Huomaathan, että seitsemättä akselia ei ole nimetty juuresta laskien järjestyksessä, koska se puuttuu perinteisistä 6-akselisista koboteista.

1.5 Koordinaatistot



Kuva 5. Kuvassa Base-koordinaatiston origo suhteessa kobottiin sekä x, y ja z suunnat.

Koordinaatisto valitaan pikavalikossa olevasta käden valinta -alavalikosta. Koordinaatisto valittaan käsille erikseen, koordinaatistot ovat seuraavat:

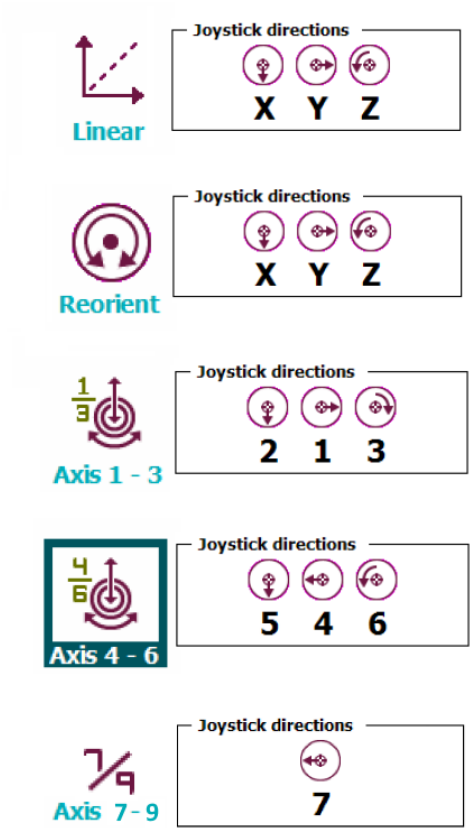
1. World, Koordinaatisto on koko robottisolulle ennalta määritelty koordinaatisto. Hyödyllinen jos samassa robottisolussa on useampi robotti.

2. Base, Oletus koordinaatisto, jonka origo on kobotin juuressa.

3. Tool, Koordinaatisto, joka johdetaan kobotin käyttämästä työkalusta.

4. Work, Koordinaatisto, joka johdetaan kobotin käsittelemästä työ objektista.

1.6 Liikkumistavat



Kuva 6. Kuvassa liikkumistavat ja niiden vallitessa olevat joystick-suunnat.

Liikkumistapa voidaan valita joko pendantin tai pikavalikon kautta. Liikkumistavat ovat seuraavat:

1. Linear, Työkalu liikkuu x, y ja z suunnissa, valitussa koordinaatistossa.

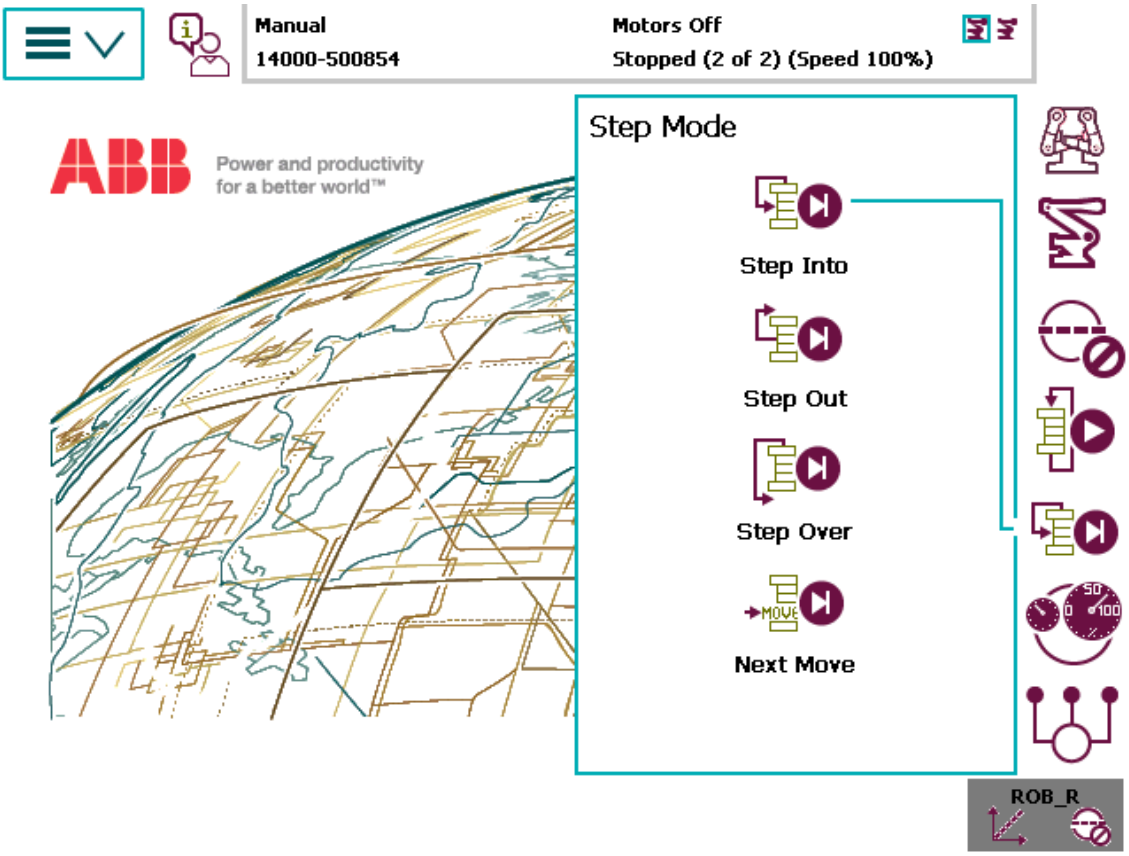
2. Reorient, Työkalun jälleen suuntaus. Työkalu pysyy paikallaan mutta sen suunta muuttuu.

3. Axis 13, Käsivarsien liikuttaminen.

4. Axis 46, Ranteen ja työkalun liikuttaminen.

5. Axis 79, Kyynärvarren pyörittäminen

1.7 Askeltilat



Kuva 7. Kuvassa pikavalikosta avattu askeltilat-valikko.

Askeltiloilla voidaan vaikuttaa kobotin toimintaan kutsuttaessa rutiineja ja käytettäessä askelpainikkeita pendantilla. Askeltilat ovat seuraavat:

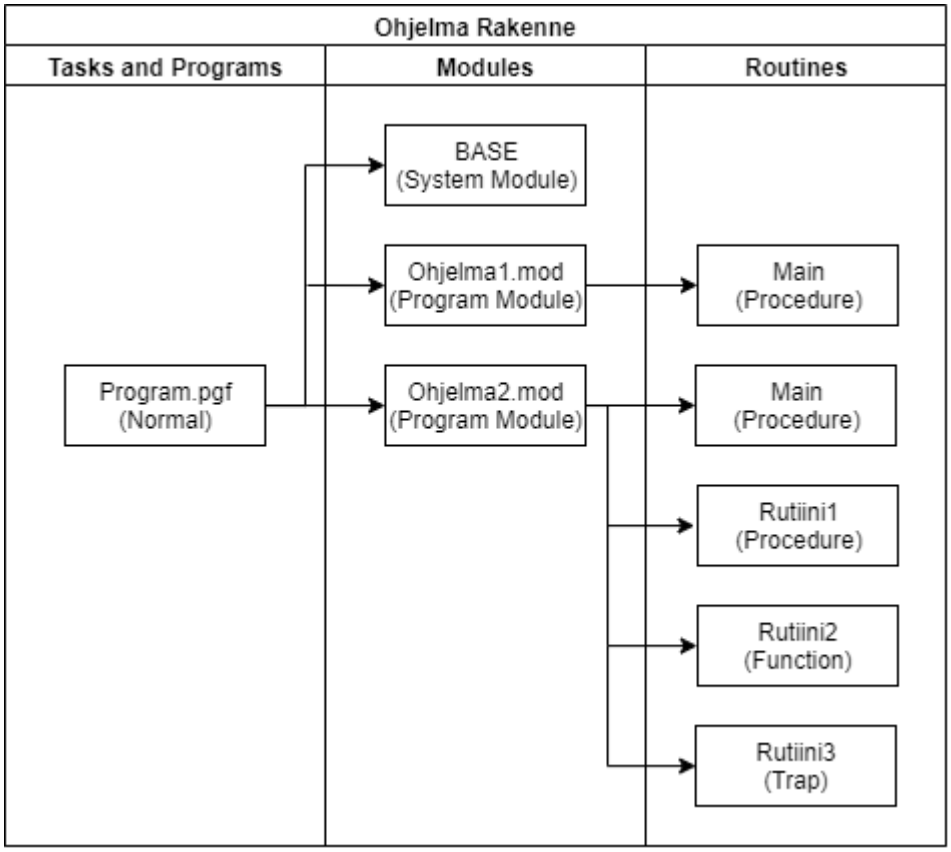
1. Step Into, Oletus askeltila, jolloin kutsuttuihin rutiineihin siirrytään heti niitä kutsuttaessa ja suoritetaan askel askeleelta.

2. Step Out, Uutta rutiinia kutsuttaessa suoritetaan kyseinen käsky loppuun ja siirrytään sitten uuteen rutiiniin. Ei voi käyttää Main rutiinissa.

3. Step Over, Kutsuttu rutiini suoritetaan yhtenä askeleen.

4. Next Move, Suorittaa askeleet seuraavaan move-käskyyn asti. Pysähtyy ennen ja jälkeen jokaista move-käskyä.

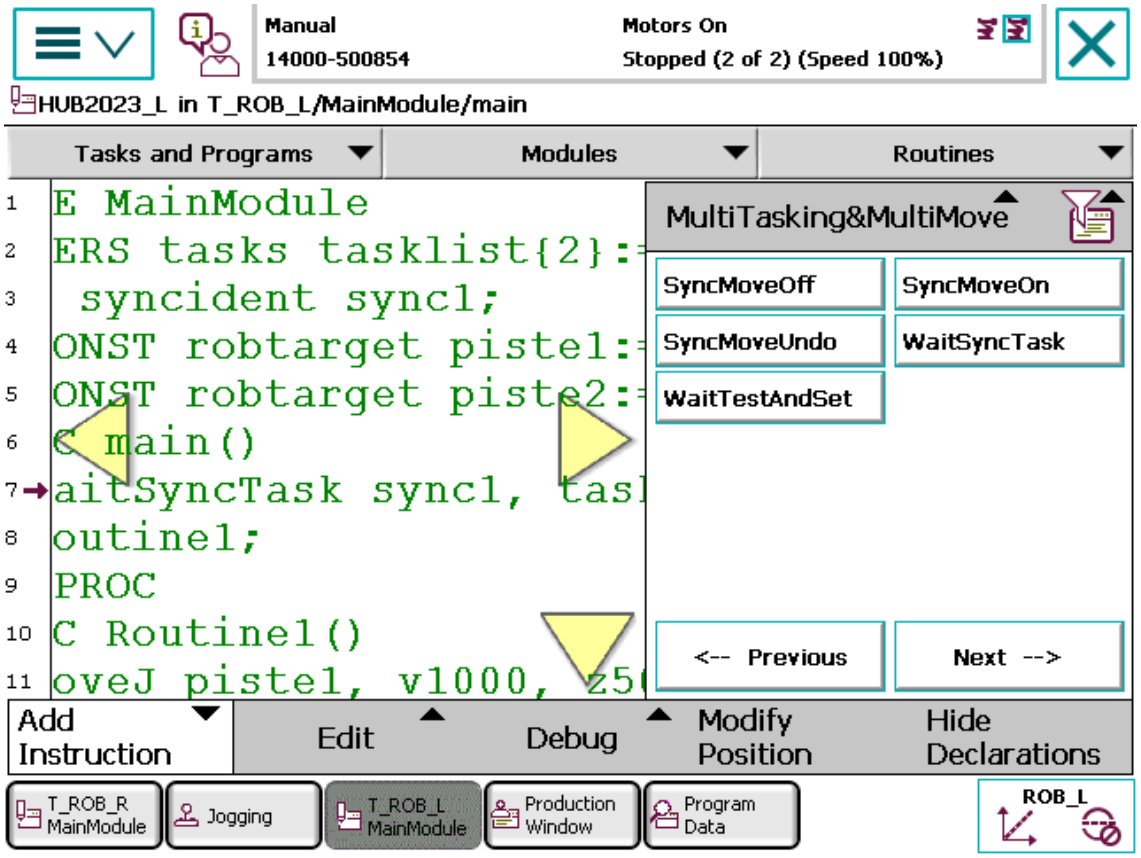
1.8 Ohjelma rakenne



Kuva 8. Kuvassa ohjelma rakenne sekä tiedostojen tyypit ja tallennusmuodot.

Ohjelmat ovat pgf-muotoisia ja niiden alle voidaan luoda ja ladata mod-muotoisia moduuleja. Moduulien alle luodaan itse rutiinit, jotka sisältävät ohjelmakoodin. Rutiinit voivat olla esimerkiksi procedure-tyyppisiä, jotka suorittavat sen sisältävän toimenpiteet. Function-rutiinit suorittavat toimenpiteet ja palauttavat lisäksi arvon. Trap-tyyppiset rutiinit ovat tarkoitettu ajettavaksi, jos normaali rutiini halutaan keskeyttää. Moduulin luotua sijaitsee main rutiini automaattisesti siinä, main rutiiniin voidaan ohjelmoida toimenpiteitä mutta sen päätarkoitus on kutsua muita rutiineja ja hallita kokonaiskuvaa.

1.9 Molempien käsien käyttäminen



Kuva 9. Kuvassa MultiMove- ja MultiTasking-komennot Program Editorissa.

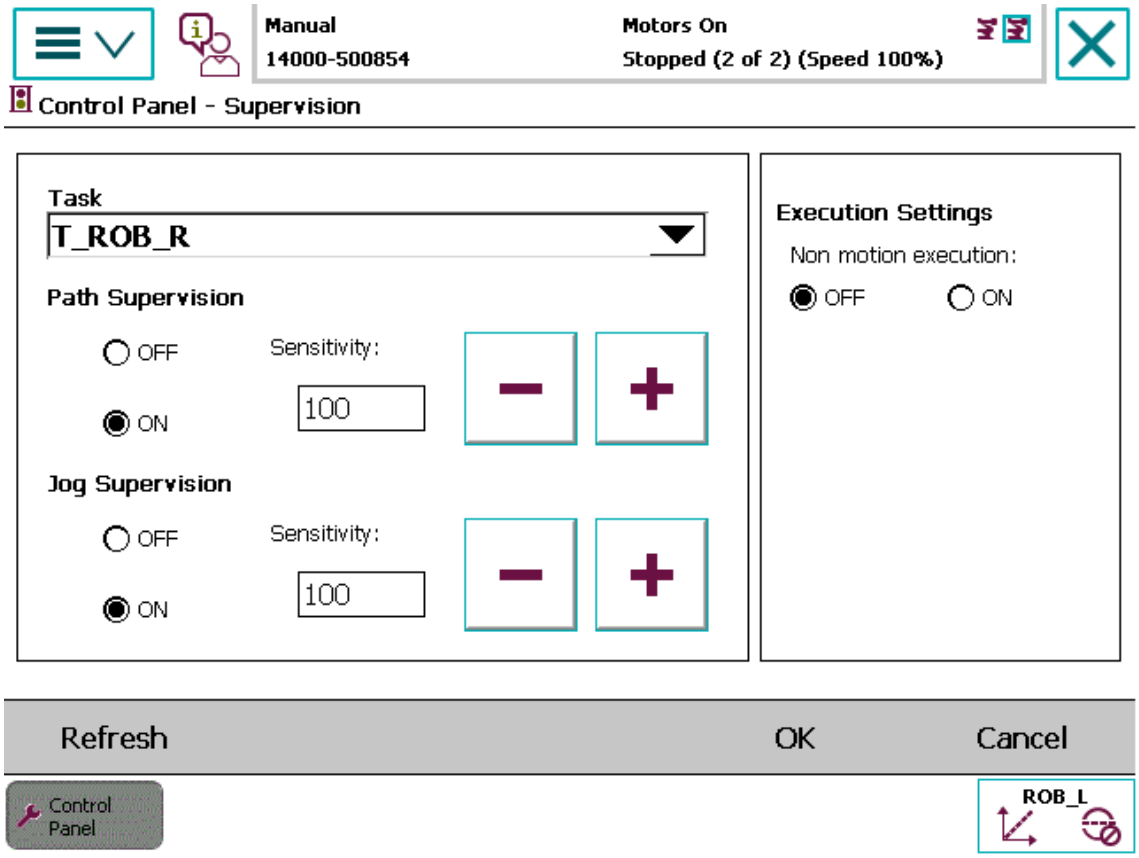
Multimove eli useamman robotin samanaikainen ohjaaminen samalla ohjaimella on kahdenkäden käyttöön vaadittava ominaisuus, sillä ohjain käsittelee kummankin käden erilliseksi robotiksi. MultiTasking on useamman tehtävän samanaikaista suorittamista samalla kädellä, kuitenkin niin että vain yksi tehtävä ohjaa liikkeitä kerrallaan. Käsiä voi käyttää yhdessä joko synkronoidusti tai samaan aikaan molempien käsien omia tehtäviä ajamalla. Kun halutaan käyttää käsiä synkronoidusti, on kaikki liikkeet tehtävä käyttäen samoja koordinaatteja. Näihin koordinaatteihin voi lisätä koordinaatisto poikkeamia sekä useita eri toimintoja, jolloin kädet eivät törmää. Synkronoitu ajo tapahtuu käyttämällä SyncMoveON komentoa ja valitsemalla parametriksi haluttu syncident tunnus ja tehtävälista, jossa on molemmat kädet. Kädet voidaan vapauttaa synkronoidusta ajosta komennolla SyncMoveOff. Ei synkronoitu ajo tapahtuu käyttämällä WaitSyncTask-

komentoa, johon valitaan myös syncident tunnus ja tehtävälista, jossa on molemmat kädet. Käsiä ei tarvitse vapauttaa, vaan kädet on synkronoitava niin usein, kun ne halutaan olevan samantahtiset. Molempien käsien ohjelmille pitää luoda samannimiset syncident tunnukset ja tehtävälistat.

```
MODULE MainModule
    PERS tasks tasklist{2}:=[["T_ROB_R"],["T_ROB_L"]];
     VAR syncident sync1;
    CONST robtarget piste1:=[[165.33,427.29,236.57],[0.0339899,-
0.605258,0.607021,-
0.513841],[0,0,0,4],[134.737,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget piste2:=[[273.02,349.42,244.61],[0.069013,-
0.605587,0.562783,-
0.55837],[0,0,0,4],[148.644,9E+09,9E+09,9E+09,9E+09,9E+09]];
     PROC main()
          WaitSyncTask sync1, tasklist;
          Routine1;
     ENDPROC
     PROC Routine1()
          MoveJ piste1, v1000, z50, tool0;
          MoveJ piste2, v1000, z50, tool0;
     ENDPROC
ENDMODULE
```

Esimerkki koodi ohjelmasta, joka on synkronoitu toisen käden kanssa, huomioi tehtävälistan esittely alussa.

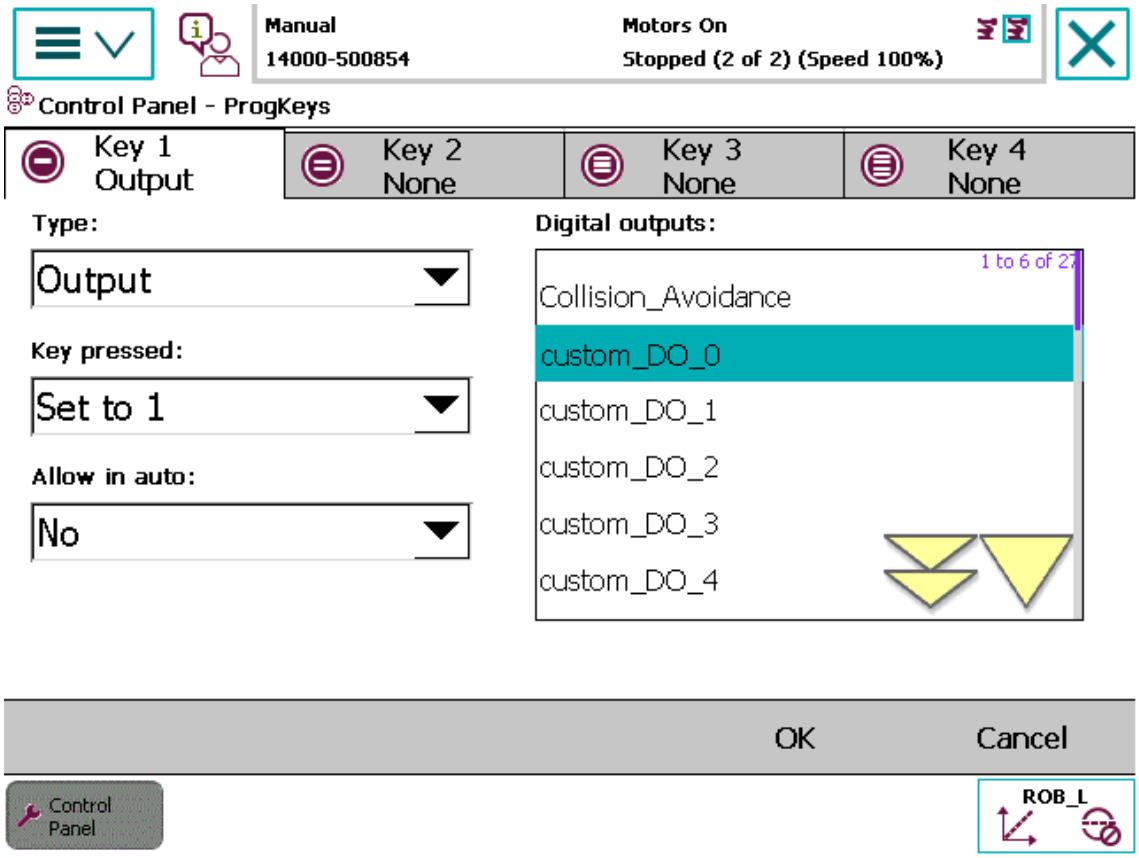
1.10 Liikkeen törmäysvalvonta



Kuva 10. Kuvassa hallintapaneelista löytyvä Supevision-ohjelma.

Herkkyyden säätämisellä viitataan siihen kuinka paljon voimaa kobotti saa käyttää ennen kuin se tulkitaan törmäykseksi. Herkkyyden maksimi voima on oletuksen 100, mutta sitä voidaan korottaa tai laskea. Voiman maksimi suuruus voidaan asettaa välille 5-300. Path supervision tarkoittaa kobotin itse tekemiä liikkeitä ja jog viittaa käyttäjän ajamiin liikkeisiin. Molemmat voidaan kytkeä halutessa pois päältä. Liikkeenvalvonta voidaan myös kytkeä ohjelmistokoodissa MotionSup \On ja MotionSup \Off -komennoilla, tämä toimii varmemmin. Oikealta löytyy kytkentä, ohjelmien ajamiseksi ilman liikkeitä. Tämä toiminto ei liity törmäysvalvontaan.

1.11 Ohjelmoitavat pikanäppäimet

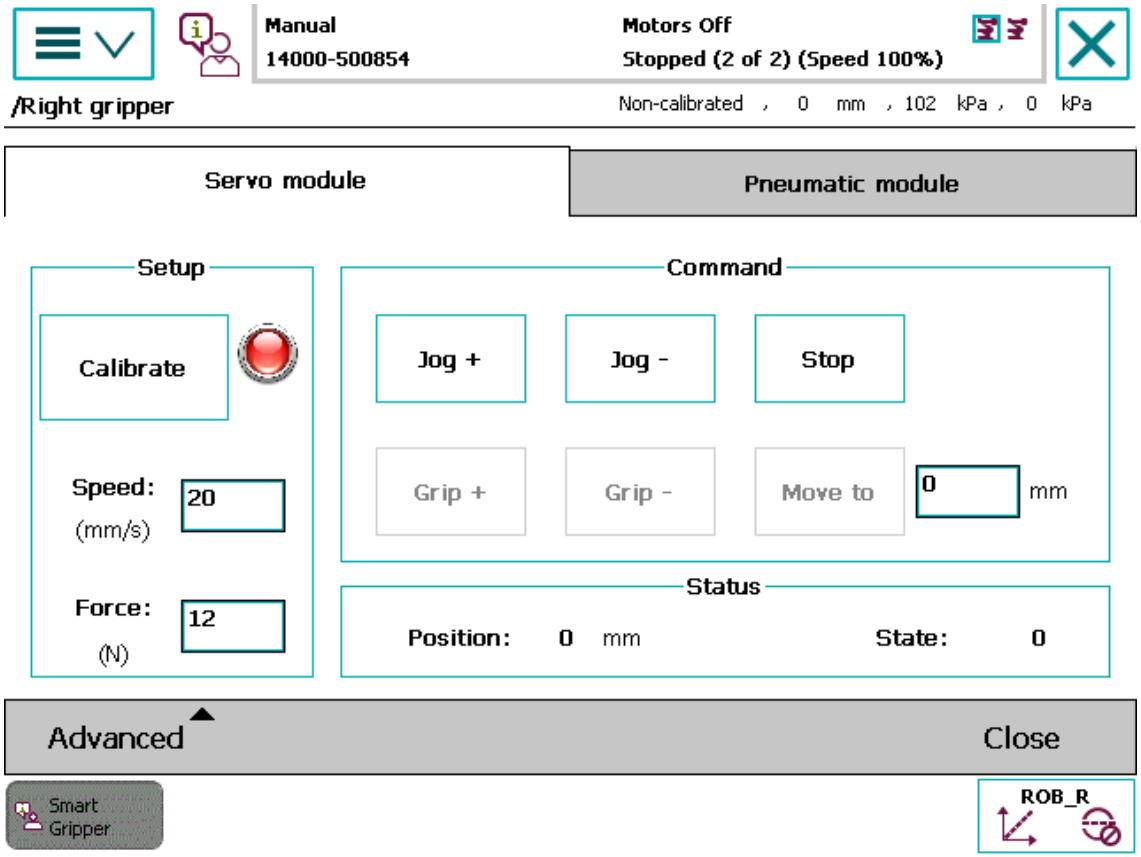


Kuva 11. Kuvassa hallintapaneelista löytyvä ProgKeys-ohjelma.

Pikanäppäinten määrittely tapahtuu control panel -ohjelmassa löytyvässä ProgKeys ohjelmassa. Näppäimelle määritellään yksikolmesta toiminnosta, jonka jälkeen toimintoa voidaan käyttää IO-portin tavoin ohjelmassa. Ensimmäinen toiminto on Input, jolloin määritellyn Digitaalisen input paikan arvoa voidaan hetkellisesti sykkiä päinvastaiseksi siitä, mitä se on sillä hetkellä. Seuraava toiminto on Output, jolloin määritellyn output paikan arvo voidaan muuttaa 5 eri tavalla. Vaihto päinvastaiseksi, asetus 1 arvoksi, asetus 0 arvoksi, asetus päinvastaiseksi arvoksi niin kauan kun painiketta painetaan ja määritellyn output paikan arvon sykkiminen kerran.

2 Ohjelmat

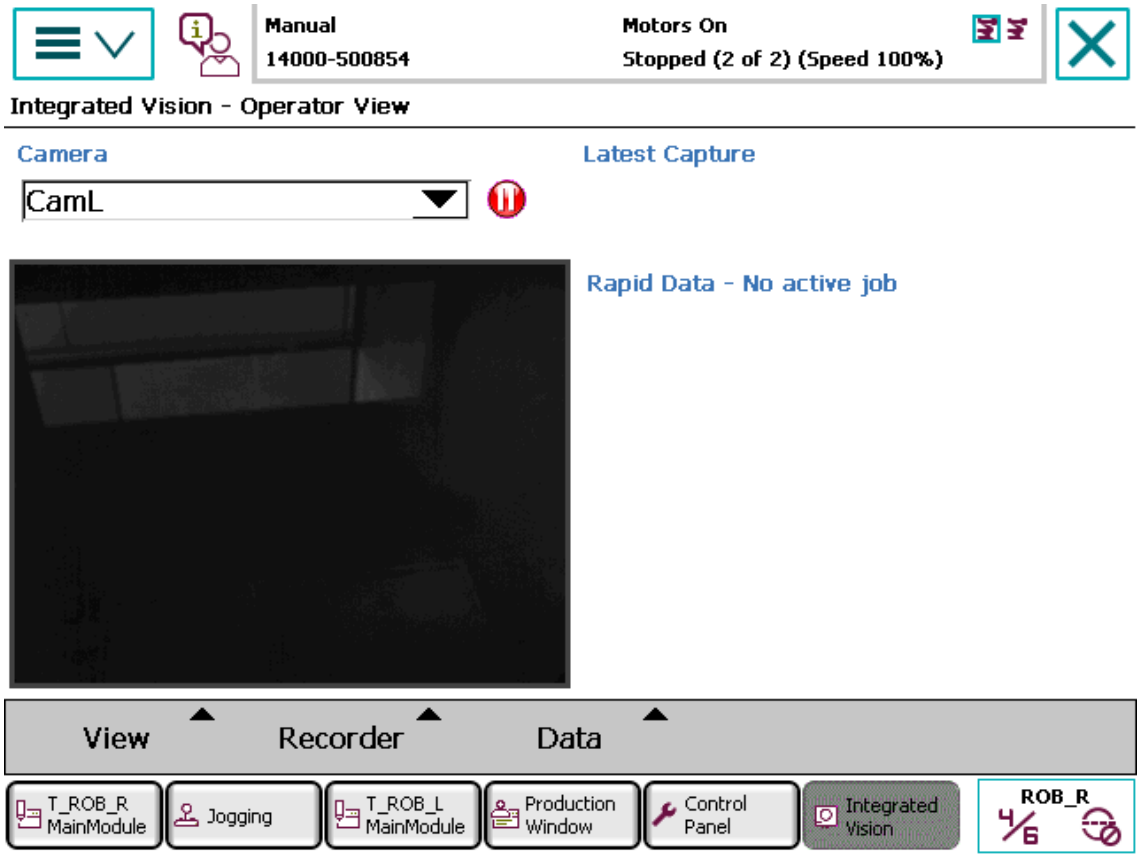
2.1 SmartGripper



Kuva 12. Kuvassa SmartGripper-ohjelman oikean tarttujan ikkuna.

SmartGripper on tarttujaa varten asennettu ohjelma, jolla onnistuu tarttujien tarkkailu ja kalibrointi. Tarttujan Kalibrointi tapahtuu ajamalla ”jog +” ja ”jog -” painikkeilla tarttuja ääriasentoihin ja painamalla tämän jälkeen calibrate-painiketta. Pneumatic module -välilehdeltä löytyvät painikkeet paineilma toiminnon testaamiselle.

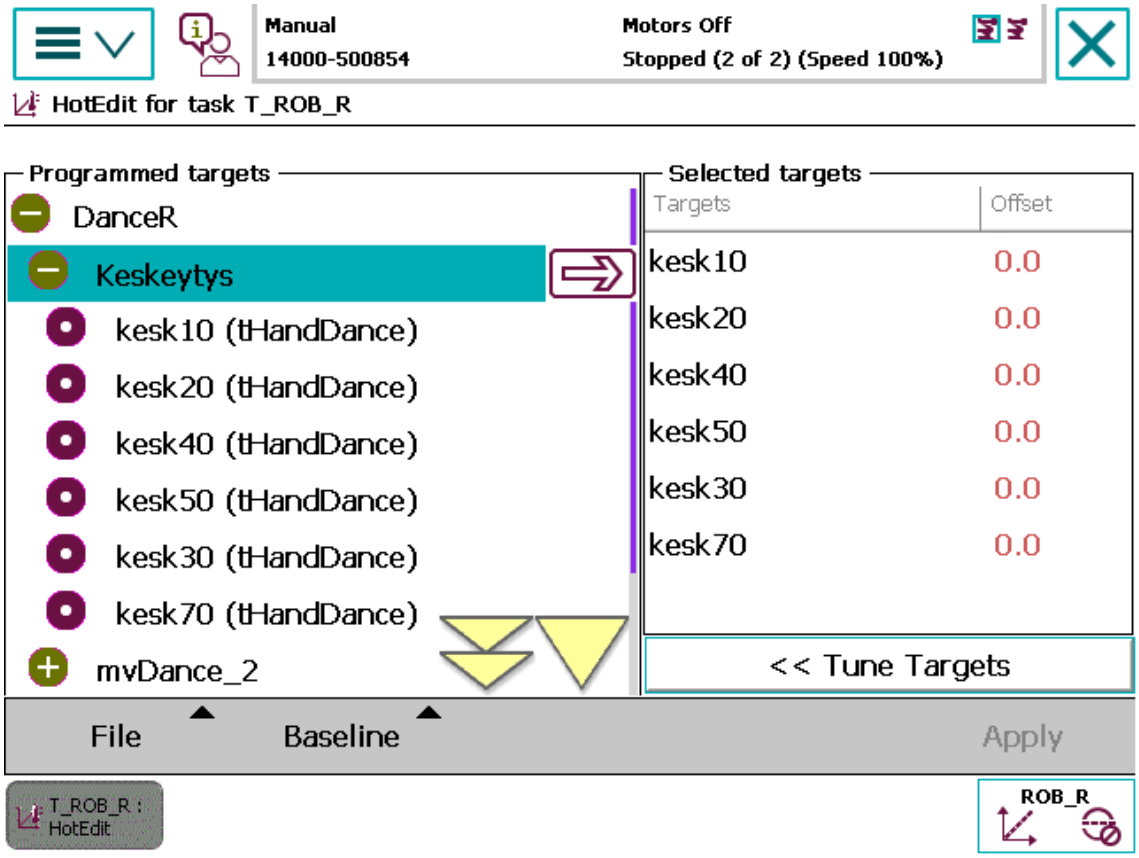
2.2 Integrated Vision



Kuva 13. Kuvassa Integrated Vision -ohjelma.

Integrated Vision -ohjelmalla voi tarkistaa kameroiden toiminta ja kuvan laatu. Itse konenäön käyttöön ei perehdytä tässä oppaassa.

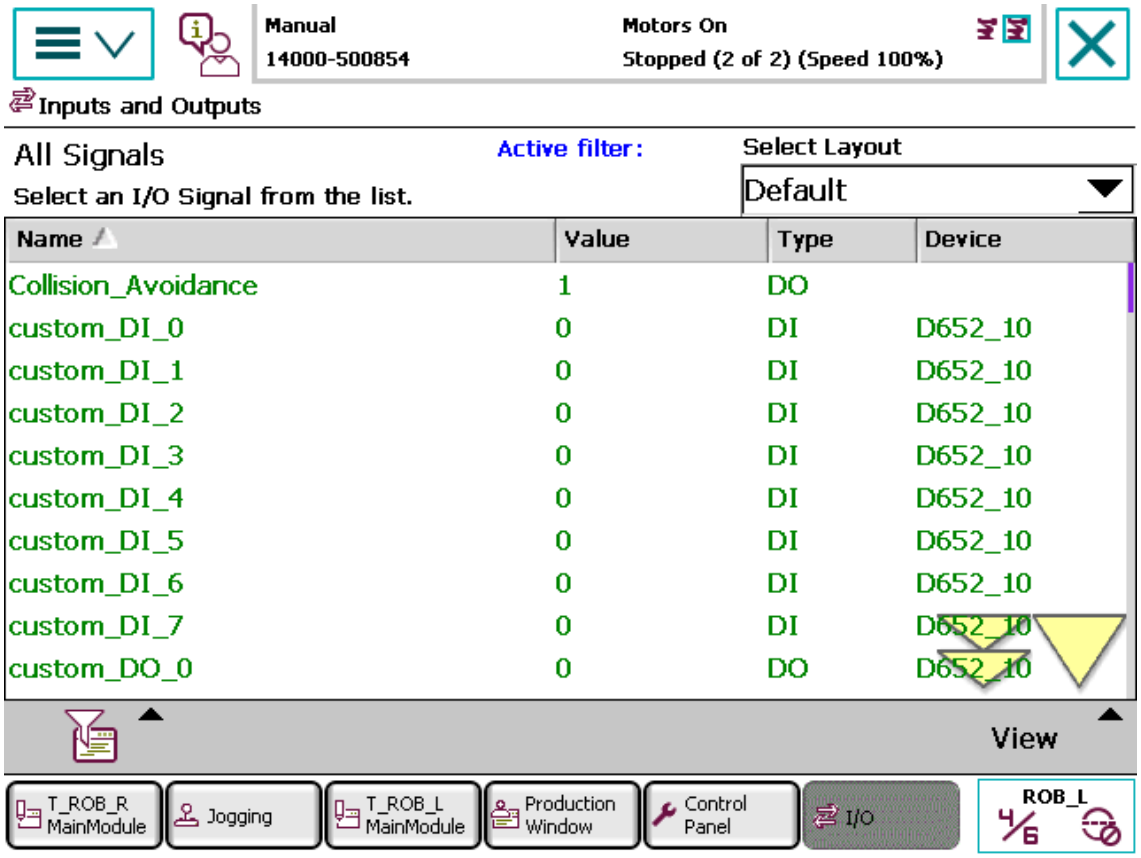
2.3 HotEdit



Kuva 14. Kuvassa HotEdit-ohjelma.

HotEdit-ohjelma on tarkoitettu tallennettujen paikkojen hienosäätämiseen. Nimensä mukaisesti tämä voidaan tehdä kuumana eli ohjelman käydessä. Pisteet valitaan hienosäädettäväksi nuolesta. Tämän jälkeen valittuihin pisteisiin voi asettaa haluttu poikkeama koordinaatistossa tai kulmana työkalussa.

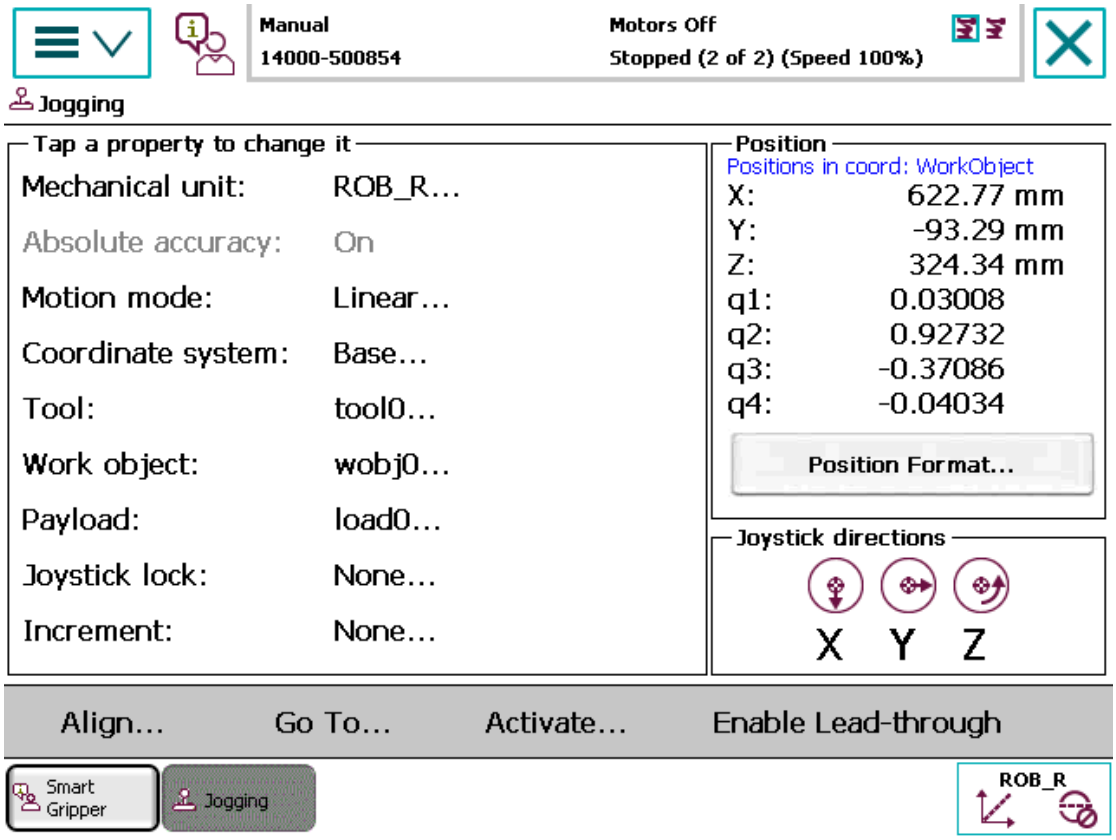
2.4 Inputs and Outputs



Kuva 15. Kuvassa Inputs and Outputs -ohjelma.

Inputs and Outputs -ohjelmassa nähdään IO-rajapinnan yhteydet. Oikealla alhaalla sijaitsee view-painike, josta valitaan mitä yhteyksiä halutaan tarkkailla. IO-yhteydet määritellään Control Panel -ohjelmalla.

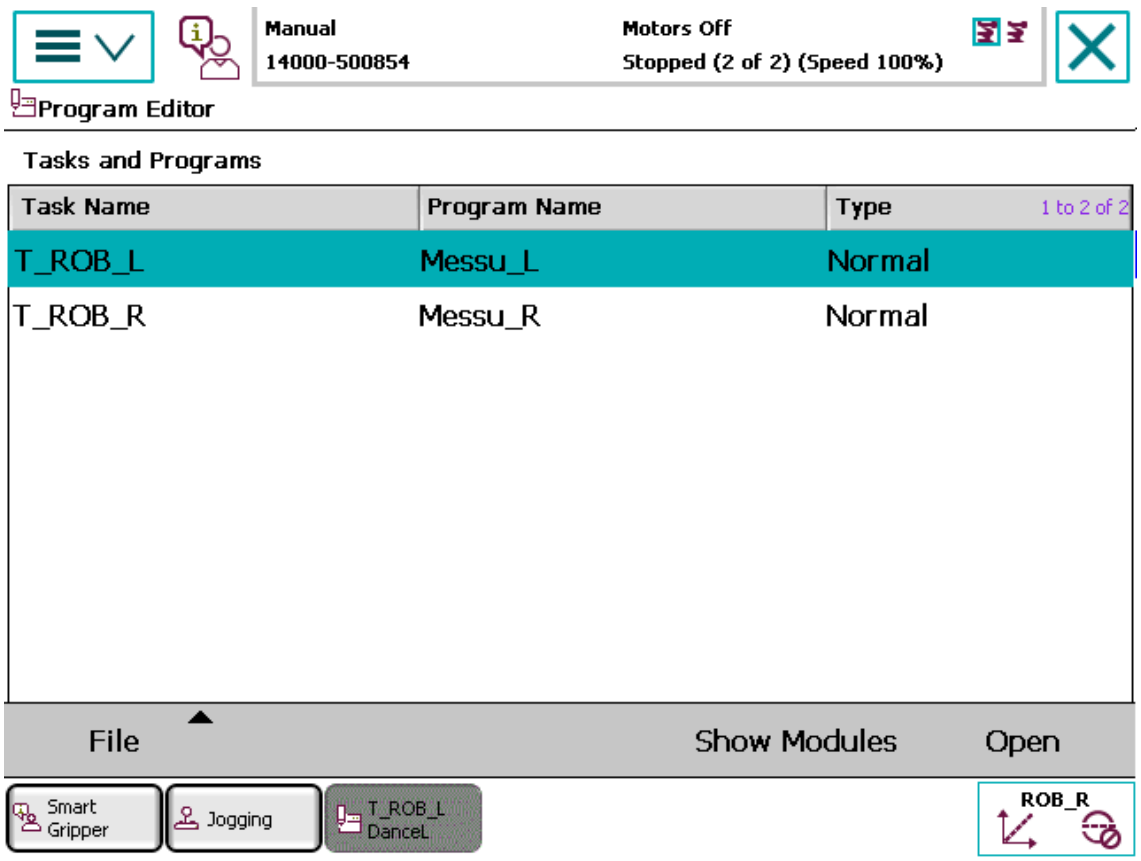
2.5 Jogging



Kuva 16. Kuvassa Jogging-ohjelma.

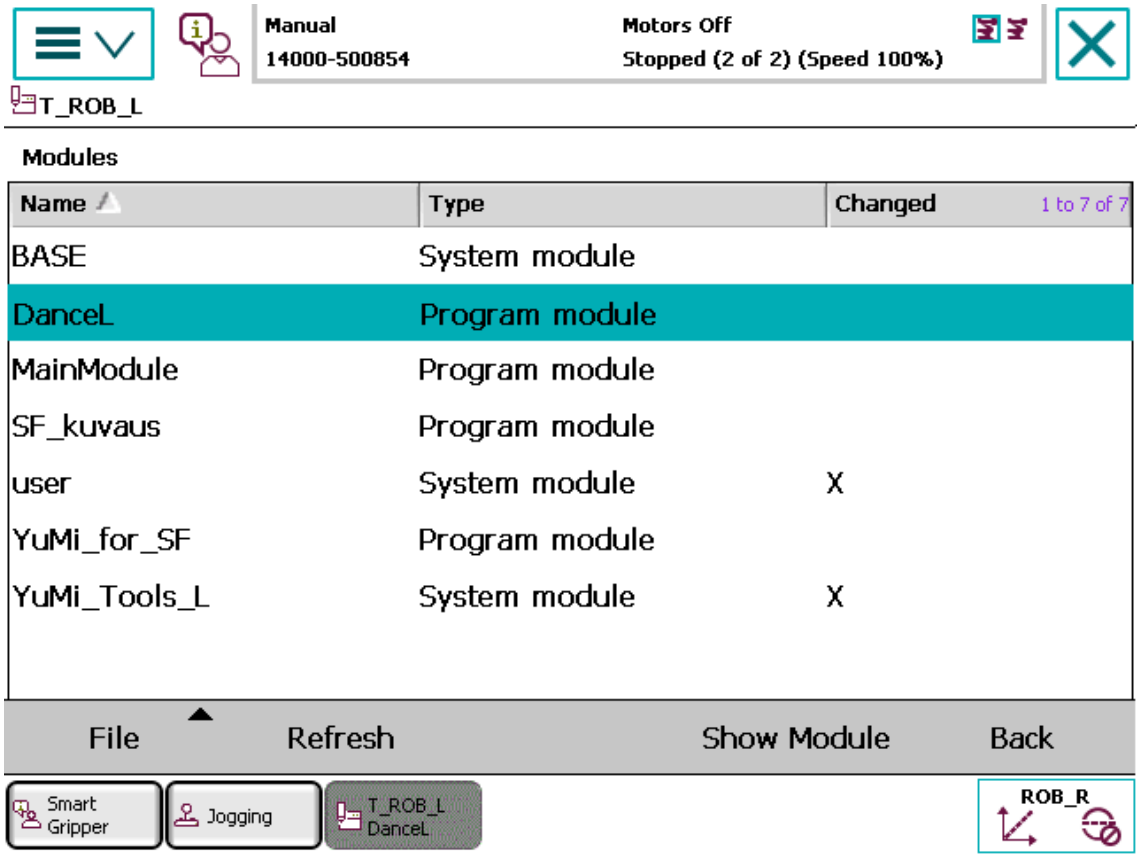
Jogging-ohjelma on tarkoitettu avuksi kobotin manuaaliseen ohjaamisen. Ennen kobotin manuaalista ajamista tulisi tutustua Liikkumistapoihin ja joystick suuntiin halutussa liikkumistavassa. Ajaminen kannattaa aloittaa hitaasti, ja jos ohjaus tuntuu liian herkältä voi käyttää etenemä ohjausta. Jotta kobottia voidaan ajaa käsiohjauksella, pitää kobotti olla manuaalitilassa ja moottorit päällä. Ohjaaminen ei onnistu, jos taustalla pyörii ohjelma tai enable lead-through on valittuna. Enable lead-throuh -painike vapauttaa lukot nivelistä, jolloin käsi ja työkalu voidaan asettaa haluttuun paikkaan fyysisesti siirtämällä. Align…-painikkeesta voi kohdistaa työkalun koordinaatisto johonkin toiseen koordinaatistoon. Tällöin työkalu liikkuu halutun koordinaatiston lähimpään z-akseliin siirtäen työkalukoordinaatiston mukanaan. Go To…-painikkeesta pääsee valikkoon, jossa on tallennetut paikat, painamalla Go To -painiketta ajaa kobotti itse valittuun paikkaa. Activate…-painike on käyttöönottoa varten.

2.6 Program Editor



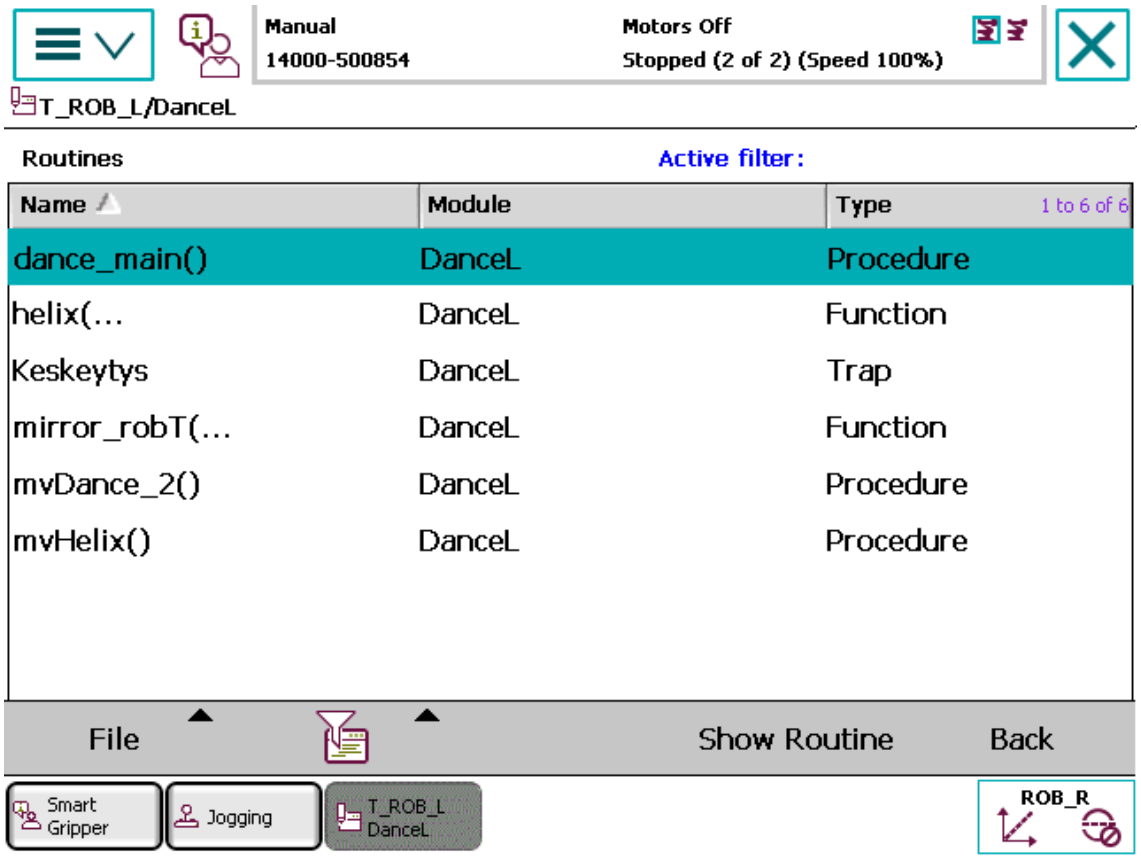
Kuva 17. Kuvassa Program editor -ohjelma, jossa tasks and programs -välilehti auki.

Program Editor on ohjelmien koodaukseen ja hallinnoitiin tarkoitettu työkalu. Program Editorissa ylempi ohjelma on vasemman käden ohjelma ja alempi oikean, uuden ohjelman aukaiseminen tapahtuu valitsemalla haluttu käsi ja painamalla File-painikkeen alta löytyvää Load Program... -painikketta. File-painikkeen alta löytyvät myös vaihtoehdot ohjelman tallentamiselle ja uuden ohjelman luomiselle. Delete Program -painikkeella voidaan sulkea haluttu ohjelma, jos olet tallentanut ohjelmasi levylle, se pysyy kyllä siellä. Show Modules -painikkeella päästään katsomaan valitun ohjelman moduuleita. Open-painikkeesta avautuu main-moduulin main-rutiini.



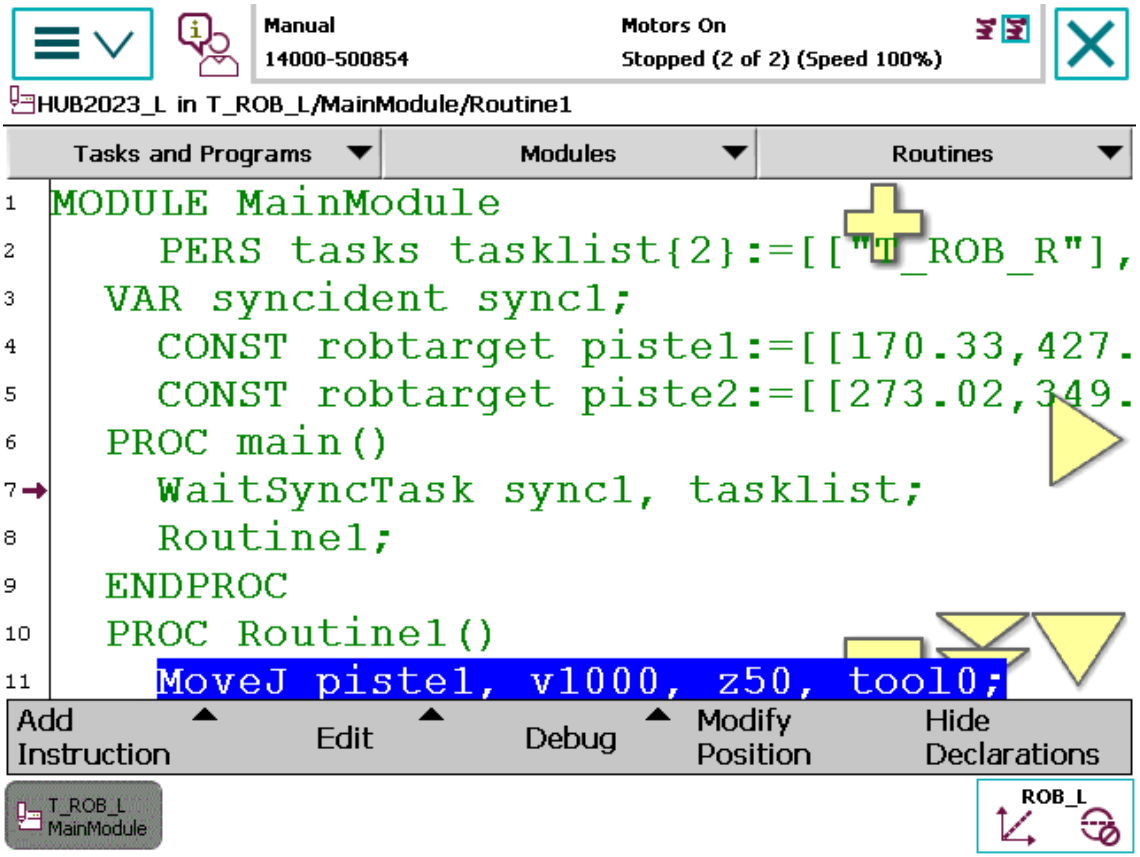
Kuva 18. Kuvassa Program editor -ohjelma, jossa modules-välilehti auki.

Modules-välilehdellä näkyvät ohjelman sisällä olevat moduulit, joita voidaan tallentaa erikseen ja ladata uusia samaan ohjelmaan. File-painikkeen alta löytyvät vaihtoehdot myös uuden moduulin luomiselle ja sulkemiselle. Change Declaration -painikkeesta voi muokata valitun moduulin nimeä ja tyyppiä. Show Module -painikkeella päästään katsomaan valitun moduulin ruutiineja.



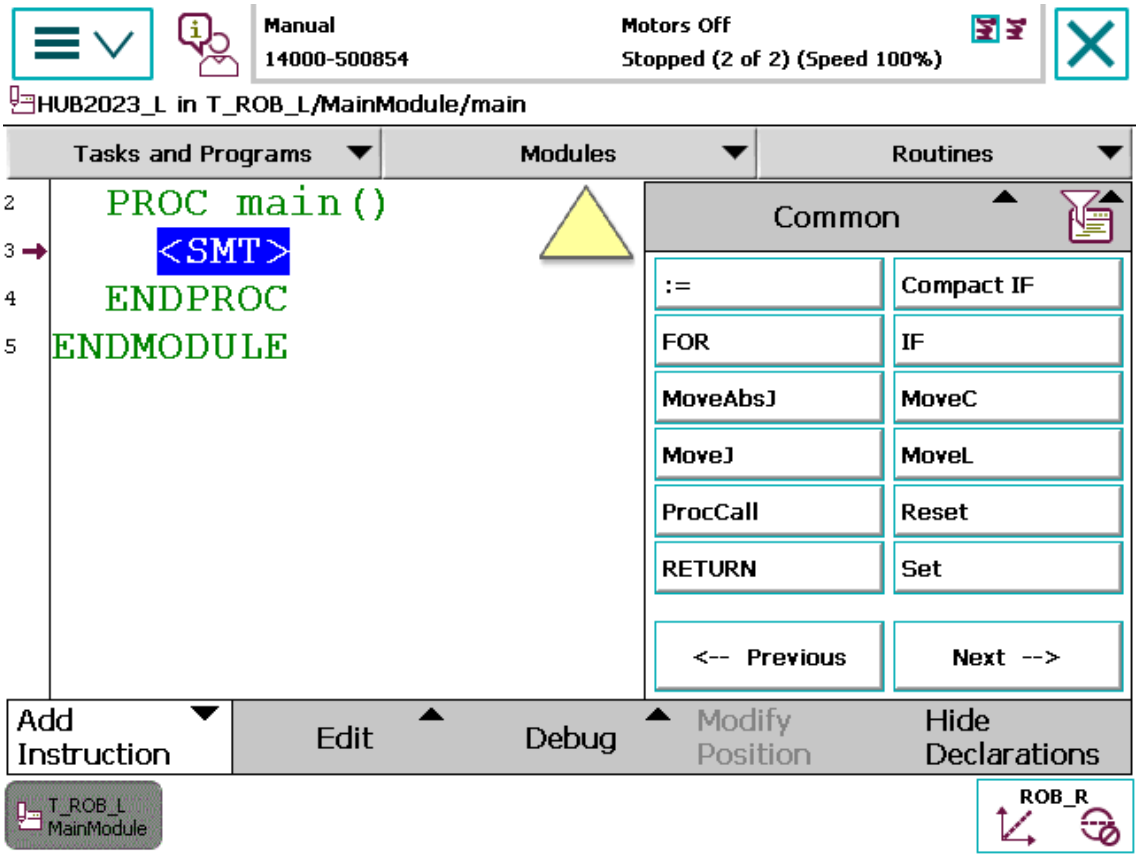
Kuva 19. Kuvassa Program editor -ohjelma, jossa routines-välilehti auki.

Routines-välilehdeltä löytyvät rutiinit, joihin itse koodi ohjelmoidaan, moduulista löytyy aina automaattisesti main-rutiini ja niitä voidaan luoda lisää. rutiineja ei voida tallentaa tai ladata muistista, mutta niitä voi kopioida ja siirtä moduulista toiseen. Main-rutiinin tarkoitus ja rutiinien tyypit on käyty sivulla 11. Show Routine -painikkeella päästää itse editor-näkymään, jossa koodi ohjelmoidaan.



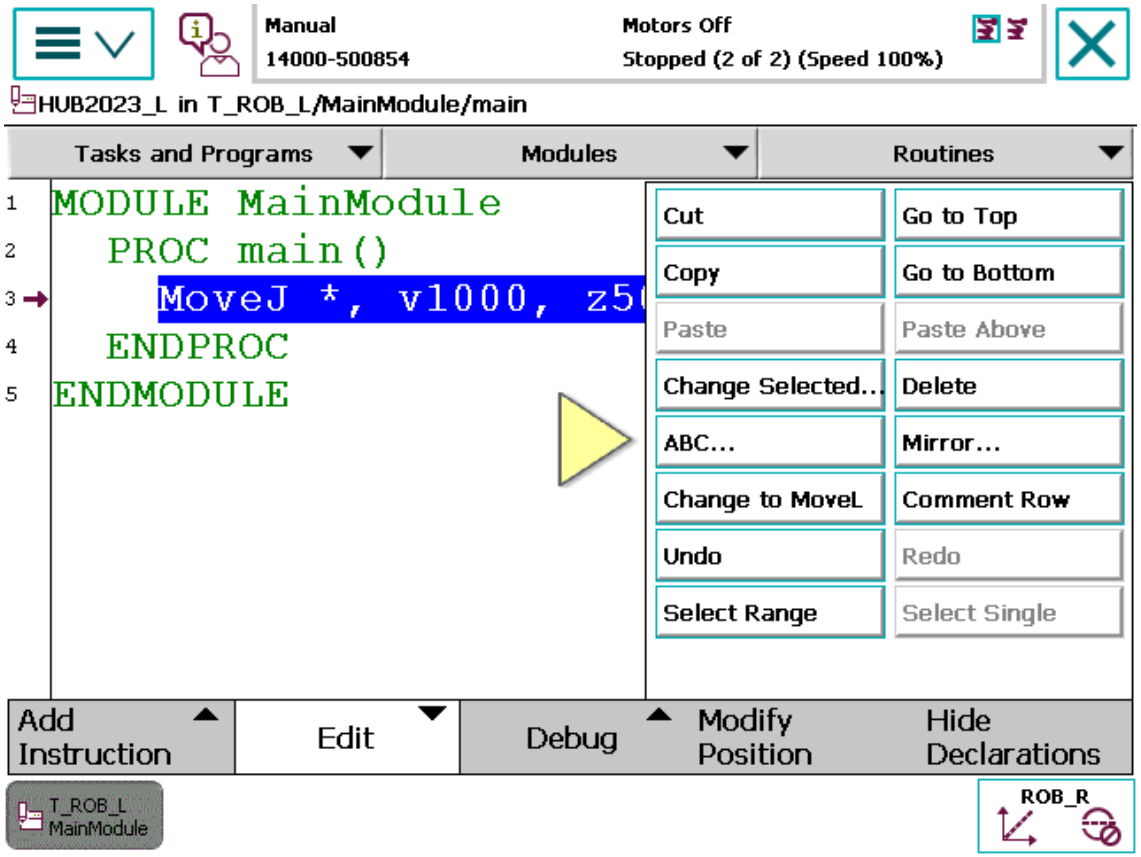
Kuva 20. Kuvassa Ohjelmointi näkymä.

Ohjelmointi näkymässä näkyy koko moduulin koodi, sitä voidaan selata keltaisilla nuoli näppäimillä. Yläpalkista voidaan siirtyä tarkkailemaan aiemmin käytyä ohjelmistorakennetta. Alapalkista löytyvät ohjelmoinnin valikot, jotka käydään seuraavaksi läpi. Hide Declarations painikkeella voidaan piilottaa kaikki muut kin valitun rutiinin koodi.



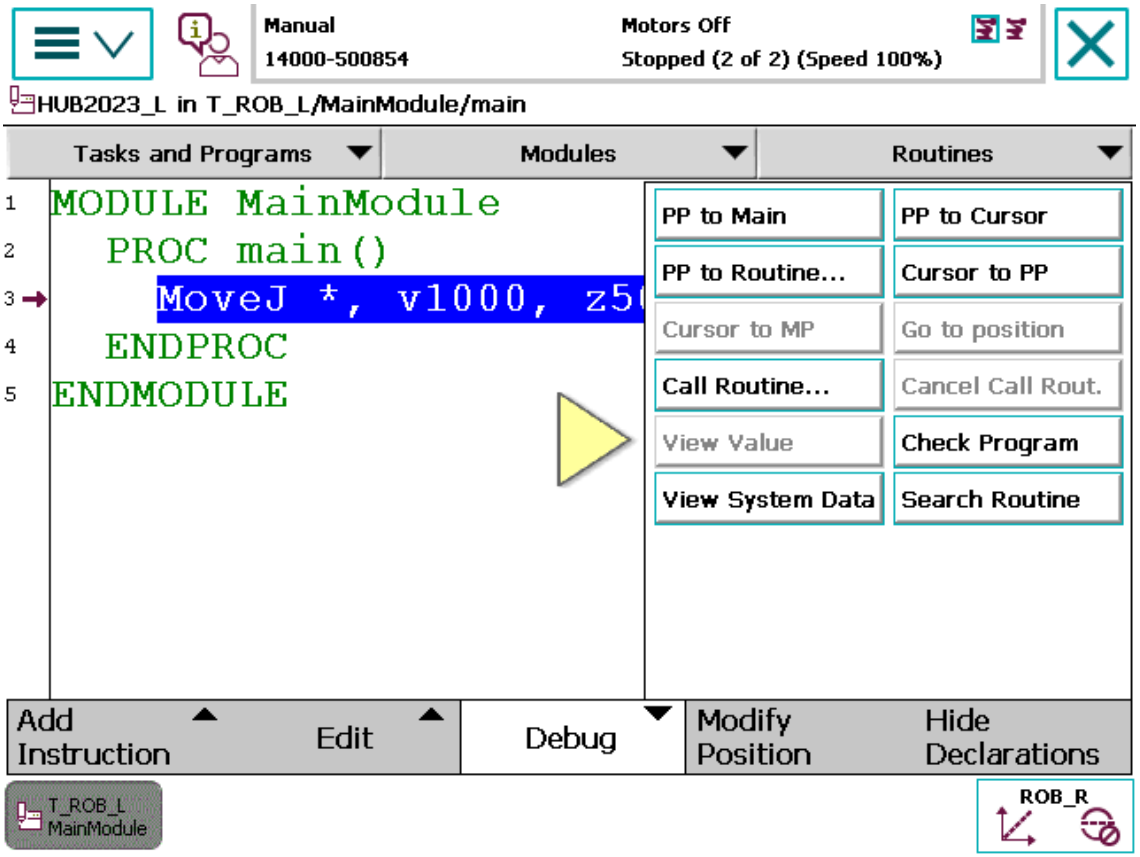
Kuva 21. Kuvassa Program editor -ohjelma, jossa main-rutiini auki ja Add Instruction -välilehti auki.

Add Instruction -välilehden kautta tapahtuu itse koodaaminen. Se tapahtuu lisäämällä komentoja sivuvalikosta. Kuvassa olevan Common-valikko voidaan vaihtaa halutuksi komentotyyppi valikoksi avaamalla alasvetovalikko sen kautta. Valikon komentoja voidaan selata Previous- ja Next-painikkeilla.



Kuva 22. Kuvassa Program editor -ohjelma, jossa main-rutiini auki ja Edit-välilehti auki.

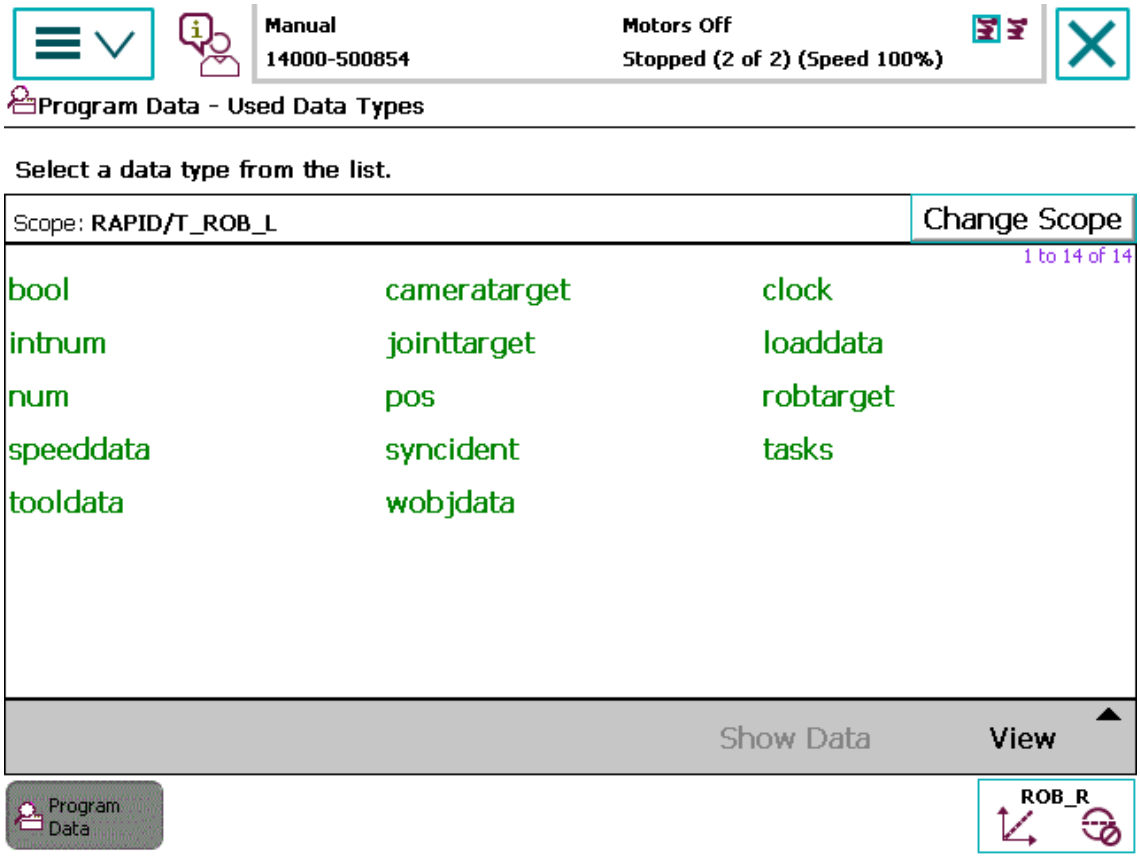
Edit-välilehti on tarkoitettu jo lisättyjen komentojen muokkaamiseen tai poistoon. Välilehdeltä löytyvät myös pikanäppäimet, joilla pääsee koodin alkuun tai loppuun. Modify Position -painikkeella voidaan muuttaa ohjelmakoodiin aikaisemmin luotu paikka siihen paikkaan, jossa kobotti on nyt.



Kuva 23. Kuvassa Program editor -ohjelma, jossa main-rutiini auki ja Debug-välilehti auki.

Debug-välilehdeltä löytyvät Program Pointerin siirtoon tehdyt komennot. Program Pointer kuvaa pistettä, jossa ohjelma on suoritushetkellä. Program Pointerin siirrettäessä halutuun komentoon, alkaa ohjelman suorittaminen tästä pisteestä, kun se käynnistetään. Välilehdeltä voidaan myös kutsua rutiini ja tarkistaa ohjelma virheiden varalta.

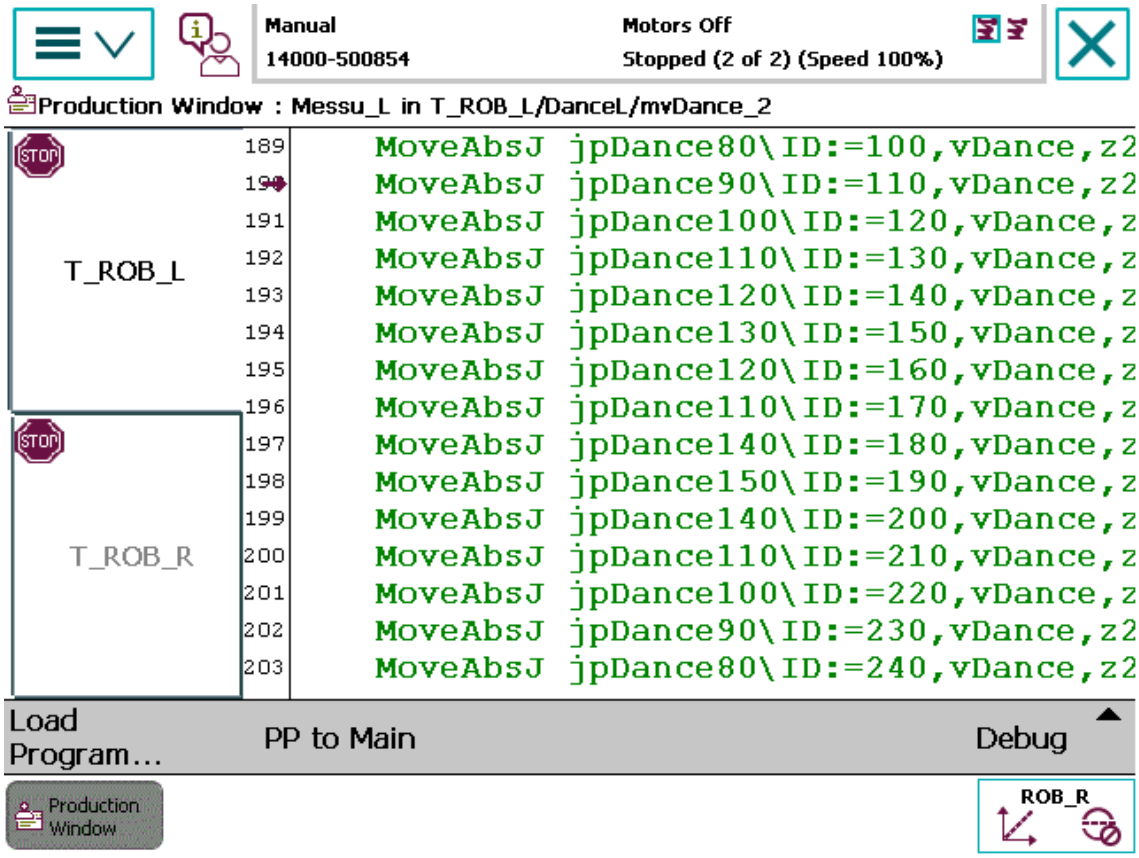
2.7 Program Data



Kuva 24. Kuvassa Program Data -ohjelman vasemman käden datatyyppejä.

Program data sisältää tiedot, joita voidaan käyttää koko ohjelman sisällä, näitä tietoja voidaan tarkastella ja muokata Program Data -ohjelmalla. Datatyypin avattua alapalkkiin tulee vaihtoehdot uuden datapisteen luomiselle ja olemassa olevien muokkaamiselle. View Data Types -painikkeella pääsee takaisin ylävalikkoon.

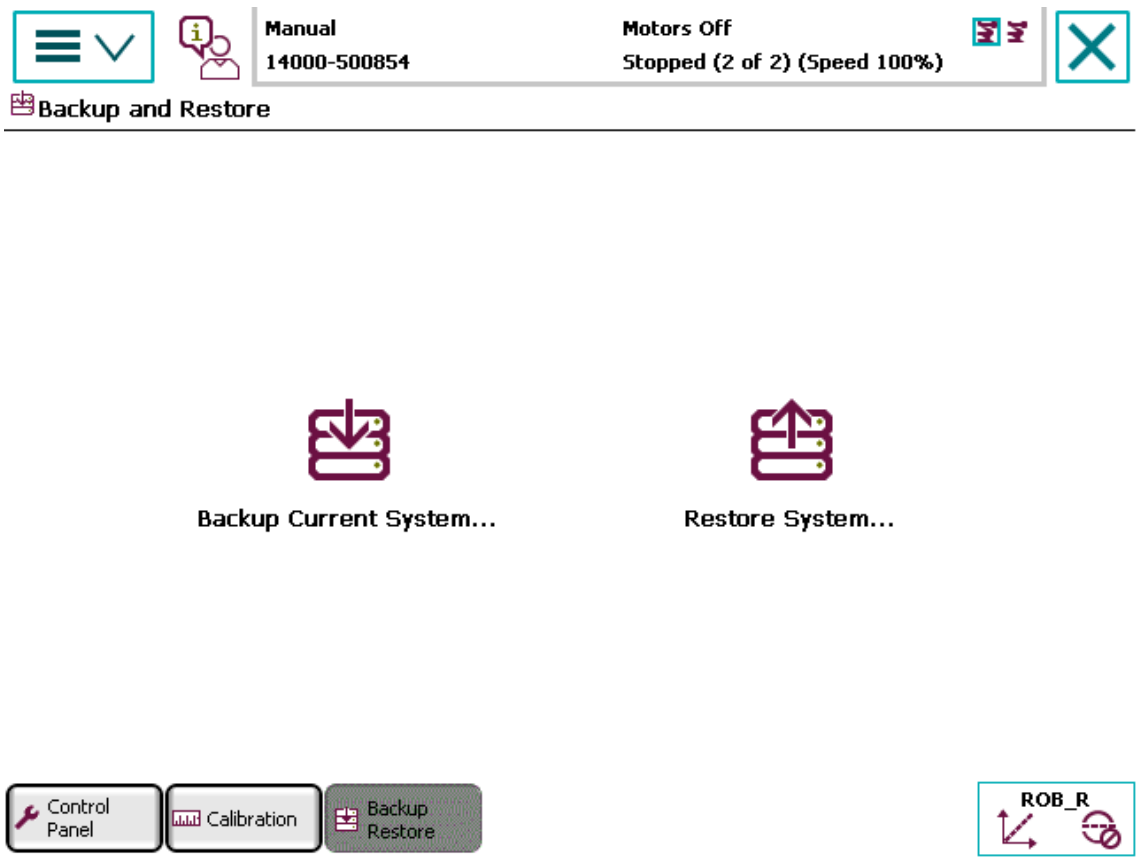
2.8 Production Window



Kuva 25. Kuvassa Production Window -ohjelma.

Production Window -ohjelmalla voidaan seurata ohjelman etenemistä riviriviltä. Program Editorista poiketen voidaan seurata molempia käsien toimintaa yhdeltä ruudulta. Load Program -painikkeella voidaan ladata ohjelma valitulle kädelle ja PP to Main -painikkeella palauttaa Program Pointer alkuun main-moduulin main-rutiiniin. Debug--valikosta löytyvät hyödylliset pikanäppäimet editor-näkymään ja Modify Position -toiminnolle.

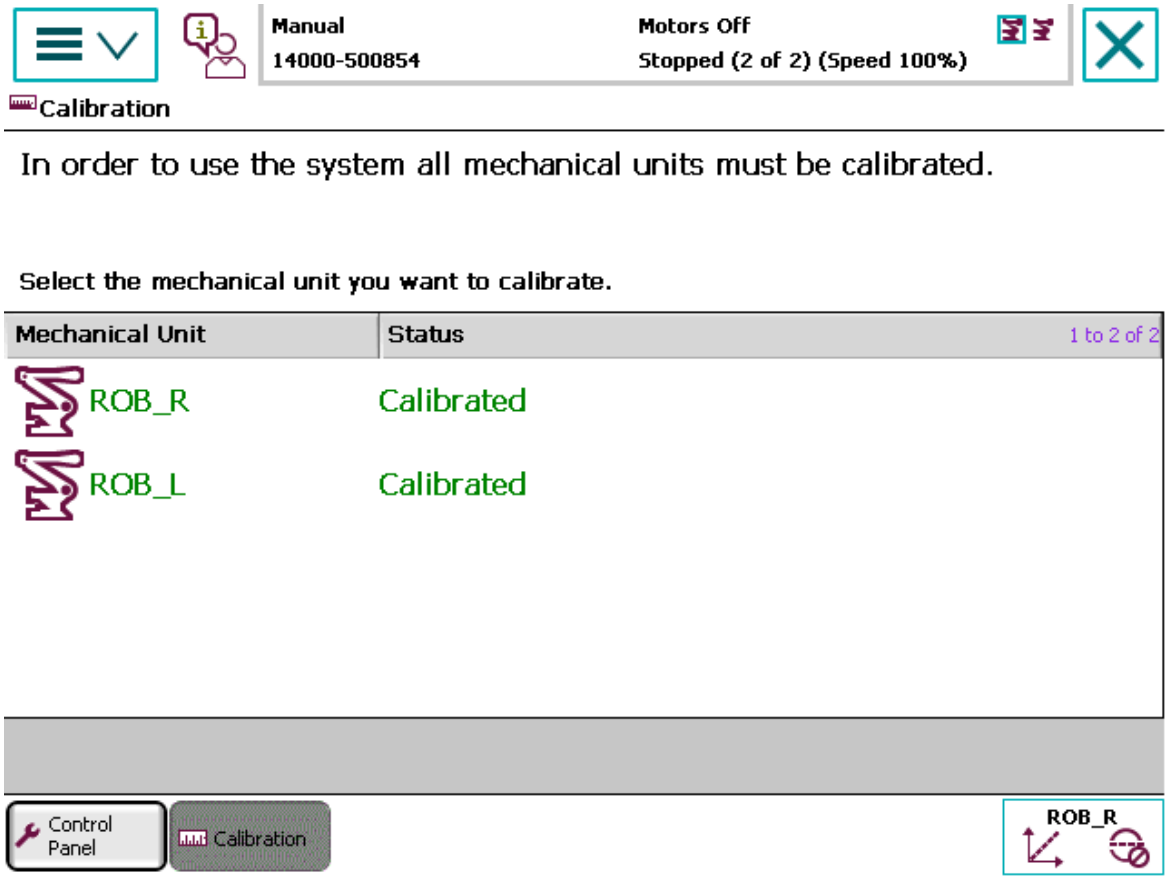
2.9 Backup and Restore



Kuva 26. Kuvassa Backup and Restore -ohjelma.

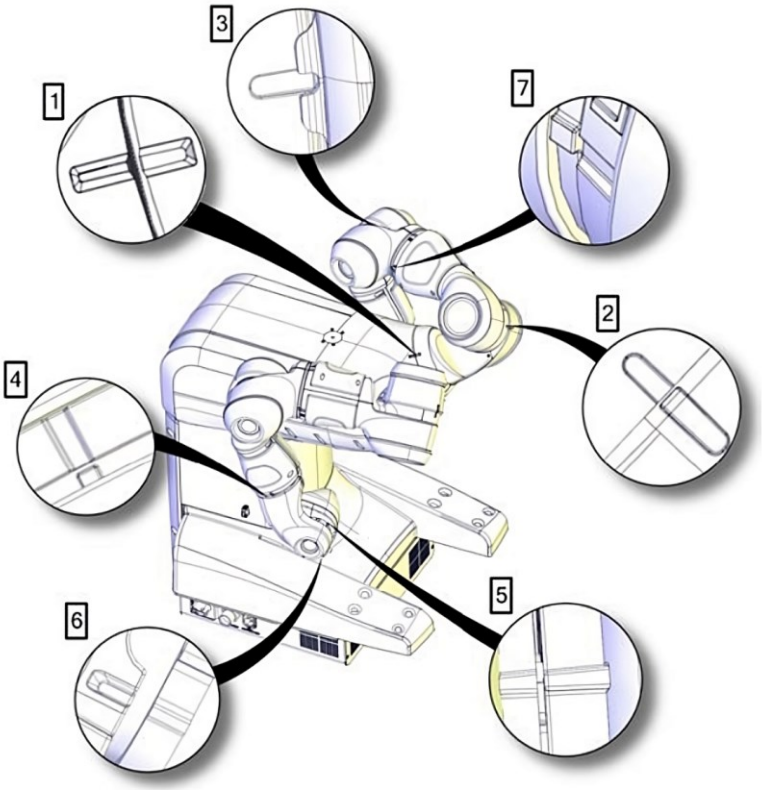
Backup and Restore -ohjelmalla voidaan luoda järjestelmäkuva myöhempää palautusta varten. Järjestelmäkuva tallennetaan levyllä olevaan BACKUP kansioon. Järjestelmäkuvaan tallentuu myös kaikki ei auki olevat ohjelmat ja asetukset.

2.10 Calibration



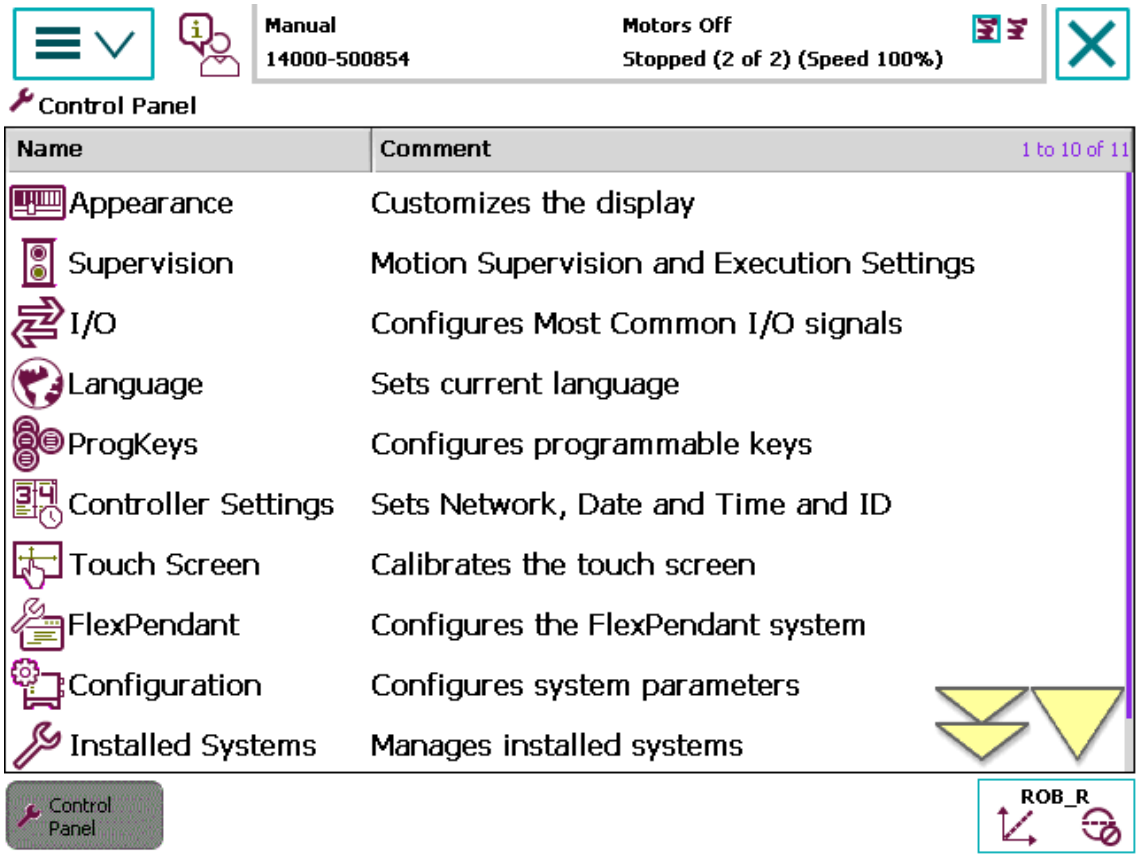
Kuva 27. Kuvassa Calibration -ohjelma.

Calibration -ohjelmaa käytetään alku käyttöönotossa tai jos kalibrointi nähdään tarpeelliseksi. Kosketusnäytön kalibrointi tehdään control paneelissa. Akselien kalibrointia varten tulee kobotti asettaa alla olevan kuvan tavoin ja ajaa hall rutiini calibration -ohjelman kautta



Kuva 28. Kuvassa kobotti kalibrointia varten olevassa asennossa.

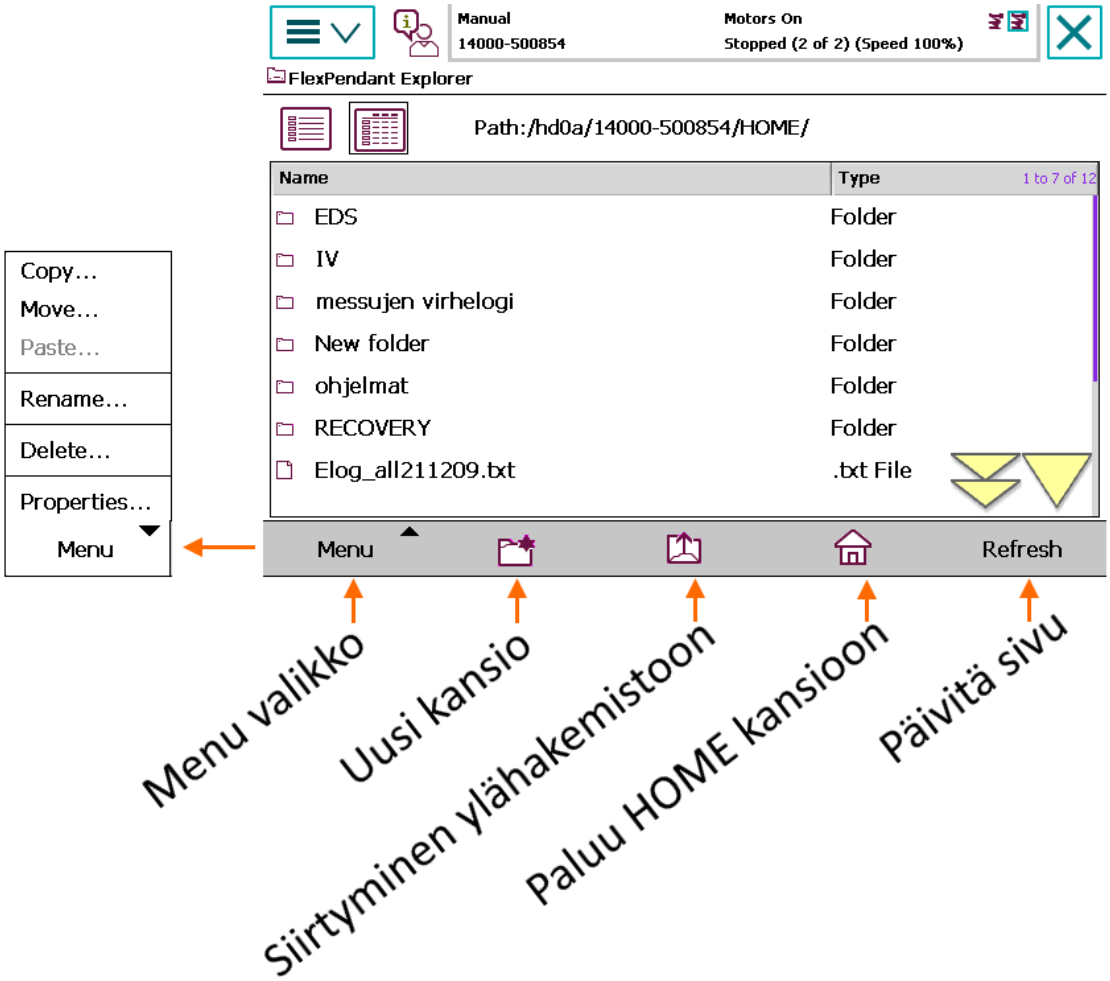
2.11 Control Panel



Kuva 29. Kuvassa Control Panel -ohjelma.

Control panel -ohjelmasta löytyvät kaikki kobotin ja pendantin mukauttamiseen liittyvät asetukset. Tässä oppaassa ei käydä läpi näiden asettamista tai kaikkia Control Panel -ohjelmasta löytyviä ominaisuuksia. Olennaiset herkkyyden muokkaaminen ja ohjelmoitavat pikänäppäimet käydään läpi sivuilla 14 ja 15.

2.12 FlexPendant Explorer



Kuva 30. Kuvassa FlexPendant Explorer, johon on merkattu nuolilla alapalkista löytyvät toiminnot.

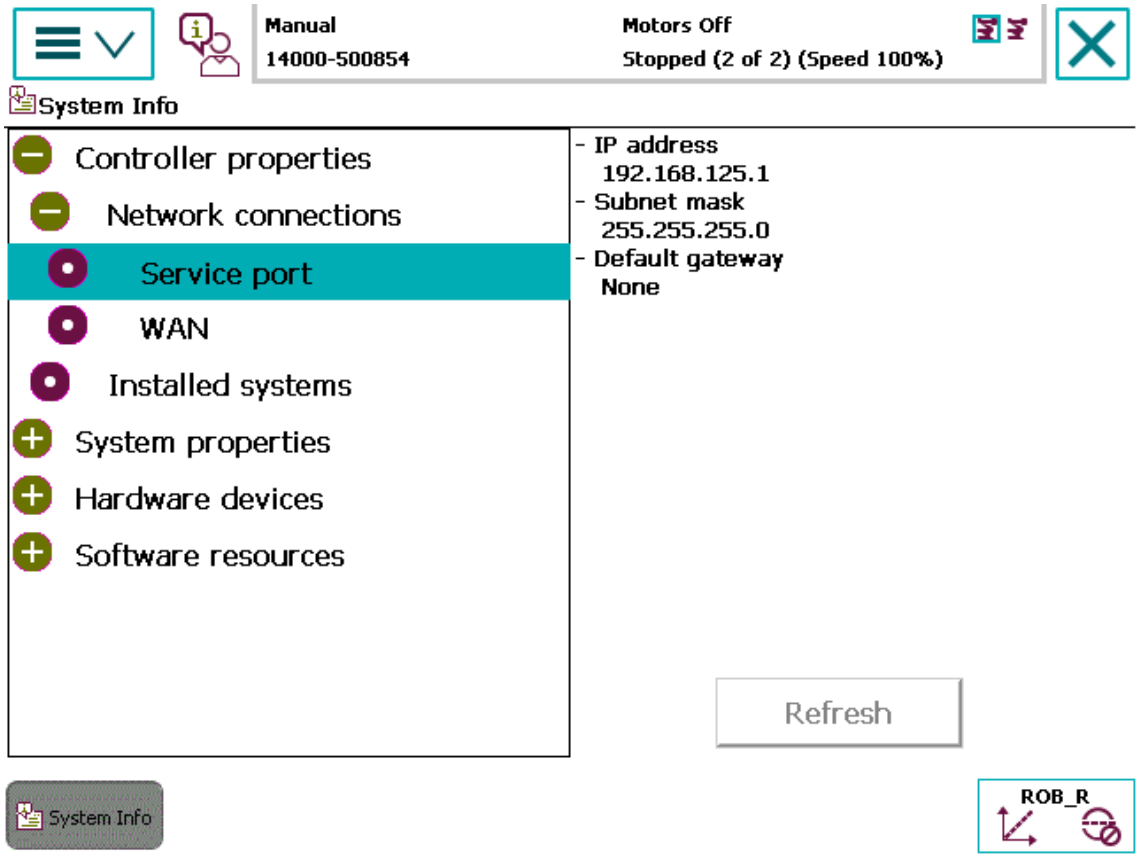
FlexPendant Explorer on tiedostonhallinta ohjelma, jolla pääsee käsiksi kaikkiin backup ja käyttömuistissa oleviin tiedostoihin.

2.13 Lock Screen

Kuva 31. Kuvassa Lock Screen.

Lock Screen -ohjelma on tarkoitettu ruudun putsaamista varten, jotta ruutua putsattaessa ei tulisi painettua tahattomasti mitään.

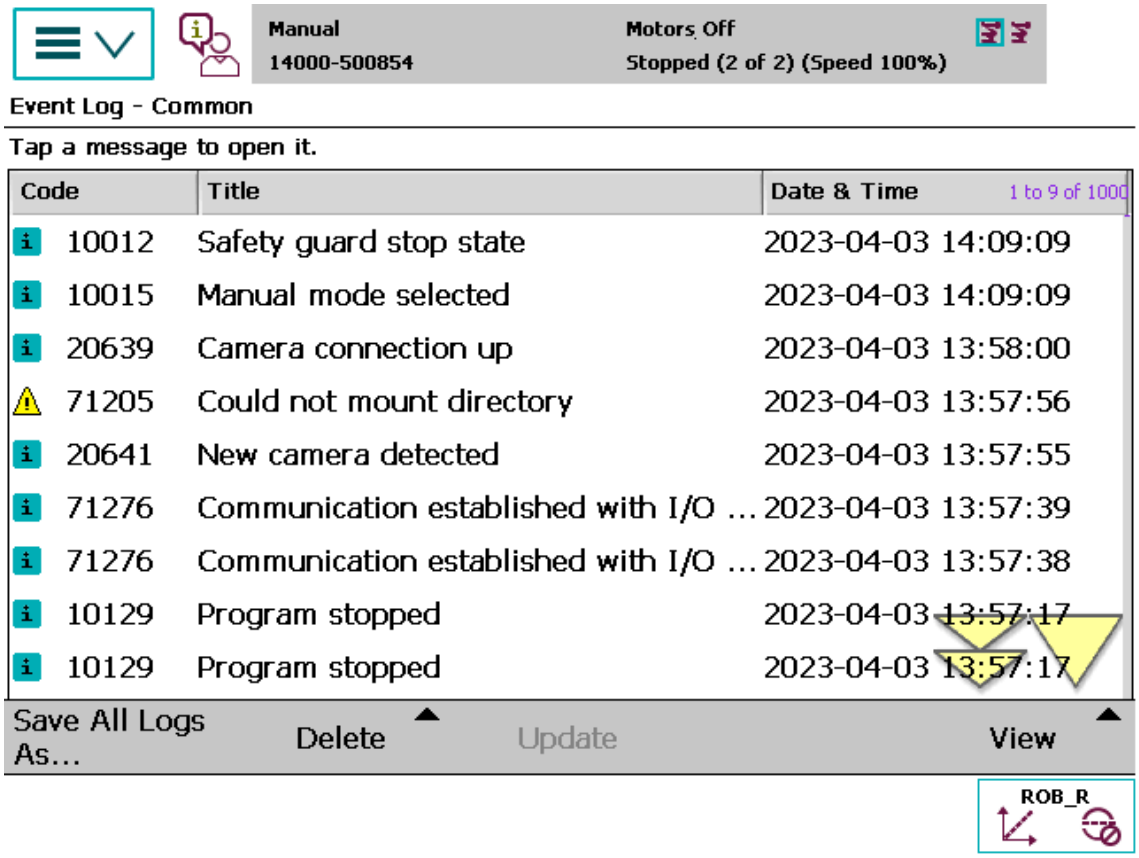
2.14 System Info



Kuva 32. Kuvassa System Info -ohjelma.

System Info -ohjelmasta löytyvät tiedot kobotista ja käyttöliittymästä.

2.15 Event Log



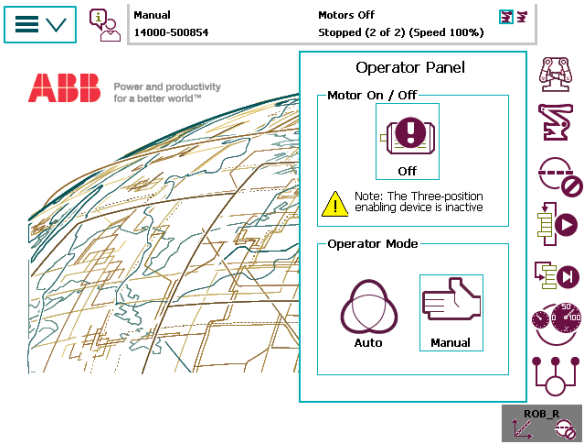
Kuva 33. Kuvassa Event Log -ohjelma.

Tapahtuma logista löytyvät poikkeavat tapahtumat aikaleimoineen. Jos tapahtumaa painaa aukeaa tapahtuma viesti, jossa on tarkempi kuvaus tapahtumasta. Tapahtumat voidaan tallentaa txt-tiedostoksi levylle ja tapahtumia voidaan poistaa logista yksi kerrallaan tai koko logi. View-valikosta voi valita vain tietyntyyppiset tapahtumat näkyväksi.

3 Nopeat visuaaliset esimerkit

3.1 Ohjelman luominen ja sen ajaminen

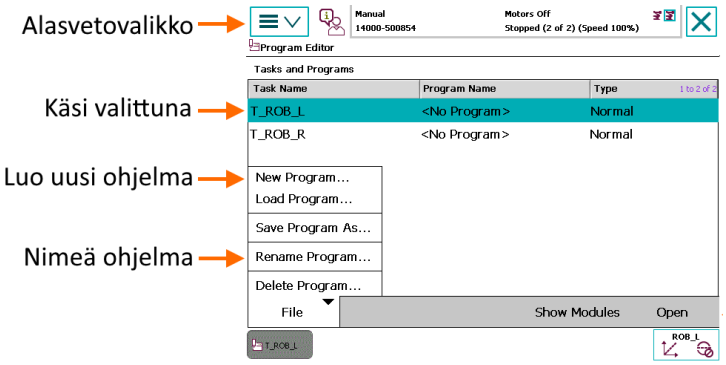
Ennen ohjelman luomista varmista pikavalikon kautta, että moottori on päällä ja kobotti on manuaalitilassa.



Vaihe 1. Avaa alasvetovalikosta Program Editor -ohjelma.

Vaihe 2. Valitse käsi ja luo uusi ohjelma painamalla New Program.

Vaihe 3. Anna ohjelmallesi nimi, valitsemallasi ohjelmasi ja painamalla Rename Program.

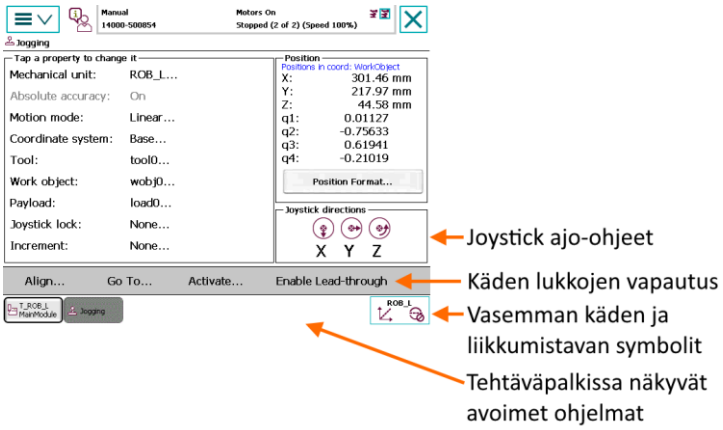


Vaihe 4. Avaa koodieditori luomallesi ohjelmalle painamalla Open.

Vaihe 5. Avaa alasvetovalikosta Jogging-ohjelma.

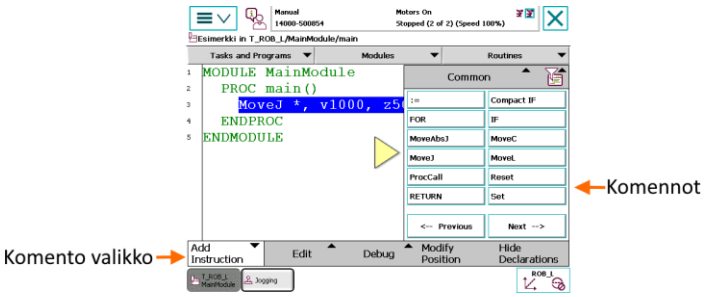
Vaihe 6. Varmista että vasen käsi on valittuna. Käden voi valita pendantin pikanäppäimellä tai pikavalikosta.

Vaihe 7. Aja käsi ensimmäiseen paikkaan käyttäen joystick-sauvaohjainta tai vapauttamalla käden lukot Enable Lead-through -painikkeella ja liikuttamalla käden käsin ensimmäiseen paikkaan.



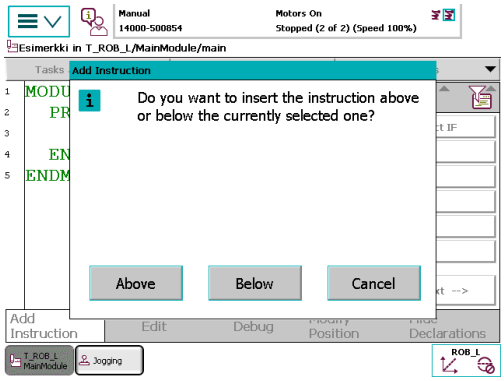
Vaihe 8. Avaa Program Editor tehtäväpalkista.

Vaihe 9. Paina Add Instruction -painiketta ja luo liikkumiskomento valitsemalla MoveJ.



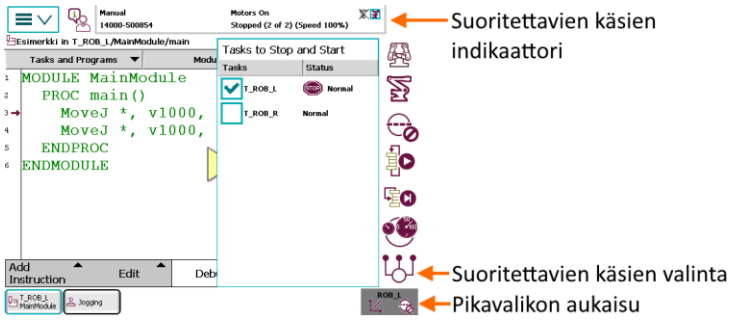
Vaihe 10. Aja tai liikuta käsi seuraavaan pisteeseen vaiheen 7 tavoin.

Vaihe 11. Lisää liikkumiskomento painamalla MoveJ. jos tietokone kysyy, laitetaanko komento edellisen ylä- vai alapuolelle, valitse Below.

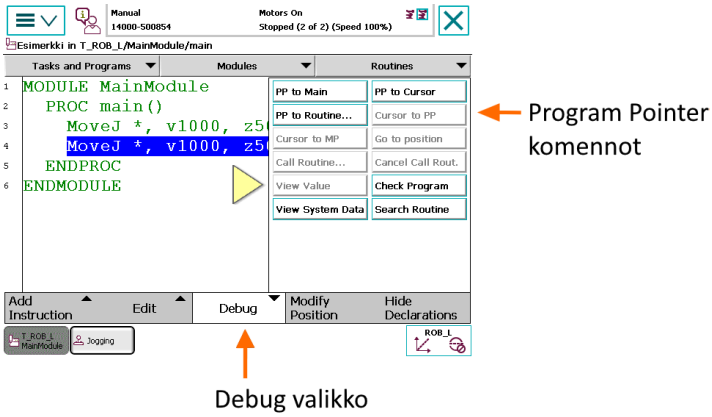


Vaihe 12. Avaa Debug-ikkuna ja paina PP to Main -painiketta. Jos painat PP to routine, valitse rutiini ja paina OK.

Vaihe 13. Avaa pikavalikosta tehtävästatus ikkuna ja varmista ettei Oikean käsi ole valittuna suoritettavaksi.

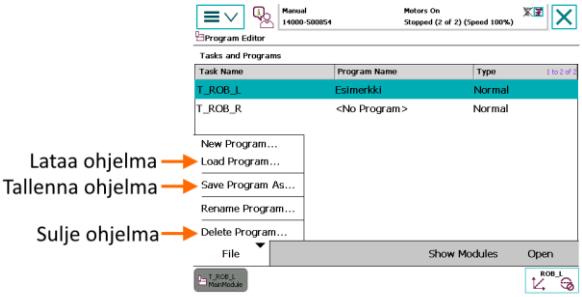


Vaihe 14. Paina käynnistä-painiketta pendantista. Ohjelmaa toistetaan, kunnes painat pysäytä-painiketta.



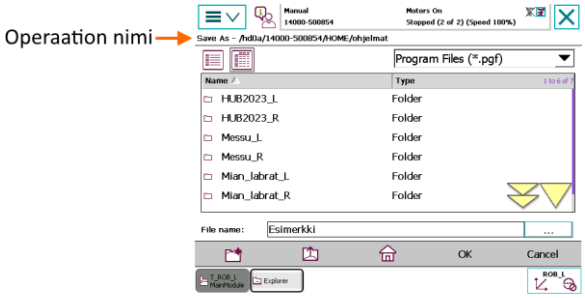
3.2 Ohjelman lataaminen/tallentaminen

File valikko



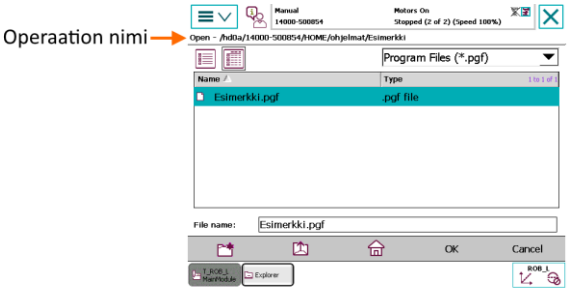
Tallentaaksesi ohjelman valitse ohjelma ja paina Save Program As. Ladataksesi ohjelman valitse käsi, jolle haluat ladata ohjelman ja paina Load Program. Sulkeaksesi Ohjelman paina Delete Program ja valitse vahvistus dialogista Confirm.

Tallentaminen



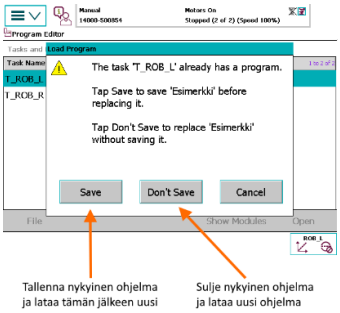
Tallennettaessa tallennetaan ohjelman kansiota, jolloin tallenna ohjelmasi aina ohjelmat kansioon. Jos korvaat vanhemman samannimisen tallennuksen uudella, valitse Overwrite dialogista OK.

Lataaminen



Kun haluat ladata ohjelman valitse ohjelman kansiosta ohjelman pgf tiedosto ja paina OK.

Ohjelman vaihtaminen



Paina Save jos haluat tallentaa nykyisen ohjelman, tällöin siirryt tallentaminen vaiheeseen. Ohjelman tallennettuasi siirryt automaattisesti lataaminen vaiheeseen. Jos et halua tallentaa nykyistä ohjelmaa paina Don't Save, jolloin nykyinen ohjelma suljetaan ja siirryt suoraan lataaminen vaiheeseen.