SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Jukka Jokela

Mobiilirobotin ohjaus REST-kommunikaatiolla

Opinnäytetyö
Kevät 2022
Automaatiotekniikan tutkinto-ohjelma

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Tutkinto-ohjelma: Automaatiotekniikka

Suuntautumisvaihtoehto: Sähköautomaatio

Tekijä: Jukka Jokela

Työn nimi: Mobiilirobotin ohjaus REST-kommunikaatiolla

Ohjaaja: Jarkko Pakkanen

Vuosi: 2022 Sivumäärä: 58

Opinnäytetyö tehtiin osana suurempaa kokonaisuutta asiakasyritykseen. Tarkoituksena oli saavuttaa toimiva case-tyyppinen ratkaisu, jossa mobiilirobotti palvelee työstökone-robottisolua. Työn toimeksiantajana toimi JTA Connection Oy, joka tarjoaa räätälöityjä tuotanto- ja logistiikka-automaatioratkaisuja lukuisille teollisuudenaloille.

Työn tavoitteena oli selvittää mahdollisuudet käyttää mobiilirobottia dynaamisessa ympäristössä, lisäksi haluttiin osoittaa robottien helppokäyttöisyys. Työssä myös olennaista on toteennäyttää mobiilirobotin hyödyt materiaalin kuljetustarkoituksessa. Lopuksi on tarkoitus perehtyä ja vakioida Beckhoff REST -kommunikaatiorakenne ja Siemens REST -kommunikaatio tulevia vastaavia toteutuksia varten.

Lopputulokseksi saatiin toimiva kokonaisuus, jossa mobiilirobotti hoitaa materiaaliliikenteen. Mobiilirobotin hallinta sekä käyttö on tehty riittävän yksinkertaiseksi. Lisäksi lopputuloksena onnistuttiin Beckhoffin ja Siemensin ohjelmoitavien logiikoiden REST-kommunikaation selvityksessä.

[1] Asiasanat: Robotiikka, Verkko-ohjelmointi, Käyttöliittymät, Käyttöönotto, Ohjelmoitavat logiikat

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Degree program: Automation Engineering

Specialization: Electrical Automation

Author: Jukka Jokela

Title of thesis: Control of a Mobile Robot Using REST Communication

Supervisor: Jarkko Pakkanen

Year: 2022 Number of pages: 58

The thesis was a part of a bigger project, where a mobile robot is used for the material handling of a machining robot cell. The thesis provider was JTA Connection Oy, which provides custom made manufacturing and logistic automation solutions to several types of industries.

The purpose of the thesis was to determine, if it is possible to use a mobile robot in highly dynamic environments. Also, it was important to create an easy to use interface for the control of the mobile robot and indicate the benefits of using mobile robots in material handling. Additionally, the thesis researched the possibility to use the programmable logics made by Beckhoff and Siemens in REST communication.

In the end, a fully functional system was achieved, where a mobile robot is used for the material handling needs of the robot cell. The mobile robot control was made simple enough, so new operators can adopt it quickly. Additionally, also the REST communication between the logics of Beckhoff and Siemens and the mobile robot was successful.

[1] Keywords: robotics, web programming, user interfaces, introduction, programmable logic controllers

SISÄLTÖ

Kuva-, kuvio- ja taulukkoluettelo

Käytetyt termit ja lyhenteet

Käyttöliittymä	Käyttöliittymä osa sovellusta tai järjestelmää, sen kautta ihmisen on mahdollista syöttää tai vastaanottaa tietoa.
REST	Representational State Transfer eli REST on verkkoarkkitehtuuri-tyyli, jonka avulla on laitteiden mahdollista keskustella verkon yli.
URI	Uniform Resource Identifier eli URI tarkoittaa merkkijonoa, joka osoittaa johonkin tallennettuun resurssiin.

1 JOHDANTO

1.1 Työn tausta

Opinnäytetyö käsittelee, kuinka työstökonepajassa voidaan automatisoida materiaalin kuljetusta ja nostaa automaatioastetta. Työssä selvitetään mahdollisuutta käyttää mobiilirobottia alati muuttuvassa ympäristössä. Tarkoituksena on lisäksi selvittää, kuinka helpoksi mobiilirobotin ohjaus on mahdollista tehdä. Työssä hyödynnetään REST-arkkitehtuurin rajapintaa, jolla muodostetaan yhteys käyttöliittymän ja mobiilirobotin välillä.

1.2 Työn tavoite

Työn tavoitteena on selvittää mahdollisuudet käyttää mobiilirobottia dynaamisessa ympäristössä sekä luoda esimerkkitoteutus, jota voidaan esittää ratkaisuna myynnin tueksi. Olennaisena tavoitteena on, että mobiilirobottia on mahdollista kenen tahansa käyttää hyödykseen. Työssä myös olennaista on osoittaa mobiilirobotin hyödyt materiaalin kuljetustarkoituksessa, mikä vähentää trukkiliikennettä ja vapauttaa työntekijät muihin tehtäviin.

Lisäksi on tarkoitus perehtyä ja vakioida Beckhoff REST -kommunikaatiorakenne ja Siemens REST -kommunikaatio tulevia vastaavia toteutuksia varten.

Opinnäytetyö oli osa isompaa projektia, jonka tarkoituksena oli luoda toimiva konsepti modulaarisesta kappaleentyöstöstä.

1.3 Työn rakenne

Opinnäytetyön rakenne alkaa ensimmäisessä luvussa johdannosta. Johdannossa esitellään työn tausta ja tavoite sekä esitellään yritys, jolle opinnäytetyö tehtiin. Toisessa luvussa käsitellään hieman robotiikan teoriaa ja erotellaan mobiilirobotteja toisistaan. Luvussa käsitellään myös mobiilirobottien turvallisuuteen liittyen skannereita. Kolmas luku käsittelee tietoteknistä teoriaa liittyen työhön ja sen toteutukseen.

Neljäs luku käsittelee, miten työn toteutus lähti liikkeelle ja mitä alkuvalmisteluita huomioitiin. Viidennessä luvussa paneudutaan mobiilirobotin käyttöönottoon. Samalla tarjotaan kevyt katsaus robotin kartoitukseen ja tehtäväarkenteeseen. Kuudes luku käsittelee työn toteutuksen eri osa-alueita. Kuudennessa luvussa käsitellään myös ohjelmoitavien logiikoiden REST-kommunikointia.

Luvussa seitsemän käydään läpi työn tuloksia. Lopuksi viimeisessä kahdeksannessa, luvussa on pohdinta ja yhteenveto tehdystä työstä.

1.4 Yritysesittely

JTA Connection tarjoaa räätälöityjä tuotanto- ja logistiikka-automaatioratkaisuja lukuisille teollisuudenaloille (JTA Connection, i.a.). JTA Connection tekee esimerkiksi robottisoluja, kuljettimia sekä kokonaisia tuotanto- ja kokoonpanolinjoja ympäri maailmaa. Tämän lisäksi JTA Connection tarjoaa tuotantoprosessien automatisointiin suunnittelupalveluita, ja automaatiolaitteiden asennus- ja huoltotyöt kuuluvat toimenkuvaan.

JTA Connection Oy:ssä työskentelee 125 henkilöä, ja yritys toimii kolmella eri paikkakunnalla (JTA Connection, i.a.). Yrityksen päätoimipiste on Tampereella ja tämän lisäksi toimipisteitä on myös Porvoossa ja Ihodessa. JTA on toteuttanut automaatioprojekteja teollisuuden parissa jo yli 20 vuotta.

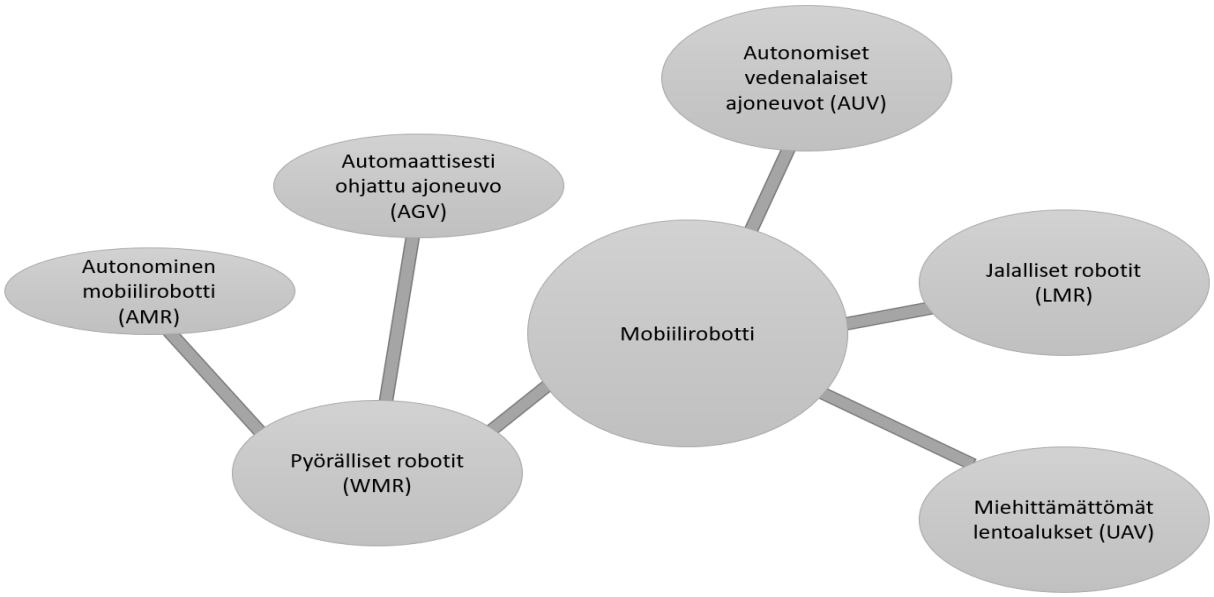
2 ROBOTIIKKA

2.1 Robottien jaottelu

Robotteja käytetään monipuolisesti laajan kirjon teollisuussovelluksissa (Radhakrishnan, 2015, s. 373–375). Työ, joka vaatii toistoja, tarkkuutta, pitkäjänteisyyttä, nopeutta ja luotettavuutta, voidaan tehdä paremmin robotilla kuin tavallisen ihmisen voimin. Tästä syystä teollisuudessa useampi työvaihe toteutetaan nykyään robotilla ihmisen sijaan. Teollisuusrobotit ovat monimutkaisia teknisiä järjestelmiä, jotka sisältävät useita toisiinsa yhteydessä olevia alijärjestelmiä, jotka toimivat harmoniassa robotin fyysisen ympäristön kanssa. Jokainen alijärjestelmä suorittaa omaa tarkkaan määriteltyä toimintoaan ja mahdollistaa teollisuusrobotin toiminnan.

2.1.1 Mobiilirobotit

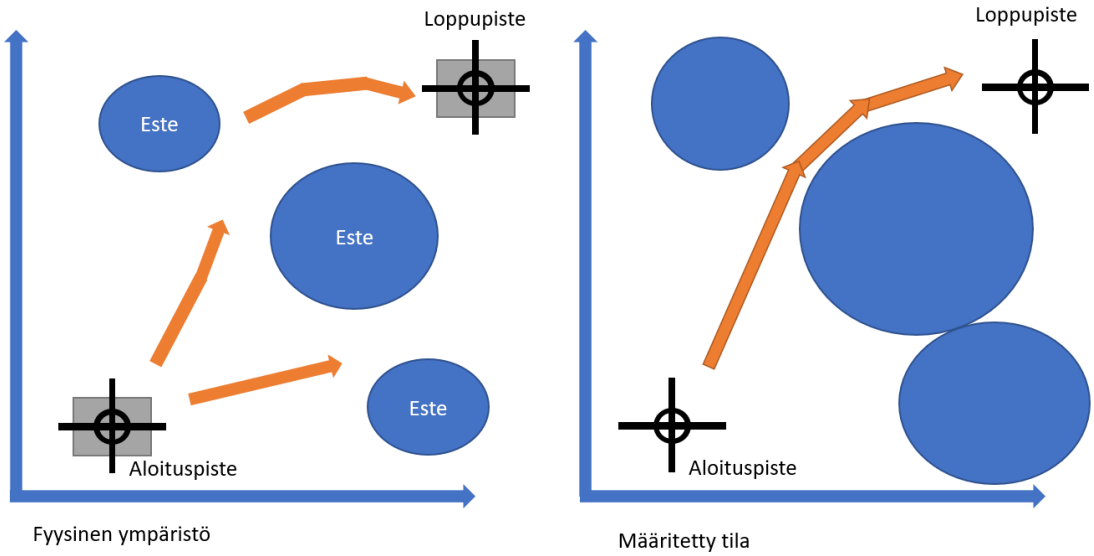
Mobiilirobotin olennainen tunnusmerkki on kyky siirtyä paikasta toiseen autonomisesti, ilman ihmisen ulkopuolista avustusta (Tzafestas, 2014, s. 1). Mobiilirobotit voidaan jakaa kolmeen eri pääryhmään robottityyppejä, joista ensimmäinen jaetaan vielä kahteen aliryhmään. Ensimmäisenä ryhmänä on pyörälliset mobiilirobotit, jotka ovat yleisin mobiilirobottityyppi suhteellisen matalan mekaanisen monimutkaisuuden ja virrankulutuksen takia. Toisena tyyppinä on jalalliset mobiilirobotit. Kolmas tyyppi on miehittämättömät lentoalukset ja neljäntenä on autonomiset vedenalaiset robotit. Tämä kyky liikkua vapaasti ennalta määrittelemättömässä ympäristössä antaa erilaisia mahdollisuuksia käyttökohteista ja tarkoituksista, toisin kuin tavalliset teollisuusrobotit, joiden liikkuminen on rajoitettua. Kuvio 1 havainnoi tätä edellä mainittua jaottelua.



Kuvio 1. Mobiilirobottien jaottelu (Tzafestas, 2014, s. 1).

Kuten kuviosta 1 voidaan huomata, mobiilirobotti-käsite pitää sisällään laajan kokonaisuuden erilaisia mahdollisia robotteja. Olennaista on valita sopiva robotti omaan käyttötarkoitukseen.

Useimmissa moderneissa mobiiliroboteissa robotin omat ohjelmoitavat ohjaimet suorittavat robotin ohjauksen (Kagan ym., 2020, s. 10). Ohjaimet seuraavat samoja toteutusperiaatteita kuin käsivarsirobotin ohjaimissa, mutta sen sijaan että ne ohjaisivat käsivarren liikkeitä, ohjaimet vastaavat robotin liikkeistä ympäristössä. Liikkeen suunnittelu vaatii, että robotilla on tiedossa polku ja liikerata ympäristössä siten, että robotti välttää yhteentörmäykset esteiden kanssa ja toisten liikkuvien asioiden kanssa. Tämän lisäksi robotilla tulee olla tiedossa euklidisen geometrian tila karteesisessa koordinaatistossa, koska se mielletään yleensä fyysiseksi tilaksi. Yleisellä tasolla liikkeen suunnittelussa mobiilirobotti ottaa huomioon erilaisia tehtäviä. Perinteisin tehtävä on polun määrittely annetusta alku- ja loppupisteestä, ilman yhteentörmäyksiä esteiden kanssa.



Kuvio 2. Fyysisen ja määritellyn tilan erot (Kagan ym., 2020, s. 11).

Kuvio 2 havainnoi ongelmaa, joka tunnetaan pianon siirtäjä ongelmana (Kagan ym., 2020, s. 11). Ongelmassa monimutkainen kappale pitäisi siirtää aloituspisteestä loppupisteeseen tilassa, jossa on erilaisia esteitä. Ensimmäisessä vaiheessa siirrettävä kappale kutistetaan pisteeksi ja esteiden kokoa kasvatetaan vastaavasti. Toisessa vaiheessa, mikäli reitti löytyy alku- ja loppupisteen välillä, se määritellään. Tämä ongelma havainnollistaa liikkeen suunnittelua tunnetussa ympäristössä, kun seurataan globaalia koordinaattimallia.

2.2 AGV ja AMR

Mobiilirobottien välillä on olennainen erottelu siinä, kuinka ne toimivat ja vuorovaikuttavat ympäristönsä kanssa (Oitzman, 2021). Autonomiset mobiilirobotit (AMR) eroavat automaattisista ohjatuista ajoneuvoista (AGV) siten, että autonomisella mobiilirobotilla on vapaus liikkua ympäristössään ja suunnitella reittiään samanaikaisesti kun sitä suoritetaan. Esimerkkinä siitä, kuinka AGV- ja AMR-käsite voidaan erottaa on, että ajatellaan AGV junana ja AMR henkilöautona. Juna ja AGV seuraavat samoja ennalta määriteltyjä raiteita ja reittejä paikasta toiseen. Henkilöauto ja AMR lähtökohtaisesti valitsevat lyhyimmän reitin kohteeseensa, mutta voivat muuttaa reittiään huomattuaan reitillään esteitä.

2.2.1 AGV

AGV eli Automated Guided Vehicle on ajoneuvo, joka on varustettu automaattisella ohjausjärjestelmällä ja pystyy seuraamaan sille ennalta määrättyä reittiä (Shivanand ym., 2006, s. 61–63). AGV-robotteja on ollut olemassa 1950-luvulta asti, mutta vasta 1970-luvun jälkeen teknologia on kehittynyt riittävästi, jotta AGV:n potentiaali saavutettiin. AGV-robotin ohjaamiseen vaaditut reitin määrittelyt voidaan toteuttaa usealla tavalla. Esimerkkeinä tästä ovat elektromagneettiset kaapelit upotettuna lattian sisään, robotti seuraa niitä sokeasti. Lisäksi laser ja heijastin paikannin järjestelmällä tai kiskoilla kulkeva robotti sopii AGV määritelmään.

AGV-järjestelmällä on minimaalinen oma “älykkyyskapasiteetti” ja kykenee seuraamaan vain yksinkertaisia ohjelmoituja toimintoja (Mobile industrial Robots (MiR), i.a.-a). AGV-järjestelmät on rajoitettu seuraamaan niille määriteltyjä reittejä, jotka yleensä on asennettu tehtaan lattiaan. Tämä tarkoittaa, että yleensä järjestelmä on suhteellisen kallis ja suuritöinen. Sen lisäksi käyttömahdollisuudet ovat rajoittuneet yhteen tehtävään, joka sille on alun perin luotu. Muutokset ovat yksinkertaisesti liian kalliita ja häiritseviä tuotannolle, jotta se olisi kustannustehokas.

2.2.2 AMR

Kuten aiemmin on mainittu, autonominen mobiilirobotti (AMR) on robottityyppi, joka kykenee ymmärtämään ja kulkemaan ympäristössään itsenäisesti (Intel, i.a.). AMR on varustettu monimutkaisella kokoelmalla antureita, tekoälyä, koneoppimista ja laskentaa reitin suunnitteluun ja ympäristönsä havainnointiin sekä sen navigointiin. Koska AMR on varustettu kameroin ja anturein, pystyy se väistämään yllättävät esteet, kuten pahvilaatikkokasan tai ihmisjoukon, ja tarpeen mukaan muuttamaan reittiään esteen kiertämiseksi ja määränpäänsä saavuttamiseksi.

Alkuvalmistelut, jotka AMR vaatii toimiakseen uudessa paikassa, ovat joko pohjapiirros tiloista, joissa robotti tulee toimimaan, tai vaihtoehtoisesti robotti voi laserskannerita käyttäen kartoittaa tilat (MiR, i.a.-b). Tämän jälkeen robotille osoitetaan kartasta paikat, joihin sen halutaan kulkevan. Näiden kahden vaiheen jälkeen robotti pystyy kulkemaan

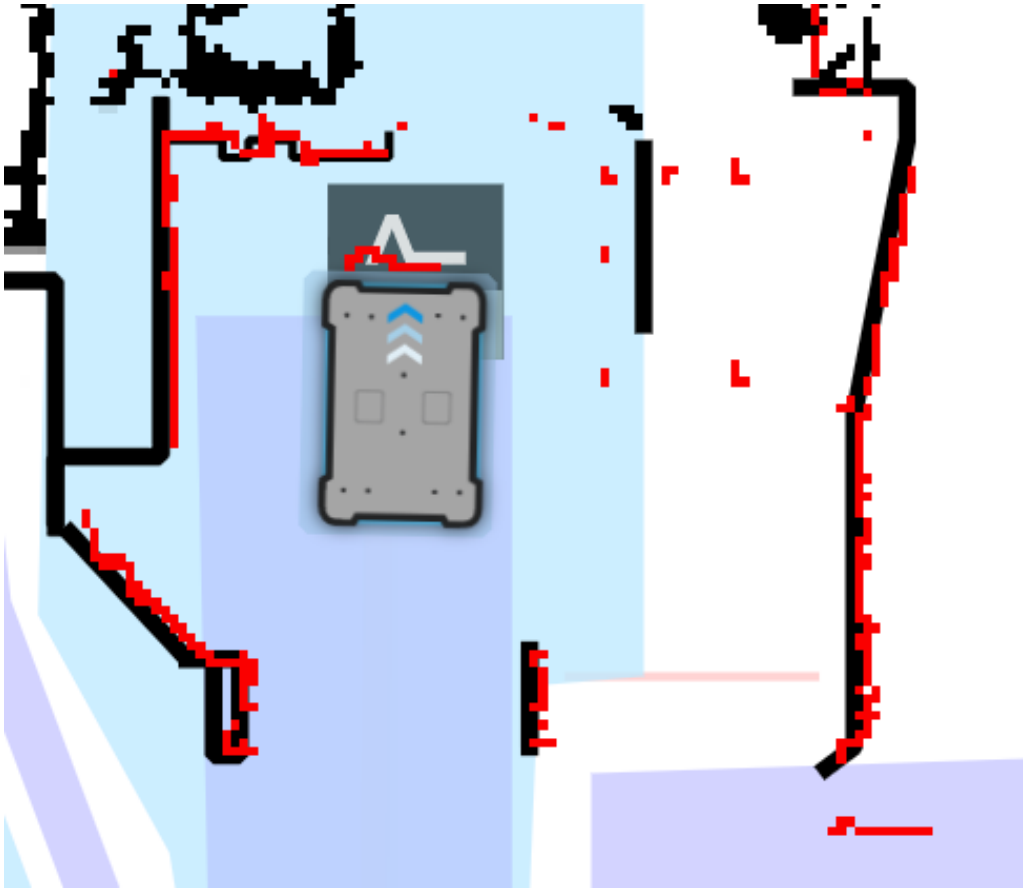
itsenäisesti annettuihin pisteisiin ja väistämään esteitä sekä löytämään vaihtoehtoisen reitin esteen kohdatessaan.

2.3 Turvallisuus

Mobiiliroboteilla turvallisuuskysymykset ovat tärkeitä, koska liikeradat ovat suurempia ja robotit toimivat ihmisten seassa (MiR, i.a.-b). AMR on yhteistyörobotti ja siten suunniteltu työskentelemään ihmisten rinnalla. Tästä syystä turvallisuus on isossa roolissa, kun robotti kulkee dynaamisissa ympäristöissä ihmisten kanssa. Robottia ympäröivät laserskannerit tunnistavat eri kohteita ja robotti arvio, voiko se ajoreittiään muuttamalla väistää vai vaatiiko tilanne pysähtymisen. Robotin ajaessa se havainnoi useita metrejä eteenpäin, jotta sillä on riittävästi aikaa arvioida, miten tulee seuraavaksi toimia. Nopeuksien pysyessä hieman kävelyvauhtia nopeampina on robotin kanssa työskentely turvallista.

2.3.1 Laserskannerit

Uusimmat turvalaserskannerit hyödyntävät ”time of flight” eli lentoaikateknologiaa (Sick, 2014, s. 3). Toimintaperiaatteena on, että skanneri lähettää pulssitettua lasersädettä ympärilleen. Mikäli lasersäde osuu kiinteään kohteeseen, skannerin vastaanotin rekisteröi heijastuksen kohteesta. Aika, joka kuluu lähetyksen ja vastaanoton impulssin välillä, on suoraan verrannollinen skannerin ja kohteen etäisyydelle. Tästä syystä lentoaikateknologia on paras luokassaan etäisyyden määritykseen.



Kuva 1. MiR-robotin laserskannerien toiminnan havainnointi

Kuvasta 1 voi havaita, kuinka laserskanneri havainnoi ympäristöään. Kuvan robotin vasemmassa yläkulmassa sekä oikeassa alakulmassa on laserskanneri, jotta saadaan kattava kuva ympäröivistä kohteista. Kuvassa 1 punaiset viivat vastaavat skannerin havaitsemia kohteita. Mustalla on merkitty esteitä ja kohteita, joita voivat olla esimerkiksi seinät. Kuten kuvasta voidaan havaita, suurin osa punaisista viivoista on mustien viivojen edessä, tällöin robotti havaitsee ennalta määrätyt seinät tai esteet. Loput punaiset viivat esittävät havaittuja esteitä, joita ei ole etukäteen ilmoitettu. Koska turvalaserskanneri lähettää laserpulssia tietyllä korkeudella viivan omaisesti, on korkeus huomioitava skanneria asentaessa.

2.3.2 LIDAR

Lidar (light detection and ranging) eli valon havainnointi ja etäisyyden mittaus on optinen etähavainnointiteknologia, joka mittaa hajautetun ja heijastetun valon ominaisuuksia löytääkseen etäisyyden tai/ja tietoa kaukaisista kohteista (Karp & Stotts, 2013, s. 151).

Yleinen tapa on määritellä etäisyys kohteeseen tai pintaan käyttämällä laserpulsseja, vaikka tutkateknologialla on mahdollista käyttää monimutkaisempia modulointeja. Lisäksi, kuten tutkateknologiassa, joka käyttää radioaaltoja valon sijaan, etäisyys määrittyy mittaamalla kaksisuuntaista viivettä lähetetyn pulssin ja havaitun heijastuneen signaalin välillä.

Laserien käyttäminen mittausprosessissa sisällyttää mitattavan kohteen jatkuvan valaisemisen laserilla (Sick, 2018., s. 4). Kun kohteet on valaistu jatkuvalla lasersäteellä, laseranturin vastaanotin hyötyy lasersäteen lähettäjän kohtisuorasta kohdistumisesta missä tahansa mittaustilanteessa. Anturi toimii itsenäisesti ilman ulkoista valonlähdettä ja sitä on mahdollista käyttää yöllä, päivällä, tunneleissa ja muissa vaihtelevissa tilanteissa aina samalla tehokkuudella, huolimatta siitä käytetäänkö sitä ulkona tai keinotekoisin valonlähtein.

3 OHJELMISTOYMPÄRISTÖ

3.1 C#

C# on moderni, kohdekeskeinen ja tyyppiturvallinen ohjelmointikieli (Microsoft, 2021). C# mahdollistaa tietoturvattujen ja vakaiden sovellusten käytön .Net-ympäristössä. C#-kielen juuret ovat C-kieliperheessä, joten C-, C++-, Java- ja JavaScript-ohjelmoijat pystyvät pienellä omaksumaan C#-kielen. C# on objekti- ja komponenttikeskeinen ohjelmointikieli, mikä lisäksi vaatii tiedon tyypityksen ja käyttäytymisen määrittelyn.

C#-kielellä sisältää yksinkertaisuudessaan vain 80 avainsanaa ja tusinan sisäänrakennettua datatyyppiä (Liberty, 2005, s. 6–7). Tästä huolimatta C#-kielellä on helppo luoda nykyaikaisia ohjelmakonsepteja. Jokaisen objektikeskeisen kielen keskiössä on tuki määrittelyyn ja käyttöön luokkien kanssa. Luokat määrittelevät uusia datatyyppejä ja mahdollistavat kielen laajennuksen sopivan paremmin kehitettävään malliin.

3.2 WPF

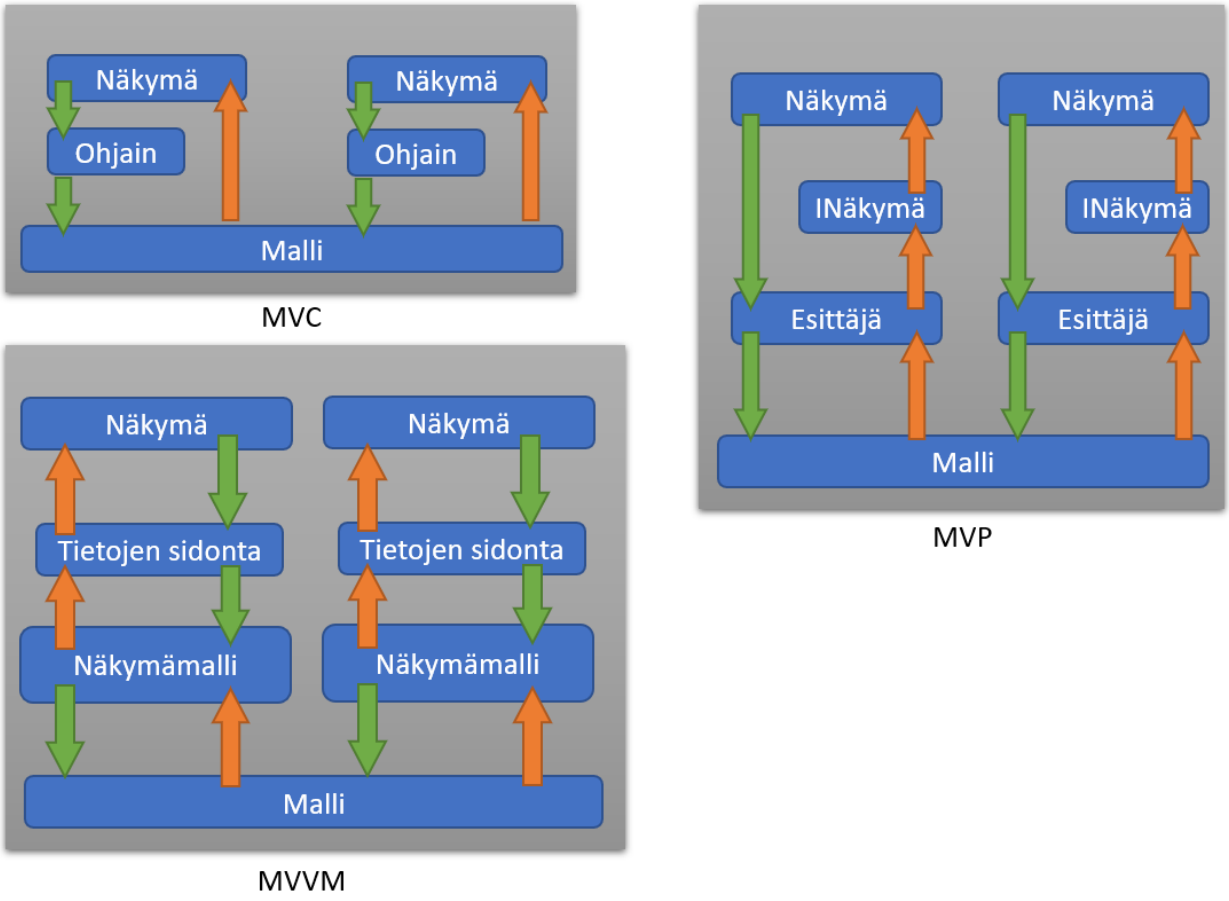
Windows Presentation Foundation (WPF) on kirjasto, jolla voidaan luoda käyttöliittymiä (Evjen ym., 2012, s. 1050). WPF-kirjaston yksi suuri etu on, että työ voidaan jakaa helposti graafisen suunnittelun ja ohjelmiston kehityksen välillä. Tämän hyödyntäminen tosin vaatii tuntemusta WPF:n graafisen puolen eXtensible Application Markup Language (XAML)-kielestä.

Vaikka WPF-kirjasto on pääasiassa tunnettu työpöytäsovelluksistaan, voidaan sillä myös luoda verkkopohjaisia sovelluksia (Chowdhury, 2018, s.10). Tästä syystä WPF-sovelluksen kaksi perustyyppiä ovat työpöytäsuoritettavat (EXE) ja verkkopohjaiset (XBAP). Työpöytäapplikaatiot ovat normaaleita .exe-suoritettavia tiedostoja, joita pystyy suorittamaan millä tahansa Windows-pohjaisella laitteella. Verkkopohjainen .xbap taas vaatii web-palvelimen ja minkä tahansa tuetun selaimen toimiakseen. Yhteinen vaatimus molemmille sovelluksille on .NET Framework, jotta niitä voi käyttää.

Vaikka ei ole pakollista käyttää XAML-kieltä käyttöliittymän luontiin, se on laajalti hyväksytty viisaaksi vaihtoehdoksi käyttöliittymäelementtien luomiseen, sen helppouden takia (Chowdhury, 2018, s.11). Käyttöliittymä voidaan luoda kirjoittamalla C#- tai VB.NET-kielillä myös, mutta tämä tekee siitä tarpeettoman vaikeaa ja hankalaa ylläpitää. Lisäksi samanaikainen kehitys vaikeutuu.

3.2.1 MVVM

Model View ViewModel (MVVM) on toimintamalli, joka paikkaa Model View Controller (MVC)- ja Model View Presenter (MVP)-mallien puutteita ja yhdistää niiden vahvuudet (Vice & Siddiqi, 2012, s. 98–99). Alla olevassa kuviossa 3 havainnoidaan niiden eroavaisuuksia.



Kuvio 3. MVC-, MVP- ja MVVM-sovellustyyppimallit (Vice & Siddiqi, 2012, s. 40, 60, 80).

Yllä olevasta kuviosta 3 voi havaita, kuinka tapa rakentaa käyttöliittymiä on muuttunut aikojen saatossa. 1979-luvulla vakiouitui MVC-malli, jonka esitysmalli muodostui

pahamaineisen joustavaksi ja tämä joustavuus osoittautui vaikeaksi oppia (Vice & Siddiqi, 2012, s. 63–65). Sovelluksessa näkymä (View) on vastuussa datan esittämisestä ja käyttäjän toimintojen keräämisestä. Näkymä saa datansa mallilta (Model). Ohjaimen (Controller) rooli on vastaanottaa käyttäjän komennot ja kommunikoida ne mallille. Itse malli toimii muistina ja mallin tehtävä on myös ilmoittaa näkymälle, mikäli datassa tapahtuu muutoksia.

Kuten kuvio 3 antaa ymmärtää, MVP-malli eroaa aiemmasta MVC-mallista hieman. MVP-malli esiintyy ensimmäisen kerran vuonna 1990 (Vice & Siddiqi, 2012, s. 82–83). MVP-mallin myötä näkymän ei enää tarvitse tarkkailla muutoksia mallissa. Esittäjä (Presenter) ottaa MVP-mallissa vastuun tiedon siirtämisestä näkymään (View) ja tiedon vastaanottamisen näkymästä ja sen siirtämisen malliin (Model). Esittäjä kommunikoi näkymän kanssa rajapinnan (IView) kautta, mikä parantaa testattavuutta, koska malli voidaan korvata jäljennöksellä.

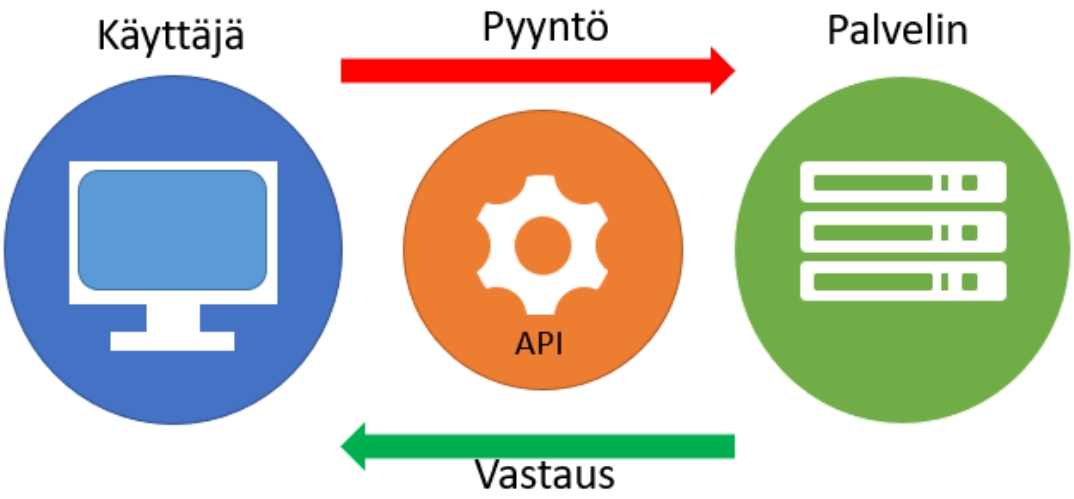
Rakenne MVVM-mallissa on samanlainen kuin MVP-mallissa, paitsi että välissä ei ole enää rajapintaa (IView) kommunikoimassa esittäjästä (Presenter) näkymään (View) (Vice & Siddiqi, 2012, s. 101). Kaikki kommunikaatio, joka tapahtuu näkymän ja näkymämallin (View Model) välillä, hoituu WPF:n sidontajärjestelmällä (Binding).

MVVM-mallissa sovelluksen malli (Model) ottaa saman roolin kuin MVP-mallissa (Vice & Siddiqi, 2012, s. 102). Näkymä on samassa roolissa kuin aikaisemmissakin malleissa, eli esittää dataa ja vastaanottaa käyttäjän toimintoja. Nyt näkymä kuitenkin välittää tiedot näkymämallille. Näkymämalli tai esitysmalli on vastuussa näkymän tilasta ja logiikasta ja on riippuvainen sidontajärjestelmästä kommunikoidessaan näkymän kanssa. Näkymämalli on välikäsi MVVM-mallissa ja huolehtii datan siirrosta mallista näkymään, sekä huolehtii käyttäjien eleet näkymästä mallille.

3.3 REST

Representational State Transfer (REST) on arkkitehtuurityyli, jota käytetään, kun halutaan luoda skaalattavia verkkopalveluita (Kanjilal, 2013, s. 24–25). REST-arkkitehtuurillinen tyyli on tullut hyvin suosituksi ympäri maailmaa verkossa kommunikoivien sovellusten

suunnittelussa. REST on käytännössä käyttäjä-palvelin-arkkitehtuuri, joka käyttää tilatonta HTTP-protokollaa. Huomioitavaa on, että REST ei ole teknologia eikä standardi. Sen sijaan se on joukko rajoitteita, joita voidaan käyttää uuden arkkitehtuurisen tyylin määrittelyyn. REST arkkitehtuurisessa tyylissä voidaan kommunikoida eri järjestelmien kanssa käyttäen HTTP-protokollaa. World Wide Web (WWW) voidaan nähdä REST-pohjaisena arkkitehtuurina. REST on tyyli, joka jakaa applikaatioiden tilan ja toiminnallisuuden resursseiksi. Nämä resurssit taas käännetään ositettaviksi osoitteiksi käyttämällä URI-osoitetta http:n yli. Näillä resursseilla on tavallinen rajapinta ja ne omistavat uniikin osoitteen. REST-pohjainen malli on tilaton, käyttäjä-palvelinpohjainen ja välimuistiin tallentava.



Kuvio 4. REST-kommunikaatio yksinkertaisesti (Kanjilal, 2013, s. 65).

Kuvio 4 havainnoi kommunikaatiota, joka tapahtuu käyttäjän ja palvelimen välillä. Resurssit vastaavat tilaa ja toiminnallisuutta (Kanjilal, 2013. s. 25). Resursseihin päästään käsiksi loogisesti rakentuvien URI-osoitteiden kautta. Tyypillisessä REST-pohjaisessa mallissa, käyttäjä ja palvelin keskustelevat pyynnöin ja vastusten avulla, kuten kuviossa 4 on esitetty. Käyttäjä lähettää pyynnön palvelimelle resurssista ja palvelin puolestaan vastaa käyttäjälle.

REST-operaatiot käyttävät HTTP-tilakoodeja, koska tilattomuutensa takia käyttäjä ei saisi muuten tietoa siitä, onko kysely onnistunut vai ei (Yellavula & Joshi, 2017. s.12). Paluukoodit hieman vaihtelevat toiminon mukaan, mutta 2xx-alkuiset koodit kertovat onnistumisesta ja 4xx-tyyppiset epäonnistumisesta. Kaikki määritellyt REST-palvelut seuraavat

muotoa, jossa on ensin verkkoaseman osoite, jota seuraa ohjelmistorajapinnan päätepiste, jonka jälkeen voidaan vielä antaa tarkentava parametri.

4 ALKUTILANNE

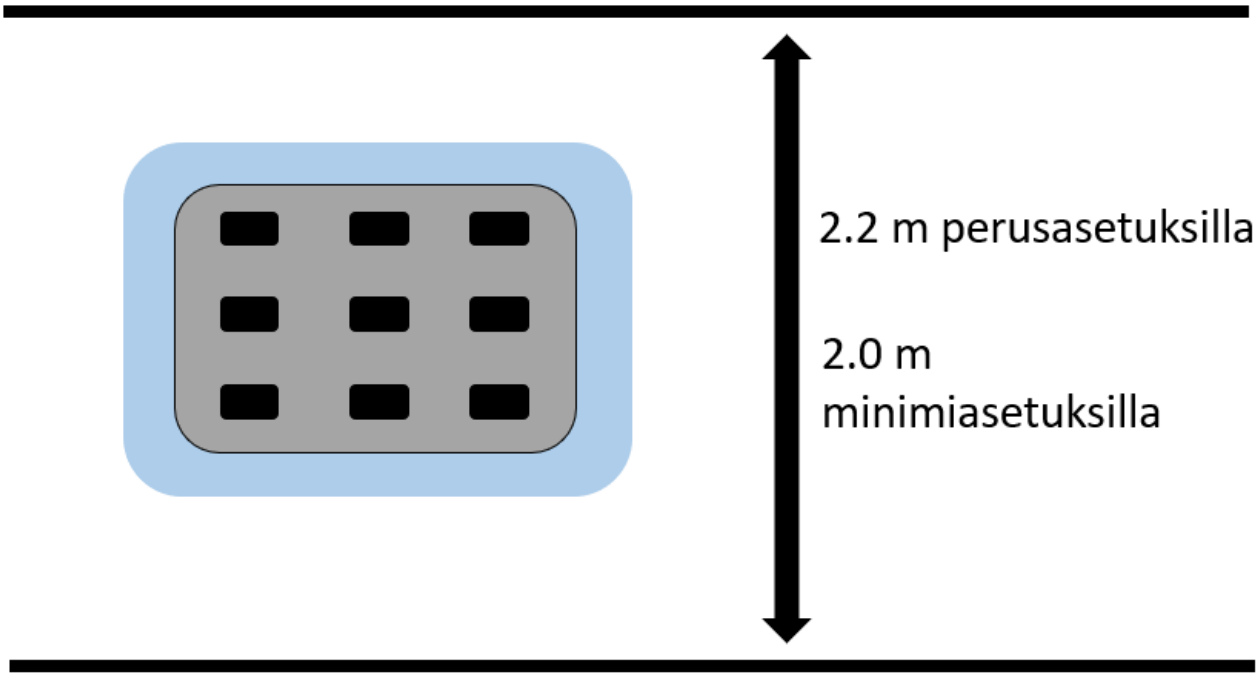
4.1 Tavoitteet

Tavoitteena oli saavuttaa toimiva case-tyyppinen ratkaisu, jossa mobiilirobotti palvelee robotisoitua työstökonesolua. Ideana ratkaisussa oli, että eri työvaihepisteet voivat sijaintinsa puolesta olla missä tahansa, koska mobiilirobotti mahdollistaa autonomisella toiminnalla modulaariset ratkaisut. Olennaisena tavoitteena oli myös osoittaa mobiilirobotin käyttömahdollisuudet työstökonepajassa, sekä todistaa mobiilirobotin hyöty ja kyky toimia sisälogistiikassa. Tavoitteena oli luoda automaattinen järjestelmä, jossa päivän aikana robottisolu ladataan täyteen ja yön aikana mobiilirobotti siirtäisi koneistettua materiaalia jäysteenpoistoon ja valmista materiaalia materiaalivarastoon.

4.2 Toteutuksen tilat

Tämä esimerkkitoteutus oli tarkoitus tehdä työstökonepajaan. Toteutuskohteeseen perehtyessä tuli selvittää robotin toimitilat eli paikat, joihin robotin tulisi päästä omatoimisesti kulkemaan. Kohteessa oli vanha puoli ja uudempi laajennettu osuus, näissä kummassakin robotin tulisi pystyä kulkemaan. Vanhemmalla puolella käytävät olivat hieman kapeammat kuin laajennetulla puolella. Tämän lisäksi paikka paikoin lattiassa oli havaittavissa kulumaa ja halkeamia, jotka tulisi ottaa huomioon käyttöönoton yhteydessä.

Mobile industrial Robotsin (29.4.2021. s. 11–14) mukaan tilavaatimukset, käytävällä sujuvaa liikkumista varten, ovat vähintään 2 metriä tilaa ja 2,2 metriä, kun käytössä on normaalikokoinen robotin tilamääritys (footprint). Oviaukkojen, joista robotin on tarkoitus ajaa, tulisi olla 1,95 metriä leveitä sujuvuuden säilyttämiseksi.



Kuvio 5. MiR-robotin tilavaatimukset suoraan ajaessa.

Kuten kuviosta 5 voidaan havaita, käytävän tulisi olla ainakin 2 metriä leveä, jotta kulkeminen on sujuvaa. Kuviossa 5 on lisäksi vaalean sinisellä merkitty niin sanottu robotin jalanjälki, eli alue minkä mukaan robotti pyrkii väistämään esteitä. Robotin jalanjälki on alue, jonka robotti mieltää sen fyysiseksi kooksi. Tästä syystä sen tulee olla määritelty isommaksi kuin itse robotti, jotta vältytään yhteentörmäyksiltä.

Tilasuositusten takia kohteessa tuli käydä suorittamassa mittauksia, jotta saattoi saada kuvan siitä, kuinka robotti mahdollisesti pystyisi kulkemaan käytäviä pitkin.

4.2.1 Staattisuus ja dynaamisuus

Mobile industrial Robotsin (27.7.2021, s. 81–83) mukaan lokalisaatioprosessin tarkoitus on antaa mahdollisuus robotille tietää, missä se sijaitsee sen omalla kartalla. Mobiilirobotilla on kolmen tyypistä dataa, joiden perusteella se määrittelee todennäköisimmän paikan itselleen kartallaan. Robotti käyttää liikkeen aloituspistettä, robotin pyörien enkooderien dataa sekä laserskannereiden tarjoamaa dataa paikkamäärittelyyn. Varmistuakseen, että lokalisaatio on mahdollisimman luotettava, tulee robotin toimintaympäristössä olla riittävästi staattisia maamerkkejä. Maamerkkeinä voivat toimia seinät, oviaukot, tolpat tai mikä

tahansa muu kohde, joka on pysyvästi sillä paikalla, kuin kartan määrityshetkellä. Robotin tulee pystyä havaitsemaan staattiset maamerkit, jotta se voi arvioida senhetkisen paikkansa. Tästä syystä tulee varmistaa, ettei toimintaympäristössä ole liikaa dynaamisia esineitä staattisten maamerkkien edessä.

Tiloihin perehdyttäessä oli huomioitava, että koska materiaalin kuljetus hoituu trukkien, Roclan käsikäyttöisten pinoamisvaunuin ja tavallisilla pumppukärryin, materiaalia oli käytävien reunoissa ja materiaali liikkui vauhdikkaasti. Tämä tarkoittaa käytännössä sitä, että ympäristö oli hyvin dynaaminen ja tämä saattaisi aiheuttamaan hankaluuksia mobiilirobotin navigoinnissa ja sen itsensä paikoittamisessa.

4.3 Verkkoarkkitehtuuri

Mobile industrial Robotsin (28.10.2021, s. 6) verkko- ja wifi-dokumentaation mukaan normaalia toimintaa varten robotti tai robotit vaativat sisäverkon, johon liittyä. Järjestelmästä riippuen vaatimukset voivat vaihdella, mutta saman verkon ollessa MiR-robottien ja toisten laitteiden käytössä, on suotavaa täyttää kriteerit, jotka esitetään alla seuraavassa taulukossa.

Taulukko 1. MiR-robotin ohjevaatimukset verkon osalta. (MiR, 28.10.2021, s. 6)

Parametri	Kuvaus	Vaatimus
Signaalin vahvuus	Signaalin vahvuus robotin näkökulmasta, kun on yhdistetty parhaaseen liityntäpisteeseen	Minimissään -67 dBm (desibelimilliwatti)
Toissijaisen signaalin vahvuus	Signaalin vahvuus robotin näkökulmasta, kun on yhdistetty toiseksi parhaaseen liityntäpisteeseen	Minimissään – 75 dBm (desibelimilliwatti)
Signaalin suhde signaalimeluun	Signaalin suhde meluun robotin näkökulmasta	Minimissään 20 dBm (desibelimilliwatti)
Tiedonsiirtonopeus	Tiedonsiirtonopeus jokaiselle robotille ja robotilta.	Minimissään 20 megabittiä sekunnissa
Kanavahäiriötä	Liityntäpisteiden lukumäärä kanavaa kohden, minkä voimakkuus on vähemmän kuin -85 dBm	Maksimissaan 2 liityntäpistettä/kanavaa -85 dBm
Viive	Aika, joka kuluu lähettää ja vastaanottaa viesti robotilta	Maksimissaan 200 millisekuntia
Kaista	Tietomäärä, joka voidaan lähettää ajan mittaan	Minimissään 1 megabittiä sekunnissa robottia kohden
Pakettihävikki	Prosentuaalinen osuus kommunikaatiopaketeista, jotka voivat kadota	Maksimissaan 2 %

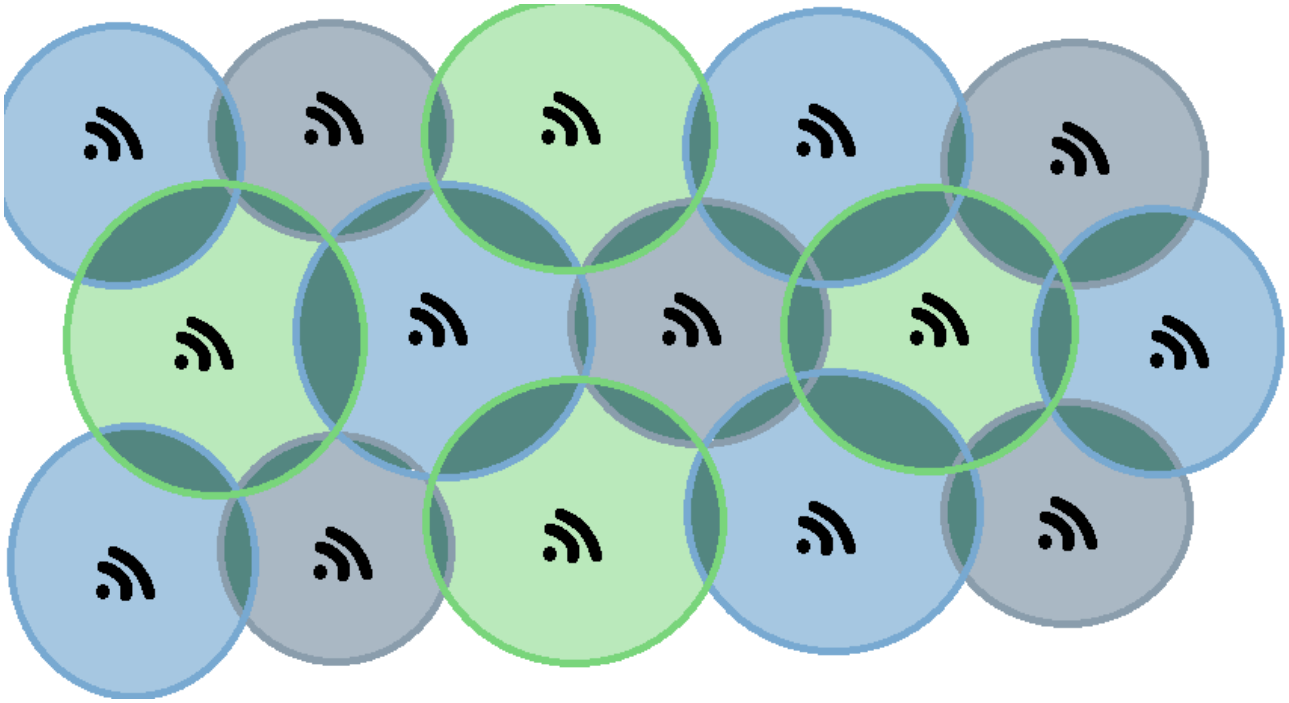
Kuten taulukosta 1 voidaan huomata, verkon vaatimukset eivät ole mitenkään mahdottomat, mutta olemassa olevan verkon kattavuus ja käyttöaste on hyvä selvittää ennen robotin käyttöönottoa. Tästä syystä kohdeympäristössä oli hyvä käydä mittaamassa verkon kuuluvuutta, kattavuutta ja kanavien käyttöasteita.

MiR-yrityksen (28.10.2021. s. 7–8) verkko- ja wifi-dokumentaatio tuo myös esiin, että käytettävän wifi-verkon tulee kokonaan kattaa robotin suunnitellut liikunta-alueet. Lisäksi, mikäli verkon liityntäpisteen käyttöaste on enemmän kuin 60 %, robottiin tai robotteihin alkaa vaikuttamaan viive ja pakettihävikki.

4.3.1 Puutteiden korjaus

Kohdeympäristöön perehtymisen jälkeen huomattiin puutteita verkossa, mikä johti siihen, että liityntäpisteitä oli lisättävä. Tämän lisäksi robottia varten luotiin oma verkko. Näillä toimenpiteillä varmistettiin, että robotin on mahdollista kulkea siellä mihin sen on tarkoituskin.

MiR-yrityksen (28.10.2021, s. 13) verkko- ja wifi-dokumentaatiossa kerrotaan, kuinka esim. radiotaajuudet voivat häiritä wifi-signaalia. Signaaliin vaikuttavat eniten toiset wifi-signaalit. Häiriön minimoinniksi on tärkeää järjestää ja määrittää liityntäpisteet siten, että viereiset liityntäpisteet käyttävät eri kanavia. Tämän lisäksi tulee varmistaa, että verkkojen kanssa ei tule päällekkäisyyksiä enempää, kuin on tarve yhteyden siirrossa liityntäpisteestä toiselle. Kuviossa 6 esitetään, kuinka verkko on jaettu eri kanaville.



Kuvio 6. Havainnointi tilanteesta, jossa on kolmea kanavaa käytetty verkon rakentamiseen (MiR, 28.10.2021, s. 16)

Kuvio 6 havainnollistaa tilannetta, jossa on rakennettu verkko, jossa jokainen liityntäpiste muodostaa ympyrän muotoisen alueen, ja jokainen liityntäpiste käyttää eri kanavaa. Tämän lisäksi liityntäpisteillä on vain vähän päällekkäisyyttä naapuripisteidensä kanssa. Kuvion mukaisesti myös kohdeympäristössä meneteltiin näin liityntäpisteiden kanavien kanssa.

5 Mobiilirobotin käyttöönotto

5.1 Mobiilirobotti

Työssä käytetty mobiilirobotti oli tanskalaisen Mobile industrial Robotsin (MiR) valmistama MiR1000-mobiilirobotti. MiR-yrityksen lisäksi erilaisia mobiilirobotteja valmistaa esimerkiksi OMRON, Standard Robots ja Robtnik. Vaihtoehdoista huolimatta tanskalainen MiR valikoitui tarkoitukseen, koska työnantaja JTA Connection Oy toimii virallisena MiR-toimittajana Suomessa, Venäjällä ja Baltiassa. Lisäksi JTA on valtuutettu MiR-integroija maailmanlaajuisesti.

Valittu MiR1000 ei ollut täysin vakiomallinen, vaan siinä oli päällysmoduulina palettinostin. Tällä kyseisellä robotilla on ilmoitettu nostokapasiteetti 1000 kiloon asti ja näin se kykenee kuljettamaan 1000 kilon kuormaa. Kuvassa 2 on kuva käytetystä MiR1000-robotista.

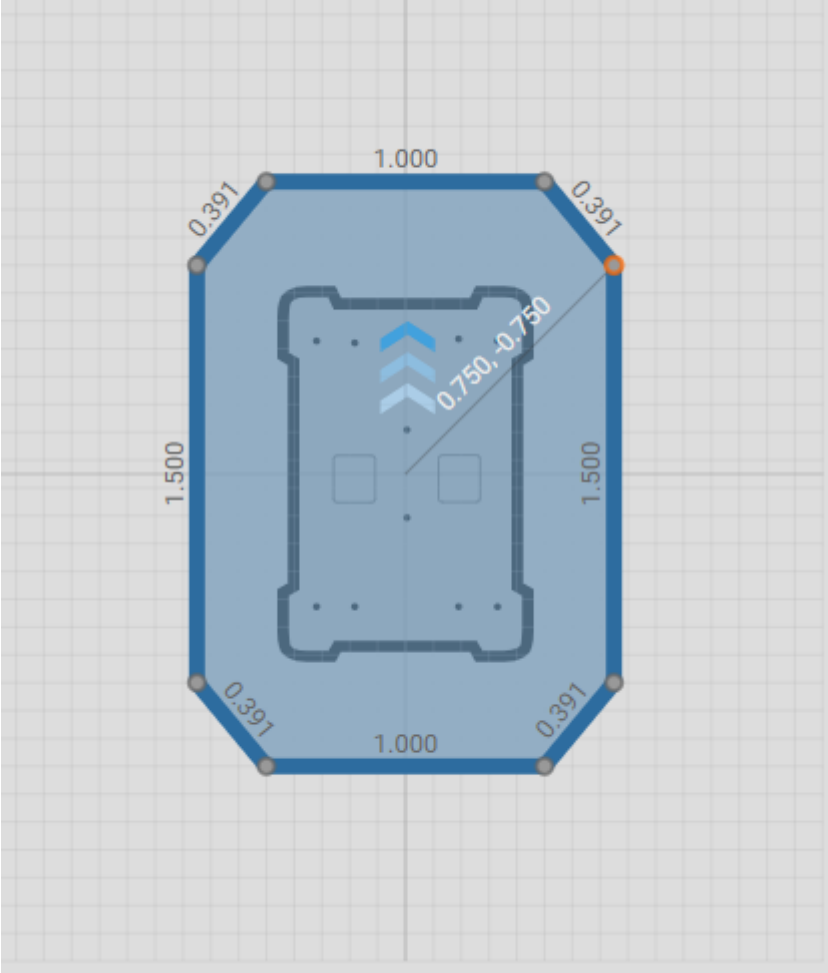
Kuva 2. MiR1000-mobiilirobotti.

Kuvan 2 robotti on täysin samanlainen kuin käytetty robotti. Robotin päällä on eurolavanostin, jonka avulla robotti kykenee siirtämään eurolavoja. Kuvassa 2 robotti on kuvattu etuvasemmalta, tämän tunnistaa robotin eteenpäin suuntaavista Lidar-skannereista.

Kuvan oikeassa reunassa näkyy myös toinen robotin laserskannereista. Robotin toinen laserskanneri on vastakkaisessa kulmassa robotin takakulmassa. Kuvassa näkyvät lisäksi puolet robottia ympäröivistä hätäseis-painikkeista. Kuvasta 2 voi myös nähdä robottia ympäröivän tilavärin, joka kulkee robotin ympäri hätäseis-painikkeiden alapuolella. Robotilla on useita eri värivaihtoehtoja, joilla on sen mahdollista indikoida erilaisia tiloja sitä ympäröiville käyttäjille. Esimerkiksi kuvassa esiintyvä sininen on merkki siitä, että robotti suorittaa tehtävää.

5.2 Ensiaskeleet

Robotin käyttöönotto aloitetaan ottamalla yhteys robotin omaan käyttöliittymään robotin sisäisen langattoman verkon kautta. Kun yhteys on saavutettu robotin käyttöliittymään, on syytä ensimmäisenä suorittaa kalibraatio robotin ajomoottorien enkoodereille, jotka vastaavat pyörien pyörintäetäisyyden mittaamisesta. Lisäksi on hyvä tarkistaa robotin footprintin eli jalanjäljen oikeellisuus. Tämän lisäksi on hyvä tarkistaa, että robotin sisäiset järjestelmät ovat toiminnassa ja ettei mikään ilmaise virhetilasta.

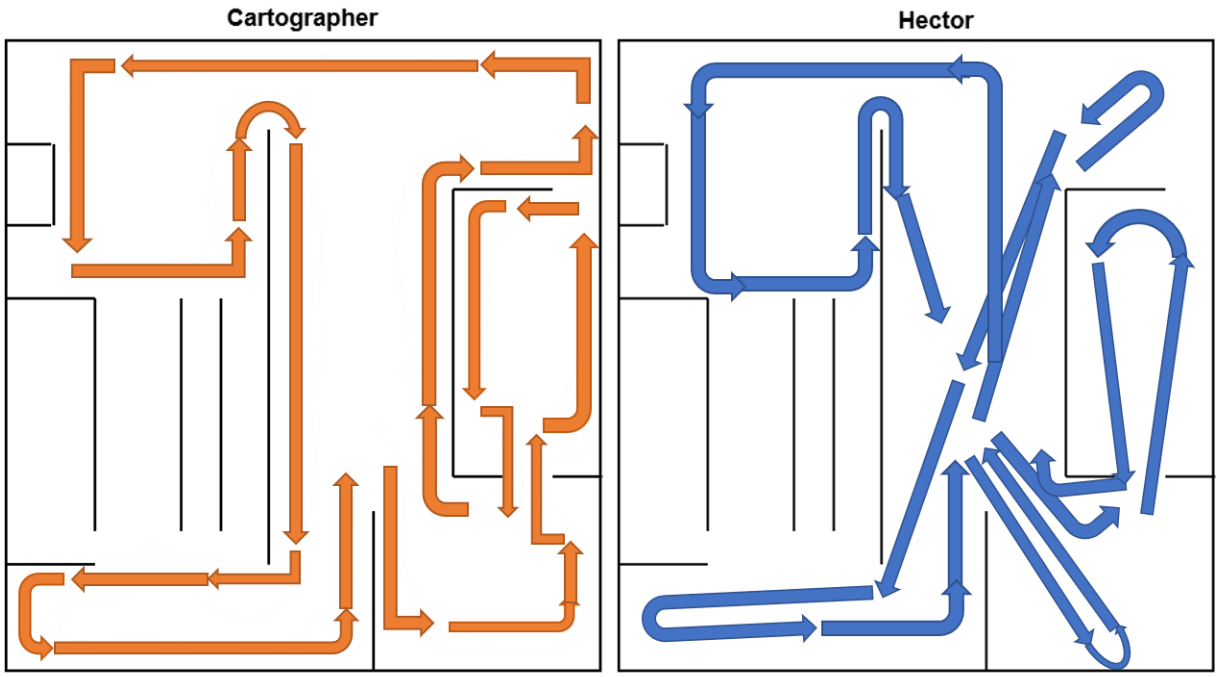


Kuva 3. MiR1000-robotin vakiojalanjälki

Kuvassa 3 esitetään robotin jalanjälki, kuinka se on robotin käyttöliittymässä näkyvissä. Jalanjälki on alue, joka kertoo robotille, minkä kokoinen se on. Tällä tavoin robotti pyrkii välttämään yhteentörmäyksiä, koska jalanjäljen törmätessä johonkin esteeseen, ei fyysinen robotti ole vielä törmännyt.

5.3 Tilan kartoittaminen

MiR Academyn (i.a.) kartoitusta käsittelevässä kurssissa tuodaan ilmi, että kartoittaminen voidaan toteuttaa kahta eri menetelmää seuraamalla. Ensimmäisenä menetelmänä on Cartographer, joka on uudempi metodi. Cartographer tallentaa useita pienempiä karttoja, joista se kokoaa yhden ison kartan kartoituksen lopuksi. Hector on toinen menetelmä, joka on perinteisempi tapa. Siinä data tallennetaan yhteen karttaan jo kartoitusvaiheessa. Etuna Cartographer-mallissa on, että kartoista tulee tarkempia. Kun luodaan pienempiä karttoja, voi se auttaa virheiden korjaamisessa. Kuviossa 7 esitetään kummatkin tavat.



Kuvio 7. Kartoitusmetodien eroavaisuudet

Kuten kuviosta 7 käy ilmi, Cartographer-mallissa pyritään kartoittamaan käytössä oleva tila ympyrän suuntaisesti. Tärkeää olisi lopettaa kartoitus samaan pisteeseen, mistä se on aloitettu. Hector-malli kuvaa haarautuvaa kartoitusta, jossa tila kartoitetaan osa kerrallaan palaten aina lähtöpisteeseen ennen seuraavan osan kartoitusta.

Ennen kartoittamisen aloittamista täytyy varmistaa, että kartoitettavalla alueella on mahdollisimman vähän dynaamisia esteitä. Tärkeää on saada karttaan tallennettua sellaiset kohteet, jotka ovat aina samassa paikassa eivätkä liiku muualle.

Kartoittaessa toteutuskohteen tiloja käytettiin Cartographer-mallia kartoitusmallina. Tämä mahdollisti tarkan kartan kohteesta. Kartoituksen päätteeksi robotille pitää vielä määritellä kartan orientaatio. Kun kartta on luotu ja kohdistettu oikeaan asentoon, tulee karttaa hieman ”siivota”. Siivoamisella tarkoitetaan dynaamisten esineiden poistamista kartasta, tämä edistää robotin kykyä paikoittaa itsensä tarkasti kartalla.

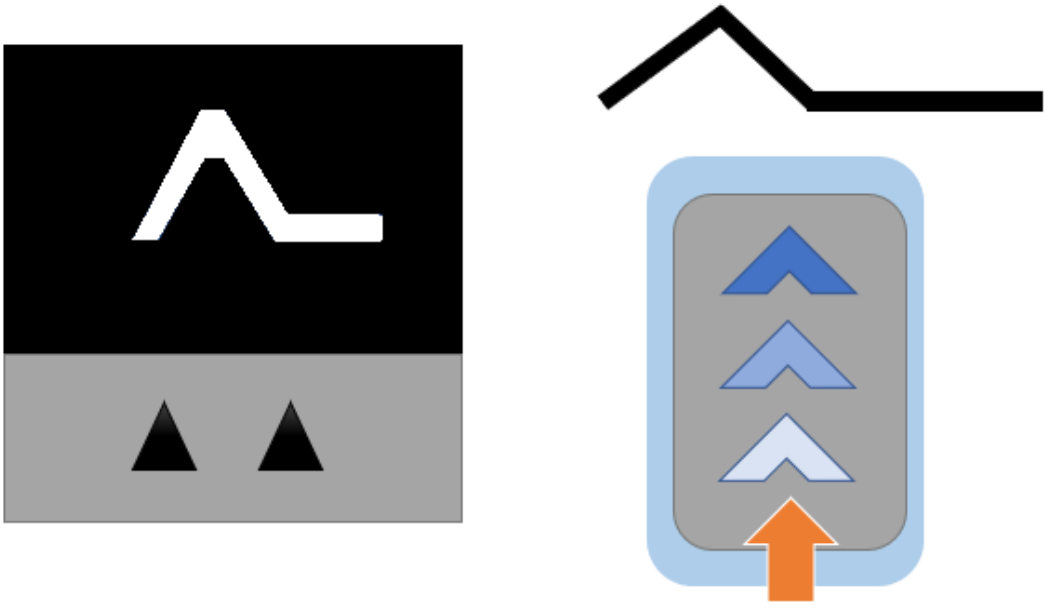
5.4 Valmiin kartan viimeistely

Kartoituksen valmistumisen jälkeen on mahdollista määritellä robotin liikepisteet, paikoituspisteet ja latauspiste. Pisteiden määrittelyiden lisäksi kartoituksen jälkeen on mahdollista

muokata karttaa erilaisilla alueilla. Alueiden avulla on mahdollista esimerkiksi rajoittaa robotin kykyä kulkea tietylle alueelle tai ohjeistaa suosimaan tiettyä reittiä toisen sijasta.

Liikepisteitä voidaan määritellä karttaan helposti robotin oman käyttöliittymän kautta. Liikepiste voi olla osana tehtävää, esimerkiksi pitkällä siirtymällä välipisteenä tai päämääränä, mihin robotti siirtyy. Liikepisteitä ei ole sidottu mihinkään fyysiseen pisteeseen ja tästä syystä ne voivat olla hieman epätarkkoja. Mikäli robotin paikkatieto on paikkansapitävä, pitäisi robotin silti pystyä löytämään paikkapiste noin kymmenen sentin tarkkuudella.

Paikoituspisteet, toisin kuin liikepisteet, on sidottu fyysisiin kohteisiin, jotka paikoittuvat karttaan fyysisten piirteiden avulla (MiR, 27.7.2021, s. 120–121). Robottien paikotuspisteitä on hyvä käyttää, kun tarvitaan tarkkuutta, esimerkiksi kun robotti asemoidaan noutopisteeseen tai jättöpisteeseen. Robotin latauspisteessä on oma paikoituspiste, jotta robotti onnistuu pääsemään lataukseen ilman ihmisen avustusta. Laturiin pitää asemoitua tarkasti latauksen onnistumisen varmistamiseksi. Robotti käyttää laserskannereitaan löytääkseen paikoituspisteensä fyysisiä piirteitä ja pystyy paikoittamaan itsensä, joka kerta samaan paikkaan tarkasti.

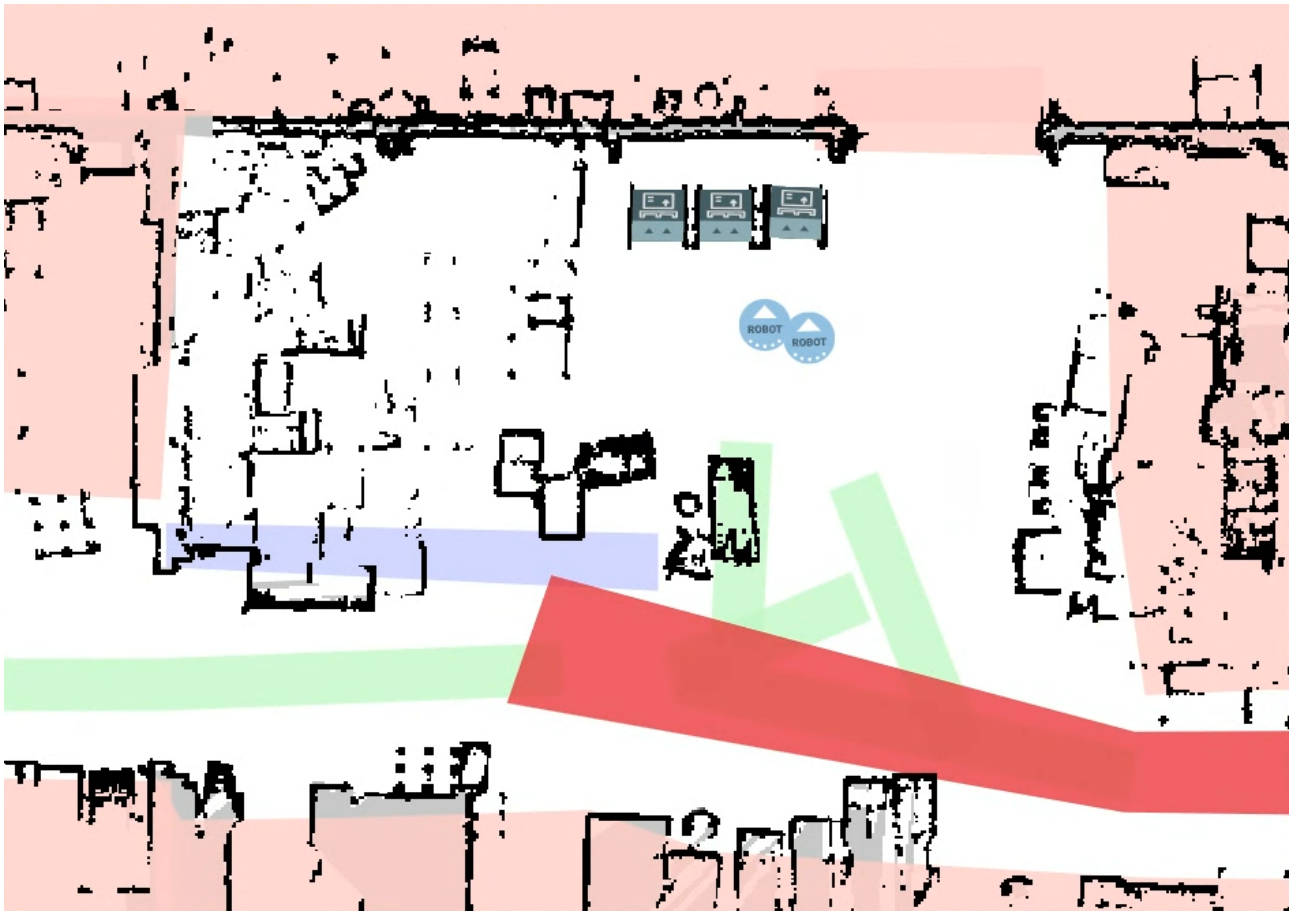


Kuvio 8. VL-markkeri ja robotin paikoittaminen. (MIR, 27.7.2021, s. 122.)

Kuviossa 8 vasemmalla on kuvaus mobiilirobotin paikoituspisteestä nimeltä VL-markkeri. Robotti käyttää VL-markkeria itsensä paikoittamiseen tarkasti. Symbolista käy ilmi paikan muoto ja oletusajosuunta. Kuviossa 8 oikealla on myös esitetty, miten robotti paikoittaa itsensä merkin eteen, lisäksi kuviosta käy ilmi myös merkin fyysinen muoto. Merkin nimi VL on myös tunnistettavissa merkin fyysisestä muodosta. Oikeasta markkerista löytyy V-kirjainta muistuttava kuoppa, jonka vieressä on tasainen pinta. VL-merkin lisäksi on V-, L- ja palkkimerkkejä, mihin robotti on mahdollista paikoittaa. V-merkistä ja L-merkistä voidaan päätellä miltä ne fyysisesti näyttävät, mutta palkkimerkki on fyysisiltä ominaisuuksiltaan kaksi yhdensuuntaista palkkia, joiden väliin robotti paikoittaa itsensä.

Viimeisenä karttaan liittyvänä maininnan arvoisena asiana on erilaisten alueiden määrittely. Alueilla voidaan muuttaa sitä, miten robotti käyttäytyy jollakin osalla karttaa. Aluemäärittelyn tekeminen on hyödyllistä, kun esimerkiksi on ahdas oviaukko, josta robotin tulee kulkea, tai halutaan määritellä robotille haluttu kulkureitti, jota robotti pyrkii seuraamaan mahdollisuuksien mukaan. Hyödyllinen alue, joka kannattaa merkitä jokaiseen karttaan on

forbidden zone eli kielletty alue. Robotti ei aja tai suunnittele reittejään kielletylle alueelle. Tämä mahdollistaa robotin kulkureitin rajaamisen tai estää robotin ajamisen paikkoihin, joista se ei pääse kohteeseensa. Olennainen ominaisuus kielletyllä alueella on myös se, että robotti ei yritä edes suunnitella kielletylle alueelle reittiä. Tästä syystä varsinkin suuremmilla kartoilla tämä lyhentää robotin reitinsuunnitteluaikaa.



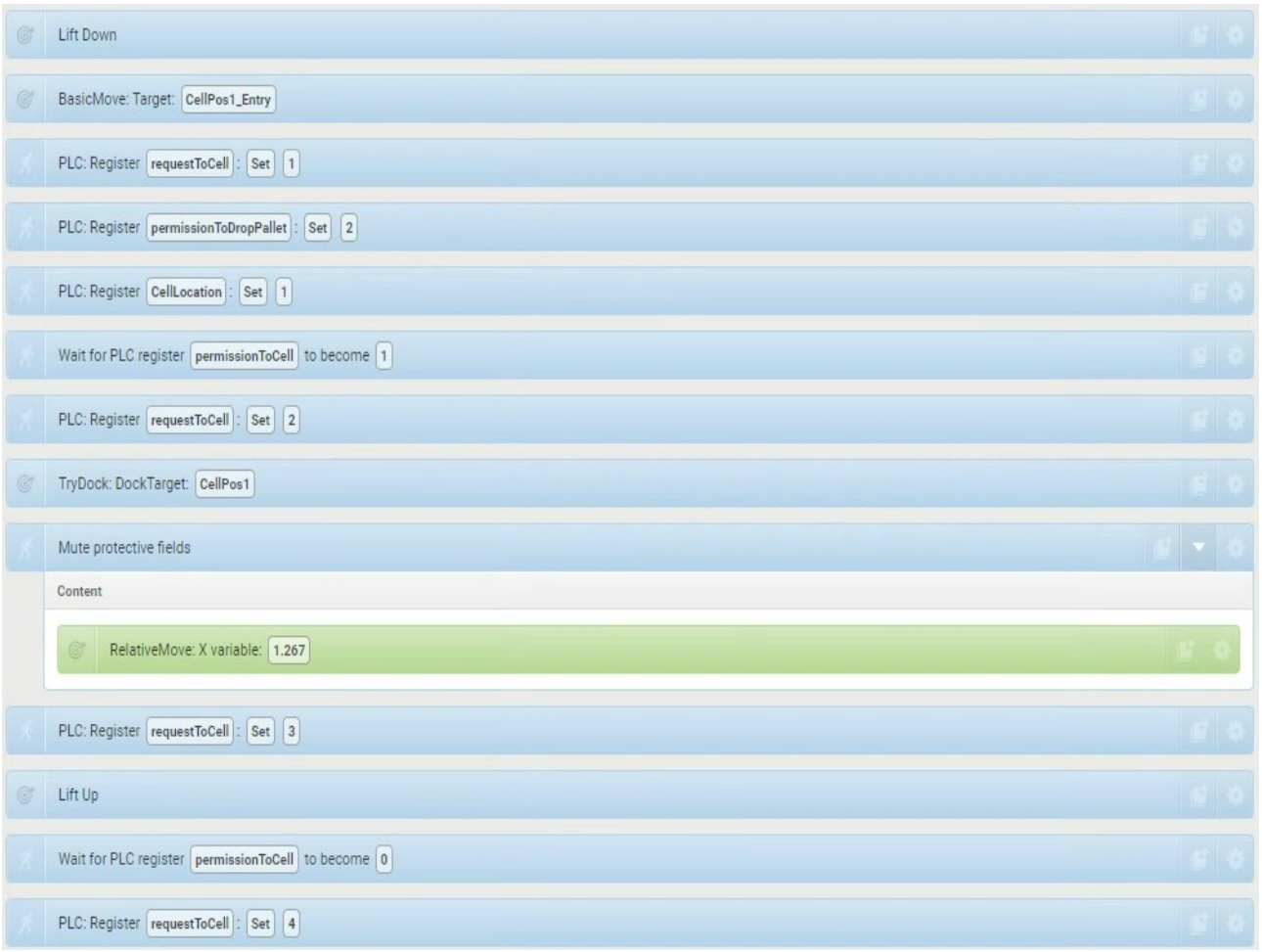
Kuva 4. Karttanäkymä toteutuskohteesta

Kuvassa 4 on näkymä toteutuskohteen yhdestä robotin toimipaikasta. Kuvassa vaaleanpunainen alue kuvaa kiellettyä aluetta, johon robotti ei voi ajaa. Punaisella värillä kuvassa on merkitty nopeusalue, jolle voidaan määrittää tavallisesta poikkeava nopeus käytettäväksi tällä alueella. Vihreät alueet kuvaavat alueita, jotka on suositeltuja kulkureittejä robotille. Robotti pyrkii suunnittelemaan reittinsä vihreitä alueita pitkin, mutta mikäli tämä estynyt, käyttää se toista reittiä. Kuvassa siniset pisteet ovat liikepisteitä, joihin robotti tulee, kun se siirtyy toiselta alueelta tälle alueelle. Sinisten pisteiden yläpuolella on kolme paikoituspistettä tavaran toimitusta ja noutoa varten. Mustat viivat ja pisteet ovat seiniä, koneita, laatikoita ja muita esteitä, jotka robotti on havainnut laserskannerillaan kartoitusvaiheessa.

5.5 Tehtävät

Robotti toimii missioiden eli tehtävien kautta, niitä luodaan käyttöönottovaiheessa (MIR, 27.7.2021, s. 125). Tehtävä koostuu toiminnoista, kuten liiketoiminnoista, logiikkatoiminnoista, paikoitustoiminnoista ja äänistä, jotka voidaan yhdistää tehtäväksi niin monella toiminnalla kuin tarvitaan. Tehtäviä voidaan upottaa toisiin tehtäviin, mikä mahdollistaa aliohjelmien luonnin ja sujuvan käytön. Suurin osa toiminnoista omaa muutettavat parametrit, esimerkiksi liiketoiminnossa voidaan valita mihin mennään. Lisäksi suurin osa toiminnoista voi käyttää muuttujia, käyttäjä voi valita parametrille uuden arvon joka kerta tehtävää käytettäessä.

MiR-mobiilirobotin tehtävien luominen ja muokkaaminen on tehty helpoksi oppia. Tehtävät rakentuvat raahattavista rivimäisistä toiminnoista, jotka on visualisesti yksinkertaisia ja helppoja tunnistaa. Tehtävien helpolla luomisella on omat heikkoutensa. Heikkouksista suurimpana on se, että tehtävien rakenne ja toiminnot on rajoitettu valmiisiin rakenteisiin. Tämä tarkoittaa sitä, että halutulle toiminnolle ei löydy valmista rakennetta sitä ei voi luoda suoraan. Tämä tarkoittaa, että tehtävät ovat rajoittuneet melko yksinkertaisiin sovelluksiin. Jos haluaisi tehdä monimutkaisempia sovelluksia, tarvitsee tehdä jokin ulkoinen sovellus robotin tehtävien rinnalle.



Kuva 5. Tehtävän sisällä olevat toiminnot

Kuvassa 5 on esimerkki tehdystä ja toimivasta tehtävästä. Tässä haetaan jotain paikasta yksi ja toimitetaan se paikkaan kaksi. Tehtävä ei näy kokonaisuudessaan kuvassa 5, mutta auttaa asian hahmottamisessa. Tehtävä aloitetaan alitehtävällä, jonka tarkoitus on varmistaa, että palettinostin on ala-asennossa. Mikäli se ei ole, ajetaan se ala-asentoon. Tämän jälkeen suoritetaan alitehtävä liittyen liikkumiseen. Liikkeen jälkeen tulee keskustella ulkoisen sovelluksen kanssa, jotta voidaan varmistua siitä, että on lupa hakea tai viedä kyseiseen paikkaan. Tämän jälkeen tehtävä jatkuu erinäisillä rekisterimuutoksilla, liikekomennoilla ja nostimen tilan vaihteluilla. Kuvasta 5 käy ilmi aikaisemmin mainitut toiminnot ja kuinka ne ovat riveittäin suoritusjärjestyksessä ylhäältä alaspäin. Lisäksi kuvassa on toiminto toisen toiminnon sisässä.

5.6 Fleet ja verkko

MiR Fleet on keskusyksikkö, joka ohjaa ja valvoo MiR-robottien toimintaa (MiR, 14.1.2022, s. 9). Käyttäjät pääsevät käsiksi Fleetin verkkopohjaiseen käyttöliittymään minkä tahansa selaimen kautta. MiR Fleetin vastuulla on järjestää tehtäviä roboteille, lisäksi se autonomisesti lähettää robotit laturiin. Fleetin yksi tehtävistä on myös valvoa, etteivät robotit törmää keskenään.

Jotta robotit kykenevät kommunikoimaan Fleetin kanssa, tulee olla yhteys samaan verkkoon (MiR, 14.1.2022, s. 16, 19–20). Robotit tarvitsevat jatkuvan ja riittävän tasaisen sekä vahvan yhteyden, jottei robottien toiminnassa muodostu ongelmia. Fleet tulee olla verkossa kiinteällä yhteydellä ja robotit langattomasti.

Kun käytössä on vain yksi robotti, kuten tämänkin toteutuksen yhteydessä, ei ole tarpeen hankkia MiR Fleetiä. Fleet kuitenkin päätettiin hankkia, koska tämä mahdollistaa tulevaisuudessa uusien robottien lisäämisen järjestelmään vaivattomasti. Yhdellä robotilla Fleet ei tuo lisäarvoa järjestelmään, ellei suunnitelmissa ole myöhemmin lisätä järjestelmään lisää robotteja. Fleet jakaa tehtävät siihen yhdistettyjen robottien kesken sen mukaan pystyykö robotti suorittamaan tehtävää ja mikä robotti on lähimpänä. Fleet myös jakaa tehtävä- ja karttamuutokset automattisesti siihen liitetyille roboteille.

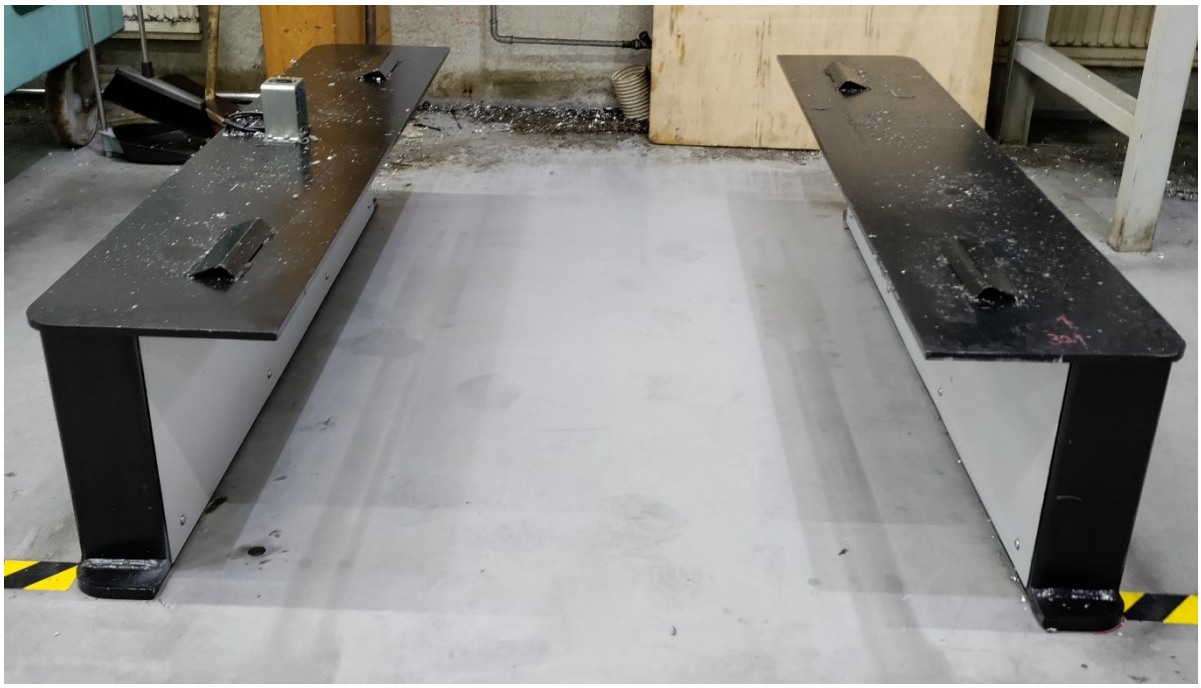
MiR Fleet asennettiin Linux-pohjaiselle palvelintietokoneelle. Asennuksen ja verkon määrittelyn jälkeen oli mahdollista päästä käsiksi Fleetin käyttöliittymään IP-osoitteen kautta. Vielä ennen robotin lisäämistä Fleetiin tuli päivittää Fleet uusimpaan ohjelmistoversioon. Ohjelmistoversiopäivitysten ollessa kunnossa oli vuorossa robotin yhdistäminen samaan verkkoon Fleetin kanssa. Kun robotti on samassa verkossa Fleetin kanssa voidaan se liittää osaksi Fleetiä. Liittämisen jälkeen robotille on mahdollista antaa tehtäviä Fleetin toimesta.

Robotin ollessa yhteydessä Fleetiin ongelmat verkon kattavuudessa tulevat ilmeisiksi hyvin nopeasti, mikäli niitä löytyy. Verkossa havaittiin puutteita, joita sittemmin selvitettiin vian paikantamisella ja ratkaisujen läpikäynnillä. Ratkaisuksi löytyi verkkokanavien määrittely, mikä auttoi yhteysongelmaan poistamalla ylimääräistä odottelua.

6 TYÖN TOTEUTUS

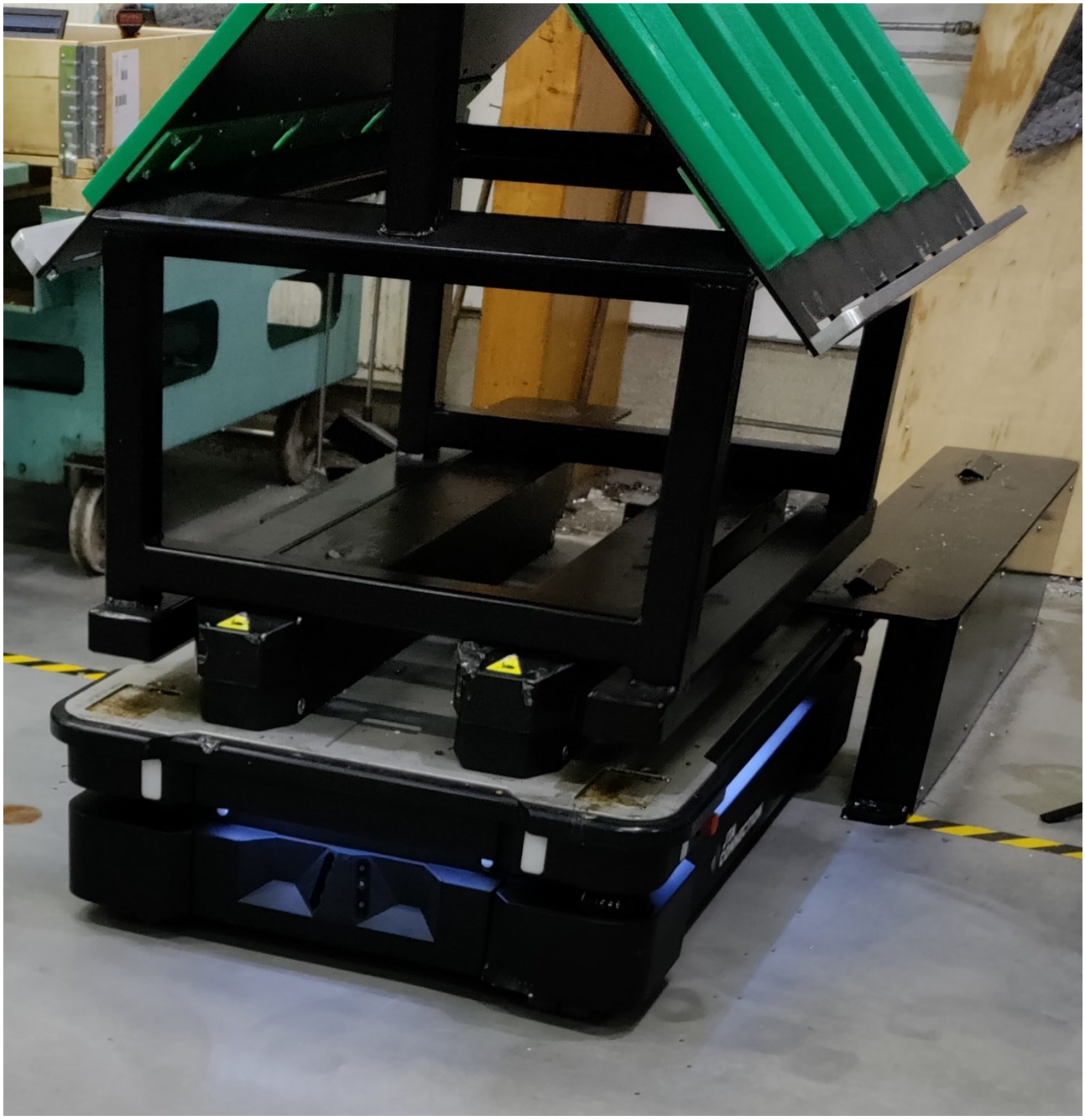
6.1 Mobiilirobotin toiminta

Toteutuskohteen ollessa työstökonepaja, kohteessa oli olemassa tiedostettuja haasteita toiminnan kanssa. Työn toteutuskohteessa vallitsee korkea dynaamisuus, eli ympäristömuutosten vaikutus robotin toimintaan ja varmuuteen. Tämän lisäksi paikka paikoin kapeat käytävät haittaavat toimintavarmuutta ja turvallisuutta. Lisäksi mobiilirobottia varten tuli asentaa niin sanotut palettipaikat, jotta robotti pystyy toimittamaan tavaraa.



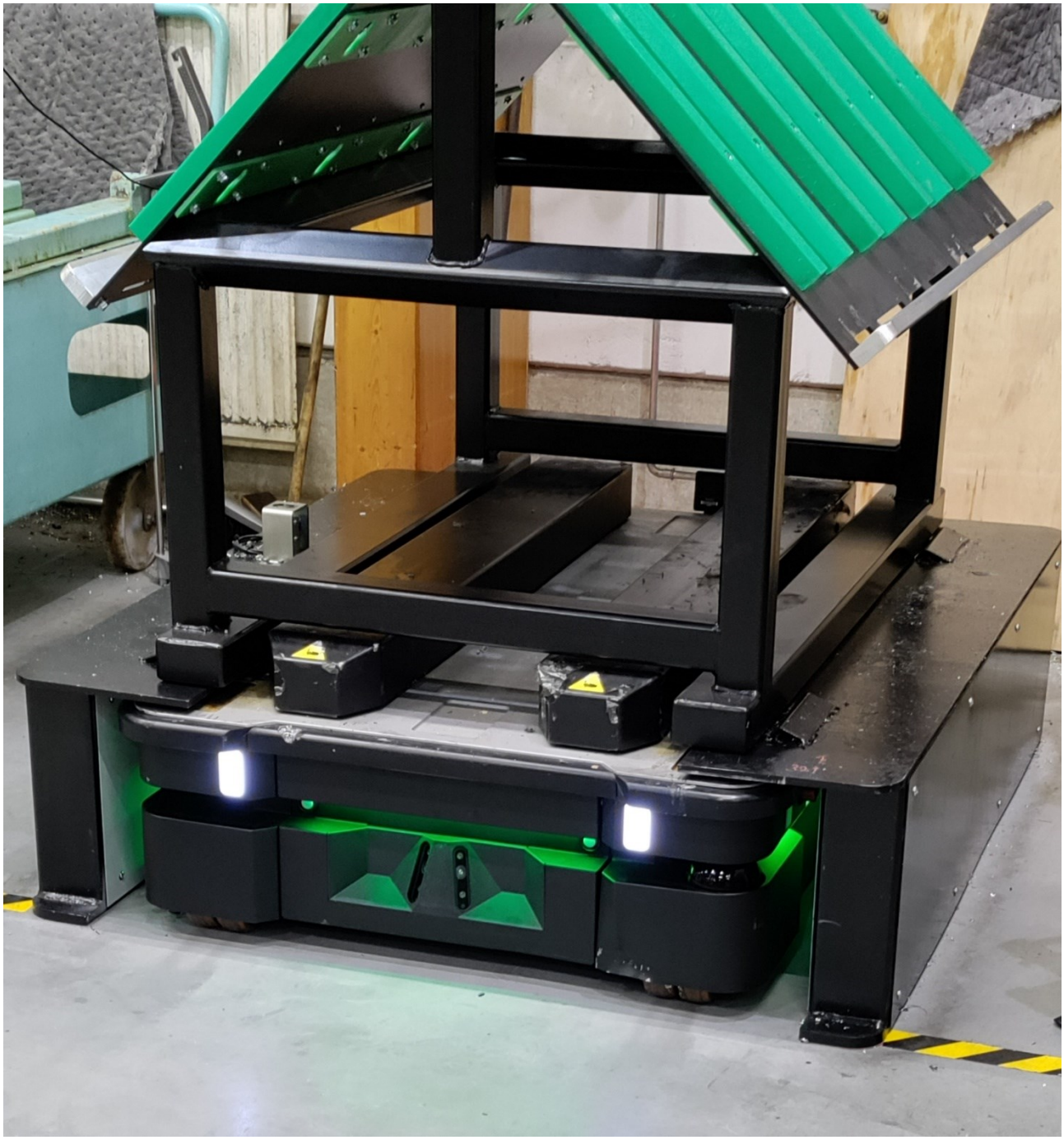
Kuva 6. Eurolavateline MiR-robotille

Kuvassa 6 on mobiilirobotin tarvitsema eurolavateline. Tapa, jolla robotti käyttää paikkaa hyödykseen, on seuraava: robotti asemoi itsensä paikan eteen ja joko nostaa tai laskee oman nostimensa riippuen siitä, onko se viemässä vai hakemassa lastia. Tämän jälkeen robotti ajaa telineeseen siten, että on kokonaan telineen sisällä. Lopuksi robotin nostin tekee päinvastaisen liikkeen kuin ulkopuolella, minkä jälkeen robotti poistuu telineestä.



Kuva 7. Mobiilirobotti toimittamassa lastin.

Kuva 7 esittää aiemmin mainittua toimintaa, jossa robotilla on nostotelineet yläasennossa ja se on paikoittumassa telineeseen. Kuvassa robotti paikoittaa itseään takaperin telineeseen, kun normaalitilanteessa robotti menisi etuperin.



Kuva 8. Mobiilirobotti paikoittunut noutopaikkaan

Kuvan 8 robotti on paikoittunut telineeseen onnistuneesti. Kuten kuvasta näkyy, robotin kummallakin puolella on melko vähän tilaa. Tästä syystä on tärkeää, että robotti ei kadota lokalisaatiotaan. Kuvassa robotti näyttää vihreää statusvaloa, mikä kertoo robotin olevan valmis suorittamaan seuraavaa tehtävää.

Robotin käyttöönoton yhteydessä runsaan testauksen tuloksena kävi ilmi, kuinka dynaaminen ympäristö eli usein muuttuva ympäristö teki mobiilirobotin toiminnasta epävarmaa. Toimintavarmuuden parantaminen vaatii toteutuskohteen yrityksen työntekijöiltä toimintamuutosta, mikä edistää yrityksen yleisjärjestystä, kun kuormalavoja ei jätetä kulkuväylille miten sattuu. Lisäksi erittäin dynaamisille alueille lisättiin robotille staattisia eli kiinteitä pisteitä. Muutos käytännössä tarkoittaa esimerkiksi materiaalihyllyn päähän robotin laserskannereiden korkeudelle heijastamattomien levyjen lisäämistä. Tämän lisäksi testauksen aikana kävi selväksi, että kulkuväylät olivat riittävän leveät robotin sujuvaa kulkemista varten. Robotin turvaominaisuuksien testaamisen ja turvallisuuden varmistamisen osalta ei löytynyt merkittäviä puutteita. Ainoa merkittävä riski robotin kanssa syntyy robotin paikoittaessaan itseään eurolavatelineisiin: Jotta robotti pääsee telineisiin sille tarkoitettuun paikkaan, tulee robotin vähentää omaan turvapiiriään. Tätä riskiä on minimoitu kouluttamalla henkilöitä, jotka toimivat näiden paikkojen läheisyydessä. Lisäksi telineet ja robotin toiminta-alue on merkitty huomionauhoin, robotti itsessään vilkuttaa valoja ja soittaa äänimerkkiä aina turva-alueiden ollessa vähennettynä.

Robotin käyttöönoton jälkeen robotti aloitti suorittamaan valmiiden kappaleiden ja raakojen kappaleiden kuljetusta ilman suurempia ongelmia. Käyttöönottoprosessin aikana tuli tietenkin tehdä pieniä muutoksia toiminnan helpottamiseksi. Lisäksi koulutettiin käyttäjiä ja robotin rinnalla työskenteleviä. Mobiilirobotin käyttö vähentää trukkiliikennettä ja vähentää ihmisen turhaa edestakaista liikennettä.

6.2 Kommunikaatio

Kommunikaatiolla tarkoitetaan yhteyttä, joka vaaditaan kahden laitteen välillä toimintojen toteuttamiseksi. Kommunikaation kautta saadaan tietoa toisesta osapuolesta, minkä pohjalta voidaan antaa omaa tietoa toiselle osapuolelle. Työssä pääasialliset osapuolet ovat käyttöliittymä ja mobiilirobotti, mutta myös käyttöliittymän ja MiR Fleetin välillä vallitsee kommunikaatio. MiR-mobiilirobotin kanssa kommunikointi onnistuu vain kahdella tavalla: Joko käytetään tämän työn tavoin REST-kommunikaatiota tai käytetään Modbus-protokollaa.

MiR-robotilla on omia sisäisiä PLC-rekistereitä, jotka soveltuvat erinomaisesti kommunikaation välikappaleeksi. PLC-rekisterit ovat yksinkertaisuudessaan kaksi erilaista numeromuistia, joissa toiseen voidaan tallentaa kokonaislukuja ja toiseen desimaalilukuja. Rekisterejä on mahdollista kirjoittaa ja lukea REST-kommunikaation kautta. Rekisterit eivät lähtökohtaisesti viittaa mihinkään tiettyyn tai tarkoita mitään tiettyä, tästä syystä voidaan päättää rekisterille omia merkityksiä. Esimerkiksi PLC-rekistereitä voidaan käyttää niin, että robotti kirjoittaa rekisteriin 1 arvon 1, joka voisi tarkoittaa esim., että robotti on saapunut sille tarkoitettuun paikkaan. Tämän jälkeen robotti jäisi odottamaan, että rekisteri 2 saisi arvon 1, mikä voisi tarkoittaa, että robotti saa jatkaa matkaa.

6.2.1 REST ja Modbus

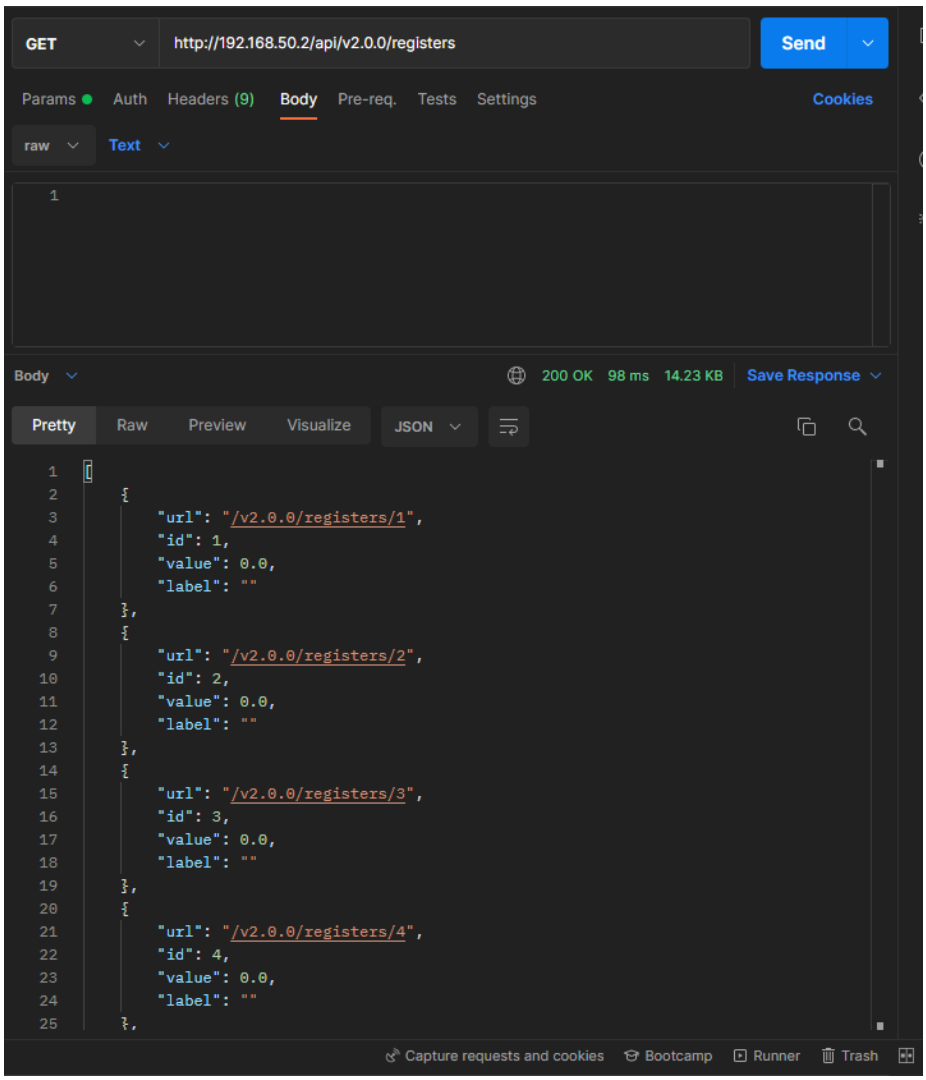
REST-kommunikaatiota käsitellään teoriaosassa enemmän. REST on tilaton kommunikaatio muoto, mikä tarkoittaa, että esimerkiksi käyttöliittymän ja robotin välillä ei passiivisesti kulje tieto tilamuutoksista. Kaikki tilamuutokset tulee pyytää robotilta omaan sovellukseen sopivalla aikavälillä. Robotti ei itse kysele päätelaitteelta mitään, joten ainoat viestit, jotka robotin suuntaan lähtevät, ovat käyttäjän muutoksia. REST valikoitui tavaksi kommunikoitua, koska dokumentaatio oli laajempaa ja parempilaatuista. Lisäksi toteutus oli tekijälle helpompi.

Wagonin (i.a.) mukaan Modbus-protokolla on tiedonsiirtoprotokolla, joka perustuu hieman REST-tyylin tavoin käyttäjä/palvelin-malliin. Modbus perustuu 1979 tehtyyn ohjelmoitavien logiikoiden Modbus-protokollaan. Modbusin tarkoitus on olla yksinkertainen ja tehokas protokolla tiedonsiirtoon. Universaali datarakenne varmistaa toiminnan eri valmistajien laitteiden välillä. Modbus-protokollaa esiintyy kolmessa erilaisessa käyttötilassa, mutta Modbus TCP on ainoa tiedonsiirtotapa käyttäjä/palvelin-yhteydessä. Vaatimuksena on, että laitteet on yhdistetty samaan IP avaruuteen.

6.2.2 Postman

Postman on API-alusta ohjelmointirajapintojen rakentamiseen sekä käyttöön (Postman, i.a.). Postman yksinkertaistaa ohjelmointirajapinnan ylläpitoa ja suoraviivaistaa yhteistyötä, jotta voidaan luoda parempia ohjelmointirajapintoja nopeammin.

Postmanista löytyy lukuisia ominaisuuksia ja toimintoja, joita ei tarvittu työn tekemiseen, mutta se helpottaa huomattavasti kommunikoinnin havainnointia ja testausta. Postman toimi työkaluna REST-tyylin ymmärtämiseen ja tarvittavien toimintojen testaukseen sekä rakentamiseen. Kuva 9 havainnollistaa, mitä REST-kommunikaatio palauttaa.



Kuva 9. Postman-näkymä onnistuneen REST-viestin jälkeen

Kuvassa 9 ylälaidassa näkyy osoiteriviä vastaava kirjoituskenttä ja kentän vasemmalla puolella on REST-metodi, jota käytetään viestin lähetyksessä. Kirjoituskenttään tulee antaa osoite, josta haluaa viestin saada. Ensimmäisenä annetaan IP-osoite, johon yhteys halutaan muodostaa. IP-osoitteen jälkeen määritellään ohjelmointirajapinnan versio, jota käytetään. Tämän jälkeen päästään käsiksi erilaisiin tietovaihtoehtoihin. Esimerkiksi kuvassa oleva päätepiste ”registers” on päätepisteenä robotin sisäisille PLC-rekistereille, mikä palauttaa päätepisteen sisältämän tiedon. Esimerkkinä tämä voitaisiin tehdä myös

tarkemmaksi siten, että lisätään perään vielä rekisterin numero, kuten ”registers/1”. Edellä mainittu esimerkki palauttaa vain rekisterin 1 arvon. Paluuarvoja voi tarkastella kuvan 9 puolenvälin alapuolelta.

Päätypisteiden tulee olla hyvin dokumentoituna, koska niitä ei saa selville mistään. Tästäkin syystä Postman toimii erinomaisena testausta tukevana työkaluna, koska sen avulla voidaan selvittää mikä toimii, miten se toimii ja mitä se palauttaa. Tämän jälkeen on huomattavan helppoa poimia tarvittu tieto talteen.

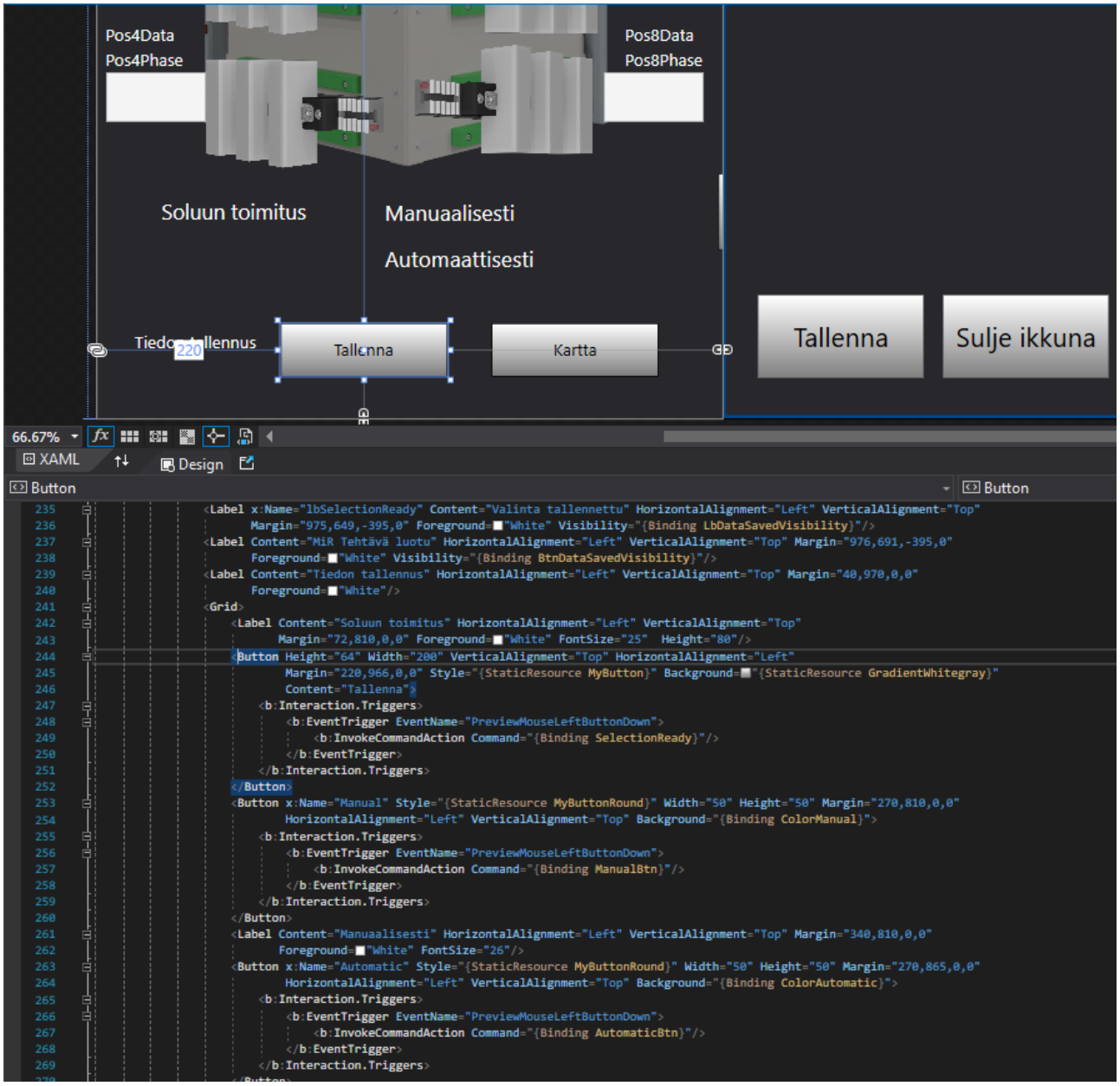
6.3 Käyttöliittymä

Kun kehitetään minkä tahansa tyyppistä sovellusta tai opetellaan uutta kieltä, käytettään Visual Studion Integrated Development Enviroment (IDE)-ympäristöä (Microsoft, i.a.). Koodin editoinnin lisäksi Visual Studio IDE tuo graafisen suunnittelun, koodi ehdotus kirjoitustyökalut, laajennukset ja monet muut ominaisuudet yhteen paikkaan.

Käyttöliittymän toiminta on osa laajempaa kokonaisuutta kuin vain mobiilirobotin ohjaaminen, mutta tässä työssä käsitellään vain mobiilirobottiin liittyvää osuutta. Työhön rakennettu käyttöliittymä on koodattu Microsoftin Visual Studio-ohjelmistoympäristöllä. Käyttöliittymän graafiset elementit on tehty Windows Presentation Foundationilla (WPF) ja taustalla toimiva logiikka on kirjoitettu C#-kielellä. Käyttöliittymä asennettiin Beckhoff-paneeli-PC:lle, jossa käyttöjärjestelmänä toimii Windows. Paneeli-PC on myös kosketusnäyttöinen, mikä tulee huomioida graafisten elementtien suunnittelussa käytettävyyden säilyttämiseksi.

6.3.1 Graafinen osuus

Kun Visual Studiolla tehdään käyttöliittymiä, ovat vaihtoehdot Microsoft Foundation Class (MFC), Windows Forms tai Windows Presentation Foundation (WPF). MFC on tarkoitettu C++-ohjelmointikielellä tehtyihin käyttökohteisiin. Windows Forms on käyttöliittymäkehys, kuten WPF, mutta Forms on vanhempaa teknologiaa. Tämä johti WPF-kehyksen valintaan käytettäväksi käyttöliittymän ympäristönä.



Kuva 10. Käyttöliittymän graafisen elementin koodia

Kuvassa 10 on näkymä itse graafisista elementeistä, sekä niiden luomiseen ja muokkaamiseen tarkoitettu koodi. Kuvassa alempi puolisko on XAML-kieltä, missä voi tuoda valmiita elementtejä tarpeen mukaan tai luoda omia. Tässä kuvassa on käytetty vain valmiita elementtejä. XAML-koodissa valkoinen teksti on elementin nimi tai elementin toiminto. Vaaleansininen on elementin parametri, joka määrittää millainen elementistä tulee. Tummansininen on taas parametrin muuttuja, joka on parametrin määrittelevä arvo. Viimeisenä värinä oranssinvärisen tekstit liittyvät MVVM-ohjelmointityylin sidontoihin, eli yksinkertaisuudessaan siirtävät tietoa graafisesta näkymästä taustalla toimivaan logiikkaan ja taas logiikasta näkymään.

XAML-koodi vaatii, että sen katseluun tottuu, ennen kuin osaa nopealla katsauksella sanoa, mitä se tekee. Graafista näkymää rakentaessa on hyödyllistä katsella kumpaakin näkymää yhtä aikaa, koska toiminta on helpommin hahmotettavissa. Samalla näkee missä elementit tulevat sijaitsemaan sovelluksessa. Graafinen osuus on ainoa osuus, joka näkyy loppukäyttäjän suuntaan. Tästä syystä on hyvä pitää mielessä toiminnallisuus ja yksinkertaisuus. Toiminnallisuus on otettuna huomioon, koska loppulaite on kosketusnäyttöinen laite. Kosketusnäytön toiminnallisuus saattaa helposti unohtua, kun testausta suoritetaan perinteisellä hiirellä ja näppäimistöllä.

Yksinkertaisuus on kaunista varsinkin, jos se tarkoittaa, että kuka tahansa osaa käyttää yksinkertaista sovellusta. Aivan turhaa on tehdä tarpeettoman monimutkainen sovellus, jos sitä ei osaa käyttää kukaan muu kuin sovelluksen tekijä. Yksinkertaisuus ei tietenkään tarkoita, että sovelluksen tarvitsee olla kankea ja kapea ominaisuuksiltaan. Pikemminkin tarkoitus on tehdä visuaalisista elementeistä yksinkertaisia ja helposti lähestyttäviä. Tässä työssä painettavat napit ja muut kosketusta vaativat elementit suunniteltiin osittain jopa liian suuriksi. Tämän lisäksi yksinkertaisuuden varmistamiseksi päädyttiin ratkaisuun, missä mahdollisimman suuren osuuden päätöksistä tekee ohjelma, käyttäjälle näytetään vain tarvittavat elementit.

6.3.2 Taustalogiikka

Työssä käytettiin ohjelmointitapaa nimeltä MVVM, mikä tarkoittaa yksinkertaisuudessaan, että näkymille on omat ViewModel-näkymämallinsa, ja näkymämalleilla saattaa olla omat mallinsa, joista näkymämalli perii toimintoja. MVVM tarkoittaa myös, ettei itse graafisten ikkunoiden taustakoodiin kirjoiteta muuta kuin linkitys näkymämalliin. Kuten kuvasta 10 voidaan nähdä, tiedonsiirto ja toiminnallisuus siirtyvät MVVM:n myötä sidonnan kautta näkymämalliin. Tämän lisäksi on hyvä tiedostaa, että graafisissa elementeissä ei ole mitään oikeaa toiminnallisuutta. Tämä tarkoittaa, että vaikka näkymässä olisi painikkeita, ei niiden painaminen tee mitään ennen kuin taustalle rakennetaan toiminnallisuus napin painallukseen. Kaikki näkymän toiminnallisuus on sen näkymän näkymämallissa toteutettuna.

Taustalogiikassa on tarvittu ylimääräisiä kirjastolaajennuksia toivotun toiminnan saavuttamiseksi. Ensimmäisenä merkittävänä kirjastona on RestSharp-niminen kirjasto, joka

mahdollistaa REST-käyttäjän luomisen ja REST-viestin sekä metodin luomisen. Ilman kirjastoa ei ole mahdollista lähettää REST-viestejä, mutta kirjaston sisältämät metodit ja luokat on mahdollista tehdä itse, jos ei ulkopuolisia kirjastoja halua ladata. Toisena kirjastona on käytetty Web.WebView2-nimistä kirjastoa. Tämä kirjasto mahdollistaa verkkosivun näyttämisen ja toiminnallisuuden säilyttämisen käyttöliittymän sivulle. Tämä toiminto on tarpeen, jotta pääsee käsiksi robotin omaan käyttöliittymään, joka sisältää paljon enemmän tietoa robotista kuin itse WPF-käyttöliittymä. Robotin näkymä on tärkeä esimerkiksi vikatilanteessa tai pienten muutosten tekemisessä. Näiden kirjastojen lisäksi kaikki toiminnot oli mahdollista toteuttaa C#-kielen omilla ominaisuuksilla.

Taustalogiikalle ei annettu pysyvää muistia tietojen hallinnointia varten, vaan kaikki tieto, mikä vaatii sen säilömistä, kirjoitettiin tietokantaan talteen. Tietokannan lisäksi tiedon siitä, miten käyttöliittymän tulisi toimia tietyissä tilanteissa, sai järjestelmään liitetyn logiikan ja siihen yhdistettyjen antureiden kautta. Antureiden avulla käyttöliittymän taustalogiikka tietää esimerkiksi, mikä robottipaikka on vapaana mobiilirobottia varten.

6.3.3 Mobiilirobotin ohjaus

Käyttöliittymällä ei suoraa manuaalista ohjausta pysty tekemään, mutta se ei ole tarkoituksenakaan. Käyttöliittymän kautta voidaan antaa robotille tehtäviä, jotka se osaa autonomisesti suorittaa sille määritettyjen parametrien kautta. Oikeastaan tavoite on, että itse robotin toimintaan tarvitsee mahdollisimman vähän puuttua.

Robotin ohjaus eli tehtävien parametrisointi on pyritty tekemään mahdollisimman yksinkertaiseksi käyttäjälle. Yksinkertaisena järjestelmänä ohjaus on myös helppo omaksua, käyttäjän on helppoa työskennellä robotin rinnalla. Kuvassa 10 näkyvä Tallenna-painike viimeistelee ja luo tehtävän robotille. Osa robotin vaatimista tiedoista tulee käyttäjän antamina, osan tiedoista käyttöliittymän taustalogiikka kerää valmiiksi, mikä helpottaa käyttöä. Pääasiallisesti toiminnot on pyritty toteuttamaan yhden napin painalluksen taakse, jolloin käyttäjä napin painalluksella saa robotin suorittamaan tehtävää. Käyttäjä voi itse tehdä jotain muuta robotin suorittaessa tehtäväänsä.

Robotilla on kahden tyyppisiä tehtäviä, jotka vaikuttavat siihen, kuinka tehtävä luodaan taustalogiikassa robotille. Tehtävätyypit ovat parametriset tehtävät ja kiinteätyyppiset tehtävät. Kiinteätyyppisissä tehtävissä ainoa tieto, mikä tarvitaan, on tehtävän uniikki id-arvo, jotta voidaan luoda tehtäväpyyntö robotille.

Taulukko 2. MiR-tehtävän id-muoto

From221To321	91faa080-1082-11ec-a5df-00012978eb4d
From222To321	bc8fd63e-1082-11ec-a5df-00012978eb4d

Taulukossa 2 on nähtävissä kahden kiinteätyyppisen tehtävän id-arvot. Taulukon vasemmassa sarakkeessa on tehtävän nimi, joka on puhtaasti ihmistä varten kertomaan hieman luettavammin, mitä robotti tekee. Oikeassa sarakkeessa taas on tehtävän id-arvo. Silmämääräisesti kirjain- ja numerosarja ei kerro juuri mitään, mutta robotille se kertoo kaiken tarvittavan tehtävän suorittamiseksi. Käyttöliittymän taustalogiikka päättelee muuttujien perusteella, mikä id-arvo tulee valita oikean toiminnon suorittamiseksi. Tällöin käyttäjän ei milloinkaan tarvitse nähdä tai olla tietoinen tehtävän id-arvosta.

Taulukko 3. MiR-paikkojen id-tunnukset

LoadingPosition_1	7a3aec01-0f10-11ec-a5df-00012978eb4d
StandbyPosition_1	920c904c-120a-11ec-8454-00012978eb4d

Taulukossa 3 on vuorostaan nähtävissä liikepisteiden id-arvot. Taulukon 3 rakenne on vastaava kuin taulukon 2 eli taulukon vasen sarake on varattu id-arvoa vastaavalle nimelle ja oikea sarake itse id:lle. Kaikilla tehtävillä on oma id, kuten kaikilla kartan elementeilläkin. Parametrisiä tehtäviä käyttäessä tulee olla tiedossa tarvittavien parametrien tiedot. Tämän työn kohdalla parametreinä tuli antaa lähtöpisteen ja kohdepisteen tieto. Jokainen toteutuskohde on erilainen ja tarpeiden mukaan parametrejä voisi olla huomattavasti enemmänkin kuin vain kaksi kappaletta. Id-arvon uniikin luonteen takia voidaan id-arvoja säilöä kiinteinä muuttujina käyttöliittymän muistissa. Käyttöliittymällä robotin ohjaus siis kiteytyy eri tehtävien antamiseen. Lisäksi käyttöliittymän kautta on mahdollista ohjata robottia ilman tehtävärakennetta, käyttäen hyväksi robotin omaa käyttöliittymää. Tämä ei ole kuitenkaan tarkoitettua toimintaa pyrittäessä mahdollisimman autonomiseen toimintaan.

Itse ohjaamisen lisäksi käyttöliittymän kautta on mahdollista lukea robotin tehtävähistoriaa tai esimerkiksi robotin akun tilaa. Näiden tietojen saamiseksi tulee lähettää robotille

kyselyviesti säännöllisen ajanjakson välein, koska käytetyn REST-kommunikaation tilattomuus ei muuten tarjoa muutoksia tiedoissa. Nämä tiedot ovat vain käyttöliittymän käyttäjän tiedoksi. Jotta taustalogiikka voi käyttää tietoja, ei niitä tarvitse tuoda käyttöliittymälle näkyviin. Mobiilirobotilta on mahdollista saada suuri joukko erilaista tietoa sekä toteuttaa erilaisia toimintoja käyttäen REST-kommunikointia. Tätä varten robotin valmistaja on koonnut kaikki REST-kommunikointiin tarvittavat päätypisteet yhteen dokumenttiin, johon perehtymällä on mahdollista toteuttaa tarvittavat toiminnot. Tämän työn aikana tarvittiin vain murto-osa mahdollisista ominaisuuksista. Tästä syystä ei niitä sen tarkemmin ole tarve käsitellä tämän työn yhteydessä.

6.4 Beckhoff-kommunikaatiorakenne

REST-kommunikaatio on mahdollista toteuttaa myös Beckhoffin ohjelmoitavilla logiikoilla. Varsinaisen työnosuuden lisäksi tuli perehtyä, kuinka on mahdollista ohjelmoitavin logiikoin suorittaa kommunikointia mobiilirobotin kanssa, ja luoda siitä jonkinlainen pohja tai vakioitu versio tulevaa käyttöä silmällä pitäen.

6.4.1 Kuinka voidaan tehdä

Beckhoff tarjoaa kirjastoja erilaisten IOT-ratkaisujen tekemiseen. Näistä kirjastoista löytyy myös mahdollisuudet muodostaa REST-kommunikaatio olemassa olevaan palvelimeen. Tämä tehdään käyttämällä funktiota TF6760, joka sisältää HTTPS- ja REST-kommunikointiin olemassa olevan IotBase-kirjaston. Tämän lisäksi samasta funktiosta löytyy kirjasto nimeltä JsonXml, joka mahdollistaa Json-tyyppisten viestien kokoamista ja purkamista. Tämä on olennaista, koska REST-kommunikaatio tapahtuu Json-tietotyypissä.

Beckhoffin infosys on dokumenttikirjasto kaikista Beckhoffin ohjelmointitoiminnoista ja mahdollisuuksista. Eri kirjastojen dokumentoinnissa on hieman eroja riippuen aihealueesta, mutta pääasiallisesti infosysin kautta on mahdollista saada hyvä käsitys kirjaston toiminnasta ja sen sisältämistä toiminnoista. Esimerkiksi REST-kommunikaation toteutukseen löytyy useita esimerkkejä, kuinka se on mahdollista toteuttaa eri tavoin ja näiden esimerkkien kautta omat ratkaisut ovat kehittyneet. Toisaalta myöhemmin tarvittu Jsonin

kirjoitus vaati huomattavasti enemmän omaa oivallusta hieman puutteellisen dokumentaation takia.

REST-kommunikaation muodostamiseksi tarvitaan vähintään kolme valmista funktiota, jotka sisältyvät IotBase-kirjastoon. Lisäksi käytettävän logiikan tulee pystyä muodostamaan yhteys samaan IP-avaruuteen robotin kanssa, jotta kommunikointi on mahdollista. Ensimmäisenä IotHttpClient-funktiolla luodaan logiikalle käyttäjä, jolla voidaan ottaa yhteys palvelimeen. Toisena on IotHttpHeaderFieldMap-funktio, jolla annetaan käyttäjälle riittävät oikeudet ottaa yhteys. Viimeinen tarvittu funktio on IotHttpRequest. Viimeinen funktio toimii siltana itse kommunikoinnille ja metodeillaan toteuttaa viestiliikenteen määritysten mukaan. REST-kommunikaation tilattomuuden vuoksi, jos haluaa tiedon robotin tilamuutoksista, niin tulee suorittaa säännöllisesti kyselyitä robotille.

6.4.2 Pohjan rakennus

Beckhoffin logiikalle ohjelmoitiin kommunikaatiopohjaratkaisu, josta on tarpeen mukaan helppo ottaa kommunikaatio käyttöön muissa projekteissa. Tätä pohjaa varten tuli perehtyä ja selvittää kuinka rakenteesta saadaan mahdollisimman joustava ja dynaaminen, jotta se vastaa erilaisia käyttökohteita. Kommunikaatiosta muodostettiin Beckhoffin kirjastoista saatavien funktioiden tapaan funktio, jolla on omat metodinsa ja toimintonsa.

Tässä kohtaa testatessa kävi ilmi ohjelmiston omat rajoitteet, jotka pakottivat edellä mainitun JsonXml-kirjaston käytön. Dynaamisuutta tavoitellessa on mahdollista haluta saada robotilta tieto usean PLC-rekisterin tilasta, mikä ei onnistu ilman että viestin sisällöstä muodostetaan Json-viesti.

6.5 Siemens-kommunikaatio

Siemensin logiikoillakin on mahdollista muodostaa REST-kommunikaatio yhteys REST-palvelimeen. Siemensillä on hieman vastaavasti, kuten Beckhoffilla, kirjasto http-kommunikaatiota varten. Siemens-kirjasto on LHTTP ja sisältää GET- ja POST-metodit kommunikointia varten ja muutaman funktion tiedon käsittelyä varten. Tästä huolimatta tarjolla olevat työkalut ovat huomattavasti suppeammat kuin Beckhoffilla.

Siemensin kommunikaatioon perehdyttiin Beckhoffin jälkeen, joten perustoimintaidea oli selvillä ja kommunikoinnin toteuttamiseksi vaaditut parametrit olivat tiedossa. Tästäkin huolimatta toteutuksen toimintaan saattamien oli hieman hankalaa puutteellisen dokumentaation takia ja Siemensin ja Beckhoffin erilaisuuksien takia.

6.5.1 Onnistunut kommunikointi

Siemensillä on mahdollista toteuttaa tämä kommunikointi. Toteutus rakentuu funktiopalikoiden LHTTP_GET ja LHTTP_POST ympärille. Get-funktiolla voidaan hakea olemassa olevaa tietoa REST-palvelimelta, kun taas Post-funktio lähettää mahdollisesti uutta tietoa ja muuttaa olemassa olevaa dataa. Olennaista oli tietenkin Siemensin logiikan yhdistäminen samaan verkkoon kuin REST-palvelin.

Siemensillä on periaatteessa mahdollista toteuttaa kaikki samat toiminnot kuin Beckhoffilla, mutta se vaati huomattavasti enemmän omaa työtä. Siemensillä onnistuttiin lukemaan MiR-robotilta rekisterin arvoja sekä kirjoittamaan rekisterin arvoja. Tämä todisti kommunikaation toimivaksi ja huomionarvoiseksi vaihtoehdoksi tulevissa toteutuksissa, joissa tarvitsee käyttää Siemensin ohjelmoitavaa logiikkaa ja MiR-robotteja yhteistoiminnassa.

6.6 Mobiilirobotti, standardi VDA 5050

BlueBotics (i.a.) kertoo VDA 5050 olevan uusi standardisoitu tapa AGV-kommunikaatioon. Tarkemmin standardi keskittyy kommunikaatioon yksittäisen robotin ja robotteja hallitsevan ohjausjärjestelmän välillä. VDA 5050-standardin tavoitteena on standardisoida kommunikaatio, jotta lukuisat eri valmistajien robotit olisi mahdollista saada keskustelemaan keskenään. Tyypillisesti jokaisella robottivalmistajalla on oma ohjelmistonsa robotin hallintaan. Tällöin robotti ei keskustele muiden valmistajien laitteiden kanssa. VDA 5050 on Saksan autoteollisuusyhdistyksen sekä materiaalinkäsittely- ja intralogistiikkayhdistyksen yhteinen hanke.

VDA 5050 on uusi standardi mobiilirobottien kommunikoinnin standardisointiin ja aktiivisessa kehityksessä (BlueBotics, i.a.). Vaikkakin standardi pääasiassa suoraan koskettaa robotin valmistajia, on hyvä perehtyä sekä olla tietoinen uusista kehitysaskeleista ja

toimintatavoista. Lisäksi standardin olemassaolon tunteminen mahdollistaa usean eri valmistajan robotin käytön yhdessä, mikäli robottienvalmistajat seuraavat standardin ehdottamia toimintatapoja.

7 TULOKSET

Työn tuloksena valmistui onnistunut toteutus. Mobiilirobotti tuli osaksi asiakasyrityksen päivittäistä toimintaa ja hoitaa sisälogistiikkaa sille suunnitellulla osuudella. Lisäksi käyttöliittymästä tuli yksinkertainen toteutus, joka on helppo uusienkin käyttäjien omaksua. Tämän työn osuus oli osuus suurempaa kokonaisuutta. Kokonaisuutena valmistui toimiva toteutus ja konsepti, jota työnantajayritys voi käyttää tulevia kohteita silmällä pitäen.

Beckhoffin ja Siemensin ohjelmoitaviin logiikoihin REST-kommunikaation toimintamallien tekeminen ja selvittäminen onnistui riittävän hyvin, jotta niitä voidaan hyödyntää tulevaisuudessa. Beckhoff-puolella saavutettiin enemmän, koska dokumentoinnin määrä ja laatu olivat paremmalla tasolla. Beckhoffin logiikoille valmistui dynaaminen ja monikäyttöinen kommunikaatiopohja. Siemensin kommunikaation toteuttamisen onnistuminen oli tärkeä merkkipaalu, koska ennalta oli suuria epäilyksiä siitä, onko REST-kommunikaatio ylipäätään mahdollista Siemensin logiikoilla. Mutta tämän työn tiimoilta on mahdollista todeta sen olevan toteutusvaihtoehto, vaikkakin se vaatii vielä työtä, jotta päästään samalle tasolle Beckhoffin kanssa.

8 POHDINTA JA YHTEENVETO

Tämän opinnäytetyön tavoitteena oli selvittää mobiilirobotin käyttömahdollisuudet hyvin dynaamisessa ympäristössä, minkä lisäksi tuli luoda esimerkkitoteutus, joka esittelee mobiilirobotin hyötyjä ja helppokäyttöisyyttä. Tämän lisäksi tarkoitus oli perehtyä Beckhoffin ja Siemensin ohjelmoitavien logiikoiden mahdollisuuksiin kommunikoida mobiilirobotin kanssa. Kaikki työn osa-alueet tukivat enemmän ja vähemmän toisiaan ja uusien asioiden oivaltaminen avusti toisissa asioissa. Tämä opinnäytetyö oli osa suurempaa kokonaisuutta, ja keskittyy vain osaan tehdystä kokonaisuudesta.

Työ eteni tasaisella tahdilla ilman suurempia vaikeuksia. Mobiilirobotin käyttöönotto oli suhteellisen yksinkertaista selkeiden järjestelmien ja yksinkertaisen tehtävän luonnin takia. Kuitenkin hieman monimutkaisimmissa tilanteissa ongelman kierto tai ratkaisu vaihtoehtoisin menetelmin hieman hidasti käyttöönottoa. Käyttöliittymän teossa meni huomattavasti enemmän aikaa, tässä työssä esitelty osuus on vain pieni osa kokonaisuudesta. Käyttöliittymän tekoon kului paljon aikaa vähäisen kokemuksen takia ja tämän lisäksi MVVM-ohjelmointitavan opettelu sekä omaksumien vei paljon aikaa. Käyttöliittymän teko oli kokonaisuudessa melko haastava kokemus, mutta opetti tekemisestä todella paljon. Ohjelmoitavien logiikoiden kommunikaation ratkaisu oli käyttöliittymän tavoin haastavaa, mutta idean oivaltamisen jälkeen nopeampi tehdä.

Työn lopputuloksena saatiin toimiva toteutus asiakasyrityksen käyttötarkoituksiin. Tämän lisäksi toimeksiantajayritykselle saatiin toimiva kokonaisuus, jota voidaan hyödyntää uusien vastaavien toteutusten kanssa. Mobiilirobotti on päivittäisessä ajossa palvelemassa, ja käyttö oli helppo ymmärtää. Lisäksi toimivat ohjelmoitavien logiikkojen kommunikointimallit valmistuivat onnistuneesti.

Toimeksiantajayritys pystyy hyödyntämään valmistunutta kokonaisuutta myyntiä edistävässä toiminnassa ja tulevissa vastaavissa toteutuksissa. Lisäksi yritys pystyy hyödyntämään ohjelmoitavien logiikoiden REST-kommunikaatiomahdollisuuksia tulevissa sovelluksissa.

Tulevaisuudessa kehitystyö jatkuu varmasti kaikkien tämän työn aikana tehtyjen asioiden osalta, esimerkiksi mobiilirobotin käytäntömallien vakioiminen ja erilaisten

käyttötarkoitusten kehittäminen. Tämän lisäksi logiikoiden kommunikointi pohjienloppuun hiominen, varsinkin Siemensin puolella pitää jatkokehittää saavutettua tulosta, jotta päästään samaan toimivuuteen kuin Beckhoffin puolella.

Toimeksiantaja ja asiakasyritys olivat tyytyväisiä saavutettuihin tuloksiin ja toteutuksiin.

LÄHTEET

BlueBotics. (i.a.). *VDA 5050 Explained – An overview of the evolving agv communication standard*. Haettu 1.3.2022, https://bluebotics.com/vda-5050-explained-agv-communication-standard/

Chowdhury, K. (2018). *Windows Presentation Foundation Development Cookbook*. Packt Publishing.

Evjen, B., Nagel, C., Glynn, J., Watson, K., & Skinner, M. (2012). *Professional C# 2012 And .NET 4. 5.* Wiley Pub., Inc.

Intel. (i.a.). *Autonomous Mobile Robots.* https://www.intel.com/content/www/us/en/robotics/autonomous-mobile-robots/overview.html

JTA Connection Oy. (i.a.). *YLI 20 VUOTTA TYÖTÄ VALMISTAVAN TEOLLISUUDEN HYVÄKSI.* Haettu 3.1.2022, https://www.jtaconnection.fi/jta-connection-yrityksena/

Kagan, E., Shvalb, N., & Ben-Gal, I. (2020). *Autonomous mobile robots and multi-robot systems: Motion-planning, communication and swarming*. Wiley.

Kandray, D. (2010). *Programmable automation technologies: An introduction to CNC, robotics and PLCs*. Industrial Press.

Kanjilal, J. (2013). *ASP.NET Web API: Build RESTful web applications and services on the .NET framework*. Packt Publishing.

Karp, S., & Stotts, L. B. (2013). *Fundamentals of electro-optic systems design: Communications, lidar, and imaging*. Cambridge University Press.

Liberty, J. (2005). *Programming C#* (4.p.). O'Reilly.

Microsoft. (30.11.2021). *A tour of C# language.* https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/

Microsoft. (i.a.). *Getting Started with Visual Studio IDE*. Haettu 22.2.2022, https://visualstudio.microsoft.com/vs/getting-started/

MiR Academy. (i.a.). *Creating your first map*. [Verkkokurssi]. MiR Academy.

Mobile industrial Robots (MiR). (29.4.2021). *MiR500 or MiR1000 Space requirements.* https://supportportal.mobile-industrial-robots.com/support-files/manuals/html/en/space_req_mir500_1000_1.6/content/_resources/pdf/mir500%20or%20mir1000%20space%20requirements%201.6_en.pdf

Mobile industrial Robots (MiR). (27.7.2021). *MiR1000 User guide 2.1.* https://supportportal.mobile-industrial-robots.com/support-files/manuals/html/en/mir1000_user_guide_2.1/content/_resources/pdf/mir1000%20user%20guide%202.1_en.pdf

Mobile industrial Robots (MiR). (28.10.2021). *MiR Network and WiFi Guide*. https://supportportal.mobile-industrial-robots.com/support-files/manuals/html/en/wifi_guide_2.4/content/_resources/pdf/mir%20network%20and%20wifi%20guide%202.4_en.pdf

Mobile industrial Robots (MiR). (14.1.2022). *MiR Fleet PC Getting Started*. https://supportportal.mobile-industrial-robots.com/support-files/manuals/html/en/mirfleet_pc_getting_started_1.7/content/mirfleet/intro/doc_history.htm

Mobile industrial Robots (MiR). (i.a.). *MiR1000.* https://www.mobile-industrial-robots.com/solutions/robots/mir1000/

Mobile industrial Robots (MiR). (i.a.-a). *AGV vs. AMR – What's the Difference?*. https://www.mobile-industrial-robots.com/insights/get-started-with-amrs/agv-vs-amr-whats-the-difference/

Mobile industrial Robots (MiR). (I.a.-b). *4 Common Questions About Autonomous Mobile Robots.* https://www.mobile-industrial-robots.com/insights/get-started-with-amrs/4-common-questions-about-autonomous-mobile-robots/

Mobile industrial Robots (MiR). (i.a.-c). *MiR EU Pallet Rack.* https://www.mobile-industrial-robots.com/solutions/mir-applications/mir-eu-pallet-rack/

Oitzman, M. (6.8.2021). *What’s the difference between an AMR and an AGV?* Mobile robot guide. https://mobilerobotguide.com/2021/08/06/whats-the-difference-between-an-amr-and-an-agv/

Postman. (i.a.). *What is Postman*. Haettu 21.3.2022, https://www.postman.com/product/what-is-postman/

Radhakrishnan, P. (2015). *Computer numerical control machines and computer aided manufacture* (2. p.). New Academic Science Limited.

Shivanand, H. K., Benal, M. M. & Koti, V. (2006). *Flexible manufacturing system*. New Age International (P) Ltd., Publishers.

Sick. (2014). *SICK AG Whitepaper: Safety Laser Scanners vs. Safety mats.* https://cdn.sick.com/media/docs/5/15/415/whitepaper_safety_laser_scanners_vs._safety_mats_en_im0058415.pdf

Sick. (2018). *SICK AG Whitepaper: LiDAR sensor functionality and variants*. https://cdn.sick.com/media/docs/3/63/963/whitepaper_lidar_en_im0079963.pdf

Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). *Introduction to autonomous mobile robots* (2. p.). MIT Press.

Tzafestas, S. G. (2014). *Introduction to mobile robot control*. Elsevier.

Vice, R., & Siddiqi, M. S. (2012). *MVVM survival guide for enterprise architectures in Silverlight and WPF: Eliminate unnecessary code by taking advantage of the MVVM pattern - less code, fewer bugs*. Packt Publishing.

Wago. (i.a.). *NOPEA TIEDONSIIRTO AUTOMAATIO- JA KENTTÄLAITTEIDEN VÄLILLÄ: MODBUS*. Haettu 21.3.2022, https://www.wago.com/fi/modbus

Yellavula, N., & Joshi, A. (2017). *Building RESTful Web Services with Go: Learn How to Build Powerful RESTful APIs with Golang That Scale Gracefully*. Birmingham: Packt Publishing.

LIITTEET