# Woord vooraf

Als masterstudent van de UHasselt-KU Leuven ronden wij onze opleiding af met een masterproef. Wij hebben de kans gekregen om onze masterproef te voltooien bij de onderzoeksgroep ACRO van de UCLL, deze is gelegen in het technologiecentrum te Diepenbeek.

Autonome robots komen meer en meer onze leefwereld binnen. Dit is de reden waarom wij voor deze masterproef hebben gekozen. Het leek het ons immers zeer interessant om zelf eens aan de basis te liggen van de opbouw en programmering van een autonome robot. In dit geval gaat het dan nog specifieker om een elektrische rolstoel, door hier een laadstation voor te ontwerpen en de rolstoel hier automatisch in te laten aanmeren kan het leven van fysiek beperkte personen enorm worden vergemakkelijkt, wat een extra drijfveer was om dit project tot een goed einde te brengen. Dit was een heel nieuwe ervaring, we hadden immers nog geen ervaring in dit vakgebied. Al snel kwamen we uit bij ROS, 'Robot Operating System', dit is een robot softwarebibliotheek waarmee autonome robots geprogrammeerd kunnen worden. Het vergde wel erg veel tijd om ROS onder de knie te krijgen.

Doorheen onze masterproef konden we altijd rekenen op een aantal personen. Deze zouden we dan ook zeer graag bedanken. Om te beginnen onze promotor dr. Ir. Demeester Eric, hier konden we altijd op terug vallen, vooral op gebied van robotica en programmering. Ook zouden we onze copromotor dr. Ir. Johan Baeten willen bedanken. Deze heeft ons vooral als docent altijd zeer goed opgeleid en begeleid, en dit gedurende heel onze opleiding. Als laatste willen we onze ouders bedanken omdat zij ons de kans hebben gegeven om verder te studeren en ons hier altijd ten volle bij steunden.

# Inhoudsopgave

# Lijst van tabellen

# Lijst van figuren

## Abstract

Mobiele robots zoals robotstofzuigers dringen ons dagelijks leven alsmaar meer binnen. Toekomstige mobiele robots zijn waarschijnlijk groter en zullen ook mensen en goederen transporteren in ongestructureerde omgevingen. Om voldoende quality of service te garanderen en batterijbeschadiging te vermijden, dienen deze robots zichzelf automatisch en tijdig op te laden. Het doel van deze masterproef, die plaatsvindt op ACRO te Diepenbeek, is een laadstation te ontwerpen waaraan een elektrische rolstoel automatisch kan aanmeren. Hiervoor moet er ook een kaart gemaakt worden van de omgeving waarin de rolstoel zich dan moet lokaliseren. Vervolgens moet er een pad berekend en uitgevoerd worden.

De thesis beoogde de volgende stappen. Op basis van een literatuurstudie wordt een eerste prototype van het laadstation ontworpen. Kaartopbouw en lokalisatie van de rolstoel en het laadstation in de kaart gebeurt met ROS, een verzameling van robotsoftwarebibliotheken. Voor kaartopbouw wordt gekozen voor het Gmapping algoritme. De rolstoel lokaliseert zich in de opgebouwde kaart met het AMCL algoritme en plant een pad naar het laadstation. De rolstoel voert dit pad uit naar het station. Na evaluatie van de lokalisatie- en uitvoernauwkeurigheid, is een iteratie van het laadstationontwerp mogelijk.

Het ontworpen prototype is uitvoerig getest. Transfer van encoderdata verloopt vanuit de rolstoelPC via UDP naar een laptop waarop kaartopbouw, lokalisatie en padplanning draaien. Enkel het uitvoeren van het gepland pad is niet gerealiseerd.

## Abstract in English

Mobile robots such as robot vacuum cleaners invade our daily life more and more. Future mobile robots are likely to be larger and will also transport people and goods in unstructured environments. To ensure adequate quality of service and avoid battery damage, these robots should recharge themselves automatically and on time. The purpose of this master's thesis, which takes place at ACRO in Diepenbeek, is to design a charging station to which a wheelchair can dock automatically. For this, there must first be a map of the environment in which the wheelchair will then have to locate. Next, a path must be calculated and executed.

The following design steps were aimed for. Based on a literature study, a first prototype of the charging station is designed. Map building and localization of the wheelchair and charging station in the map are done with ROS, a collection of robot software libraries. The Gmapping algorithm has been chosen for map building. The wheelchair locates itself in the map with the AMCL algorithm and plans a path to the charging station. The chair executes this path to dock at the station. After evaluation of the localization and docking accuracy, an iteration of the charging station design is possible.

The designed prototype has been tested. Transfer of encoder data is performed from the wheelchair computer via UDP to a laptop on which map building, localization and path planning are calculated. Execution of the planned path with docking has not been realized.

# 1 Inleiding

## 1.1 Situering

Deze masterproef vindt plaats bij ACRO (AutomatiseringsCentrum Research en Opleiding), een onderzoeksgroep van de KU Leuven. ACRO is actief in de vakgebieden automatisering, informatica en elektronica. Eén van hun onderzoeksfocussen is het gebruik van computervisie in automatisatie- en roboticatoepassingen; deze masterproef sluit daarbij aan.

In de toekomst zullen vele autonome wagens commercieel beschikbaar zijn. [1] Deze technologie kan ook gebruik worden binnen andere sectoren waar robots of zelfs elektrische rolstoelen en dergelijke gebruikt worden. [2] In de zorgsector kan zo het opladen van de elektrische rolstoelen geautomatiseerd worden. In het verleden verrichte de KU Leuven al onderzoek naar semi-autonome elektrische rolstoelen die zowel door de rolstoelgebruiker als door een computer bestuurd worden. [3] De toen gebruikte rolstoel zal ook voor deze masterproef gebruikt worden, zie hoofdstuk 3. Deze rolstoel is uitgerust met een computer, een laserscanner, encoders op de aangedreven wielen en de nodige hardware om de besturing over te nemen.

Figuur 1: De rolstoel uit vorig onderzoek

## 1.2 Probleemstelling

Mobiele robots zoals robotstofzuigers dringen ons dagelijks leven alsmaar meer binnen. Momenteel zijn deze robots nogal klein, waardoor ze redelijk eenvoudig kunnen navigeren in huiselijke omgevingen en ongevaarlijk zijn voor mensen, huisdieren of meubilair. Toekomstige mobiele robots zijn waarschijnlijk groter en zullen ook mensen en goederen transporteren in ongestructureerde omgevingen. Om voldoende *quality of service* te garanderen en batterijbeschadiging te vermijden, dienen deze robots zichzelf automatisch en tijdig op te laden. Gegeven hun grotere afmetingen en het mogelijk gevaar voor mens, dier en omgeving is dit geen evidentie.

In de zorgsector is er een bijkomend probleem. Hier kan voor fysiek beperkte personen het manueel opladen van hun elektrische rolstoel een enorme hinderblok zijn. Deze personen zijn dikwijls niet in staat om de huidige connectoren van de laadsystemen correct te verbinden met de rolstoel en daarnaast de stekker van het laadstation ook nog eens in het stopcontact te steken. Wanneer hun rolstoel niet tijdig is opgeladen, bestaat het probleem dat ze onderweg stilvallen en hun mobiliteit verliezen. Bovendien is het meestal niet eenvoudig om bestaande woningen en gebouwen aan te passen aan deze relatief grote rolstoelen. Om dit te vermijden zou het handig zijn dat (1) de rolstoel zich autonoom kan opladen, bij voorkeur op momenten dat de rolstoel niet in gebruik is zoals 's nachts. (2) En dat dit kan gebeuren in dagelijkse omgevingen waarvan geen kaart bestaat (en daarom snel opgebouwd moet kunnen worden). Hierbij is gebruiksgemak, snelheid van kaartopbouw en lokalisatie, nauwkeurigheid van navigatie en robuustheid tegen veranderlijke omgevingen van groot belang. Daarnaast is het dus voor fysiek beperkte personen zeer belangrijk dat de afmetingen van de bestaande rolstoel zo goed als mogelijk behouden blijven. De implementatie van het laadstation mag in geen geval de algemene mobiliteit van de gebruiker verminderen.

## 1.3 Doelstellingen

De hoofddoelstelling van deze masterproef is de integratie van kaartopbouw, lokalisatie, padplanning en paduitvoering opdat een elektrische rolstoel zich volledig zelfstandig kan opladen in willekeurige binnenhuisomgevingen. Meer specifiek beoogt de masterproef de volgende deeldoelstellingen:

- literatuurstudie naar bestaande laadsystemen voor mobiele robots;
- ontwerp van een laadstation voor de rolstoel;
- 2D kaartopbouw van een willekeurige binnenhuisomgeving m.b.v. één 2D laserscanner;
- globale lokalisatie[1] van de rolstoel en het laadstation in de kaart;
- padplanning naar het laadstation via ROS;

[1] De rolstoel kan zich bij opstart op elke plaats in de omgeving bevinden.

- nagaan met welke precisie de lokalisatie en padplanning gebeurt, wanneer dit mis loopt, en hoe eenvoudig en snel kaartopbouw gebeurt;
- de rolstoel het pad laten uitvoeren, hierbij mag er geen enkele botsing optreden tussen de robot en zijn omgeving;
- de rolstoel moet zelf detecteren wanneer het tijd is om zich naar zijn laadstation te begeven;
- de computer aangesloten aan de rolstoel moet in staat zijn vanzelf op te starten wanneer het oplaadproces moet beginnen.

## 1.4 Overzicht

Hoofdstuk 2 bespreekt het laadstationontwerp op basis van een literatuurstudie van bestaande laadstations. Hierbij wordt een onderscheid gemaakt tussen enerzijds de commerciële en anderzijds de niet commerciële laadstations. Hoofdstuk 3 licht het gebruikte voertuigplatform toe. Hierbij zullen onder andere de rolstoel, de laserscanner, encoders en de socket verbinding naar een bijkomende laptop met kaartopbouw en lokalisatie besproken worden. Hoofdstuk 4 beschrijft de kaartopbouw en de lokalisatie. Na een uitleg over de gebruikte softwarebibliotheek (ROS) licht dit hoofdstuk de basisprincipes achter SLAM ("Simultaneous Localisation And Mapping" oftewel kaartobpouw) toe. Vervolgens zullen er twee kaartopbouwmethodes worden uitgelegd: één methode die geen gebruik maakt van de encoderdata (Hector SLAM), en een andere die hiervan wel gebruik maakt(Gmapping) . Na een succesvolle kaartopbouw kan de robot zich beginnen te lokaliseren in de opgebouwde kaart; ter verduidelijking zal het hiervoor gebruikte AMCL algoritme kort worden toegelicht. Hoofdstuk 5 gaat in op de padplanning van de rolstoel vanuit zijn huidige positie tot de gewenste doelpositie, in dit geval het laadstation. Hoofdstuk 6 trekt ten slotte de voornaamste globale besluiten van de masterproef.

# 2 Ontwerp van een laadstation voor zelfrijdende rolstoelen

Eén van de doelstellingen van deze masterproef is het ontwerpen van een laadstation voor autonome elektrische rolstoelen. Sectie 2.1. legt kort de randvoorwaarden of eisen voor het ontwerp uit. Sectie 2.2 bespreekt daarop de uitgevoerde literatuurstudie van bestaande laadsystemen, dit als inspiratiebron en om best practices te bepalen. Vervolgens beschrijft Sectie 2.3 het ontwerp en de bouw van een eerste prototype. Dit prototype werd dan ook zorgvuldig getest n zoals besproken in Sectie 2.4. Voor de ontwikkeling van het definitieve laadstation moeten eerst de resultaten van het navigatiegedeelte gekend zijn, hierna kunnen, indien nodig, de laatste aanpassingen nog gebeuren.

## 2.1 Ontwerpvereisten

Commercieel beschikbare elektrische rolstoelen worden reeds geleverd met hun eigen lader, die door de fabrikant optimaal is afgestemd op het rolstoelsysteem en de batterijen. Figuur 2 (links) toont de rolstoellader en connector gebruikt in deze masterproef. De gebruiker dient de laadstationconnector aan de achterkant van de joystick te bevestigen, zie Figuur 2 (rechts). Dit vereist heel wat behendigheid die technisch moeilijk te evenaren is door de rolstoel zelf. Daarom focust dit hoofdstuk op een herontwerp van de laadstationconnector zodat de connectie eenvoudiger maar toch veilig op een automatische manier kan verlopen, zonder tussenkomst van een mens. De lader zelf wordt niet herontworpen, maar hergebruikt.

Figuur 2: Rolstoelbatterijlader met connector voor de gebruikte Invacare rolstoel (links). De lader vormt de wisselspanning van het net (230 V) om tot gelijkspanning voor de in serie geplaatste rolstoelbatterijen (2 x 12 V), en detecteert ook wanneer de batterijen volledig opgeladen zijn. De rechtse figuur toont de laadstationconnector bevestigd aan de achterkant van de joystick.

Bijkomende vereisten voor het ontwerp zijn:

- Het laadstation dient **veilig** te zijn, zowel op elektrisch (vermijden van kortsluiting of electrocutie) als op mechanisch (scherpe randen, uitstekende punten) gebied;
- er dient een **goed elektrisch contact** gerealiseerd te worden, met een lage contactweerstand. Gebruiksaanwijzigingen van de lader moeten via het ontwerp

automatisch worden gerespecteerd (bijv. volgorde van connectie: eerst met rolstoel, dan met netvoeding of omgekeerd);
- het laadstation moet **onnauwkeurige rolstoelnavigatie** (door lokalisatie- en paduitvoeringsfouten) in beperkte mate kunnen opvangen (ordegrootte 5 à 10 cm positie-onnauwkeurigheid en +/- 10 graden oriëntatie-onnauwkeurigheid);
- idealiter is het **ontwerp generiek**, i.e. eenvoudig toepasbaar op verschillende rolstoel- en ladertypes;
- **integratie in een huiselijke omgeving** of openbare gebouwen dient zo eenvoudig mogelijk te zijn (bijv. vermijden van mechanische verankering aan een muur, vermijden van bijkomende communicatie-infrastructuur tussen laadstation en rolstoel);
- **gebruiksgemak**: idealiter werkt het systeem out-of-the-box en is robuust tegen gebruiksfouten. Indien de gebruiker bijv. toch de locatie van de lader op een kaart moet aanduiden, dan moet het systeem idealiter rekening houden met een eventuele onnauwkeurige of zelfs onjuiste positionering van de lader op die kaart;
- het ontwerp is liefst zo **eenvoudig en goedkoop** mogelijk, zodat: **herstelling** bij eventuele beschadiging makkelijk en goedkoop verloopt, **montage eenvoudig** verloopt, etc.;
- het laadstation moet een **hoge levensduur** hebben ook al wordt het frequent gebruikt;
- het mag **niet veel plaats** in beslag nemen maar **toch stabiel** gepositioneerd staan (en zich niet verplaatsen wanneer de rolstoel aanmeert of opnieuw vertrekt);
- idealiter wordt de rolstoel op de hoogte gebracht wanneer de lader het sein geeft dat de rolstoel is opgeladen;
- zelfs bij kleine **verplaatsingen van het laadstation** in de omgeving slaagt de rolstoel er idealiter toch in om het laadstation te herkennen en daaraan aan te meren.

De meeste van bovenstaande vereisten gelden **voor beide connectoren van het laadstation**, zowel de connector die in de omgeving staat, als de connector die op de rolstoel staat.

## 2.2 Literatuurstudie

Aan de hand van een literatuurstudie worden de best bruikbare bestaande principes gezocht en gebruikt om een laadstation te ontwerpen dat zo goed mogelijk voldoet aan bovenstaande vereisten. Er wordt hier een onderscheid gemaakt tussen enerzijds commerciële uitvoeringen (sectie 2.2.1) zoals automatische grasmaaiers en stofzuigers, en anderzijds de niet-commerciële uitvoeringen (sectie 2.2.2). Hiermee worden prototypes uit andere papers bedoeld. Sectie 0 vergelijkt deze ontwerpen met de ontwerpvereisten en trekt hieruit besluiten.

### 2.2.1 Commerciële uitvoeringen

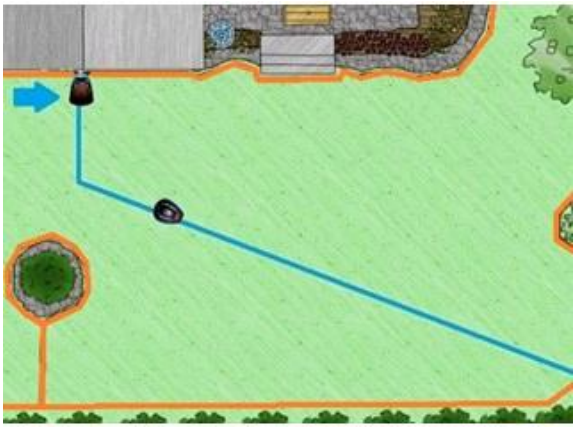
Bij een literatuurstudie naar commerciële uitvoeringen is er het grote probleem dat er weinig technisch diepgaande informatie over te vinden is. Dit komt omdat de meeste fabrikanten deze gegevens niet prijs willen geven aan hun concurrenten. Hieronder zullen dan ook alleen de werkingsprincipes beschreven worden van deze toestellen, dit enerzijds voor de automatische grasmaaiers en anderzijds voor de automatische stofzuigers. Voor elk geval bespreekt de sectie

zowel de navigatie tot bij het laadstation als de opbouw van het laadstation zelf. [4] [5] [6] [7] [8]

#### 2.2.1.1 Grasmaaier

##### Navigatie

De robotmaaier maakt typisch gebruik van twee systemen om te navigeren. Zo is er zowel een grensdraad als een begeleidingsdraad voorzien die beide op of in het gras gelegd dienen te worden. De grensdraad bakent het grasperk volledig af en wordt ook gelegd rond obstakels zoals bomen en bloemenperkjes. De robotmaaier detecteert deze grensdraad en zal hier nooit over rijden. De begeleidingsdraad wordt dwars door het grasperk gelegd vanaf het laadstation. Onderstaande afbeelding verduidelijkt dit met een oranje grensdraad en een blauwe begeleidingsdraad. Wanneer het maaiprogramma afgelopen is of de batterij bijna leeg is, gaat de robotmaaier op zoek naar de begeleidingsdraad en volgt deze tot het laadstation. Andere modellen maken geen gebruik van deze begeleidingsdraad maar volgen de grensdraad totdat ze het laadstation bereikt hebben.



Figuur 3: Grasmaaier met begeleidings- (blauw) en grensdraad (oranje) [6].

##### Laadstation en aanmeren

Wanneer de robotmaaier het laadstation heeft bereikt is het belangrijk dat deze op de juiste manier aanmeert. Door de begeleidingsdraad te volgen kan de robotmaaier gemakkelijk op het laadplatform rijden. Veel modellen van laadstations zijn ook zo ontworpen dat de robotmaaier uitgelijnd wordt. Aangezien de robotmaaier geen zwaar apparaat is kan zo een kleine fout gemakkelijk worden opgevangen. De connectie zal dan altijd juist gebeuren. Bij een rolstoel is dit wegens veel groter gewicht helaas niet mogelijk.


Figuur 4: Grasmaaierlaadstation met uitlijning [9].

Voordelen

- Aangezien de spanning voor het opladen slechts 24V is, dit is een veiligheidsspanning, dienen de laadconnecties niet afgeschermd te worden voor veiligheidsredenen.

Nadelen

- Een grensdraad is niet bruikbaar bij gebruik binnenshuis;
- het opvangen van een kleine navigatiefout bij aanmeren kan bij een rolstoel niet worden opgelost door deze uit te lijnen.

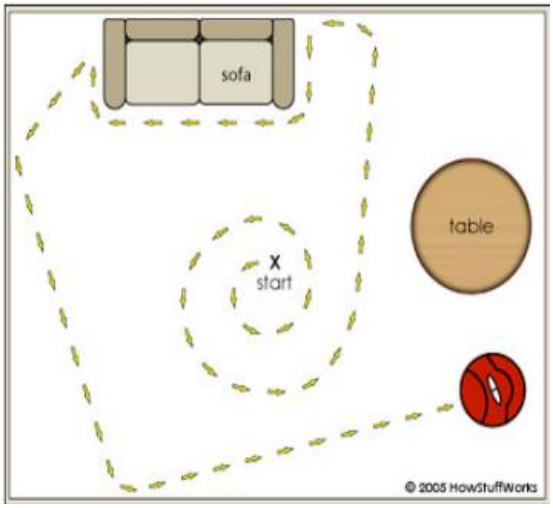
#### 2.2.1.2 Stofzuiger

##### Navigatie

Robotstofzuigers zijn op gebied van lokalisering op te splitsen in twee groepen: enerzijds is er de groep die zo goed als willekeurig in een kamer zal rondrijden, anderzijds zijn er de robotstofzuigers die een kaart van de kamer zullen maken.

##### Zonder kaartopbouw

Bij dit type zal de robotstofzuiger naar een willekeurig punt in de kamer rijden. Vanaf dit punt zal de robovac in spiraalvorm beginnen rond te rijden tot hij een obstakel tegen komt. De robotstofzuiger denkt nu dat hij het einde van de kamer bereikt heeft en zal vanaf dan de perimeter van de kamer afrijden in willekeurige richtingen. Dit gebeurt totdat zijn ingestelde werkingstijd om is, of de batterij leeg is. Nadelig aan deze manier van werken is dat er geen zekerheid is dat de robotstofzuiger op elke plaats in de kamer geweest is.




Figuur 5: Werkingsprincipe robovac zonder kaartopbouw [4]

### Met kaartopbouw

Bij robotstofzuigers met kaartopbouw zal de robot een virtuele kaart van de omgeving maken. Dit zal hij doen aan de hand van sensoren. Hiervoor worden afhankelijk van het type robot ultrasone sensoren, laserscanners of camera's gebruikt. Deze systemen werken hun kaart ook continu bij, zo zal de robot er ook rekening mee houden als er een voorwerp of persoon bijkomt in de kamer. Het voordeel van de methode met kaartopbouw is dat de robot nergens meer tegen zal botsen en dat hij niet nodeloos meerdere keren op dezelfde plaats zal komen. Hierdoor zal ook de werkingstijd verminderen.

### Laadstation en aanmeren

Om het laadstation te lokaliseren zal er gebruik gemaakt worden van infrarood signalen. Het laadstation zal een infrarood signaal uitzenden en de robot zal dit opvangen. Dit principe wordt duidelijk in onderstaande figuur. Vervolgens zal de robot in de richting van dit signaal proberen te rijden. Eenmaal de robot voor het laadstation staat zal hij dit in een rechte lijn proberen te benaderen, dit zal hij doen door continu zijn rijrichting bij te sturen totdat hij het laadstation heeft bereikt, dan kan het laden beginnen. De verbinding tussen het laadstation en de robot wordt gerealiseerd door twee oplaadstrips, deze hebben een bepaalde breedte zodat de positionering iets minder nauwkeurig moet zijn.

Figuur 6: Detectie van het laadstation a.d.h.v. infrarood sensoren [5].

Voordelen

- De robotstofzuiger kan dankzij de infraroodsignalen steeds de weg naar het laadstation terugvinden zonder dat hier extra aanpassingen aan de omgeving gemaakt dienen te worden;
- aangezien de oplaadstrips boven elkaar gemonteerd zijn is er zeker geen kans op kortsluiting;
- dankzij de breedte van de strips moet de positionering minder precies zijn.

Nadelen

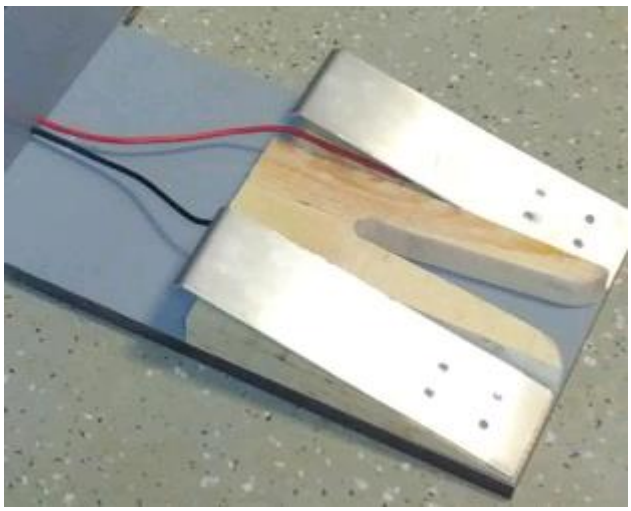
- Het implementeren van een systeem met infraroodsignalen is ingewikkeld en moeilijk om juist te programmeren.

### 2.2.2 Niet-commerciële uitvoeringen

Naast de commerciële uitvoeringen is er ook een studie gedaan naar niet-commerciële uitvoeringen. Dit zijn prototypes uit experimentele projecten [10] [11] [12]. Hierbij wordt vooral gekeken naar de constructie van de systemen. Eerst is er een korte beschrijving van de systemen, daarna volgt een opsomming van de positieve en negatieve punten met betrekking tot de vereisten voor het te ontwerpen laadstation.

#### 2.2.2.1 *Oplaadstrips*

In het eindwerk 'R&D of a loading and charging dock for an Autonomous Indoor Vehicle' werd een laadstion onworpen voor een AIV. [11] Bij dit systeem moet de robot over het laadstation rijde. Er worden bij het laadstation twee metalen oplaadstrips langs elkaar geplaatst, het principe wordt duidelijk in onderstaande figuur. Ook worden er twee oplaadstrips aan de onderkant van de robot voorzien. Doordat de oplaadstrips breed genoeg zijn ontstaat er een zekere marge. Om goed contact te voorzien, zijn er onder de oplaadstrips van het laadstation veren aangebracht, hierdoor worden de oplaadstrips van het laadstation stevig tegen die van de robot geduwd.



Figuur 7: Laadstation met oplaadstrips [11].

Voordelen

- Door de brede strips moet de robot zich minder precies positioneren bij het aanmeren, de marge is afhankelijk van de breedte van de strips;
- de robot hoeft niet recht op het laadstation staan, een kleine hoek is toegelaten;
- er is geen schade als de rolstoel te ver doorrijdt;
- eenvoudige constructie.

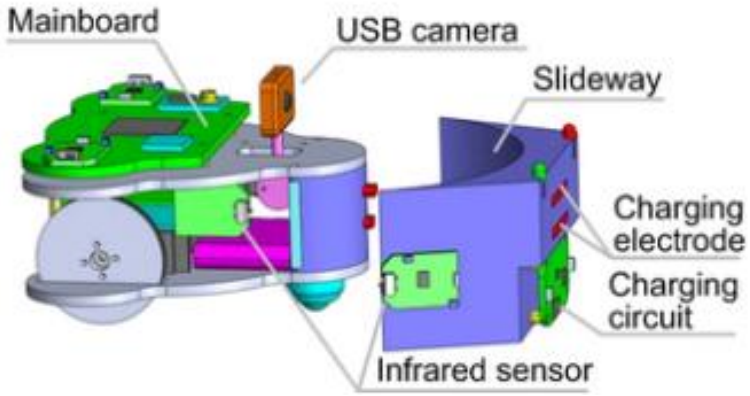
Nadelen

- Als de robot te scheef aanmeert dan is er kans op kortsluiting;
- bij vooruit aanmeren zullen voor dit project de voetplankjes van de rolstoel soms in de weg zitten omdat deze erg laag boven de grond kunnen staan, zie onderstaande figuur;
- achteruit aanmeren is niet mogelijk omdat de achterste wielen in de weg staan, zie onderstaande figuur.

Figuur 8: De voetplankjes van de rolstoel aan de voorzijde (links)en de zwenkwielen van de rolstoel aan de achterzijde (rechts) geven problemen bij het aanmeren bij het ontwerp van sectie 2.2.2.1.

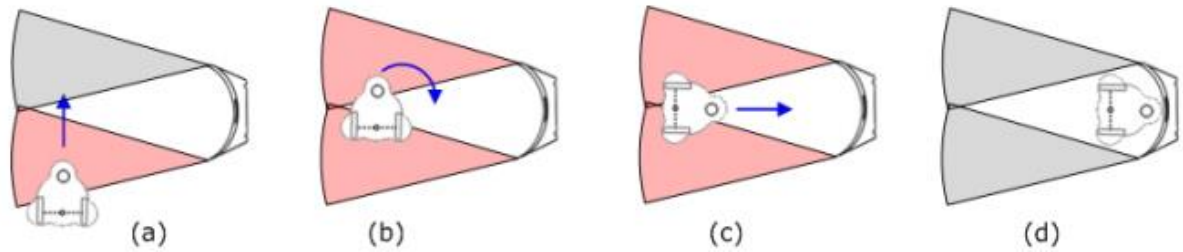
#### 2.2.2.2 *Holvormig* laadstation

Bij dit holvormig laadstation [10] zijn de twee oplaadstrips boven elkaar geplaatst, zie Figuur 9. De strips van het laadstation zijn iets breder, zodat er weer een extra marge ontstaat voor het aanmeren. Door de holvormige structuur is het niet noodzakelijk dat de robot loodrecht aan het station moet aanmeren.




Figuur 9: Holvormig laadstation [10].

De infrarood sensors die bevestigd zijn aan het laadstation bakenen het aanmeergebied af. Onderstaande figuur maakt dit duidelijk. Hierbij is het aanmeergebied de witte driehoek. Bij het aanmeren zijn de infrarood sensors van de robot tijdelijk uitgeschakeld. Dit om interferentie te voorkomen.




Figuur 10: Aanmeer procedure, (a) De robot wordt gedetecteerd door één infrarood sensor.(b) De robot wordt gedetecteerd door allebei de infrarood sensoren.(c) De robot draait zich 90° zodat hij naar het laadstation gericht staat.(d)De robot connecteert zichzelf [10]

Voordelen

- De robot hoeft niet loodrecht aan te komen in het laadstation;
- relatief eenvoudige constructie;
- geen kans op kortsluiting.

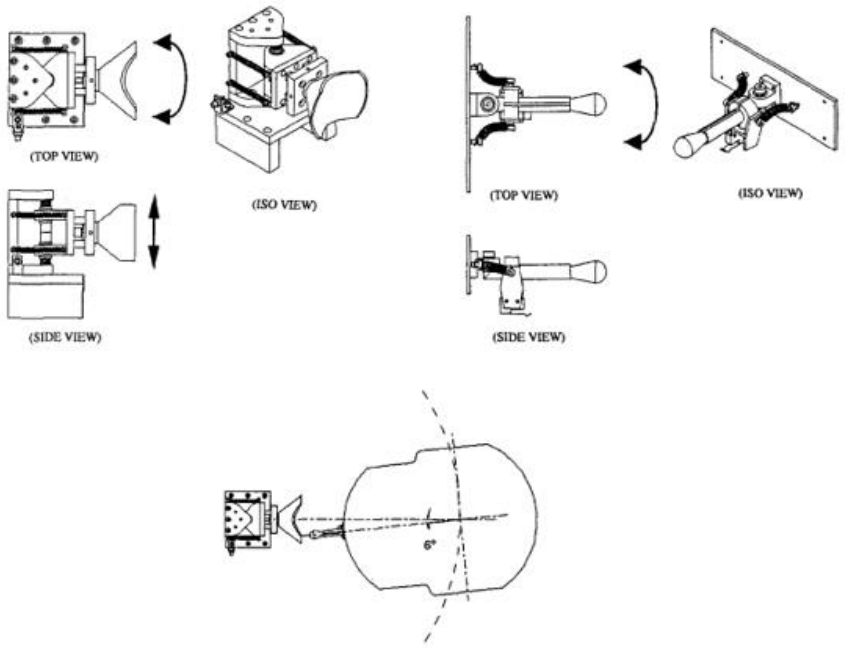
Nadelen

- De robot mag niet te ver rijden, zo kan het laadstation beschadigd worden, dit vraagt een grote precisie die met de elektrisch rolstoel moeilijk te bereiken zal zijn. Bij een kleine robot kan dit probleem nog opgelost worden door het laadstation lichtjes verend op te stellen;

- het bolvormig gedeelte aan de rolstoel zou veel plaats innemen, bij een te klein bolvormig gedeelte zou het aanmeren onder een hoek weer snel een probleem vormen.

#### 2.2.2.3 Nozzle principe

In dit ontwerp [12] zal er gebruik gemaakt worden van een nozzle of trechter aan de kant van het laadstation. Deze trechter kan enkele graden, zowel horizontaal als verticaal bewegen. Aan de robotzijde wordt er gebruik gemaakt van een soort oplaadpin, deze kan enkel in de horizontale richting bewegen. Door deze eigenschappen kan de robot onder een hoek van 6 graden aanmeren.




Figuur 11: Technische tekeningen Nozzle principe [12]

Voordelen

- De robot hoeft niet recht aan te komen bij het laadstation.

Nadelen

- Moeilijke constructie;
- veel bewegende delen;
- robot mag niet te ver rijden om beschadigingen aan het laadstation te vermijden, dit vraagt een grote precisie.

### 2.2.3 Besluit

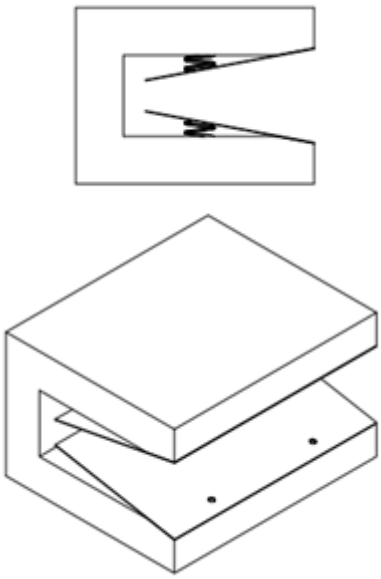
Aangezien de navigatie volledig via ROS zal gebeuren, zie volgende hoofdstukken, is er voor de ontwerp van het prototype enkel rekening gehouden met het laden en aanmeren. De navigatie binnen de kamer wordt hier niet meer besproken.

Aangezien de nadelen bij elk bestudeerd ontwerp te groot zijn leverde de literatuurstudie leverde geen ontwerp op dat rechtstreeks kon dienen voor dit project. Wel zijn er meerdere systemen gevonden die goede werkingsprincipes hadden en waar verschillende ideeën uit gehaald konden worden. Deze zijn de volgende:

- oplaadstrips boven elkaar, dit zodat er geen kortsluiting gemaakt kan worden bij scheef aanmeren;
- brede oplaadstrips zodat er een horizontale marge ontstaat;
- ervoor zorgen dat het laadstation niet beschadigd wordt als de robot er te diep inrijdt.

## 2.3 Ontwerp en bouw van eerste prototype

Aan de hand van de ideeën uit de literatuurstudie is er een laadstation ontwikkeld en uitgetekend met Autodesk Inventor. Hierbij zijn de oplaadstrips boven elkaar gepositioneerd. Dit kwam onder andere terug bij de robotstofzuigers maar is hier eerder uitgevoerd met het principe van de aluminium strips die langs elkaar gemonteerd waren.

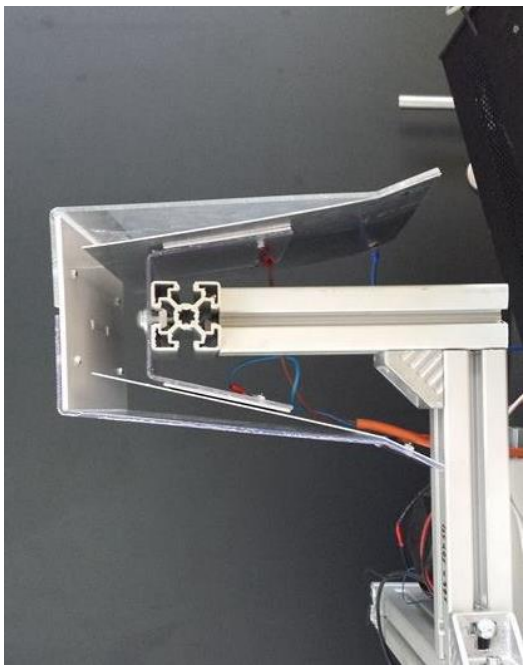


Figuur 12: Principetekening van een laadstation met oplaadstrips boven elkaar.

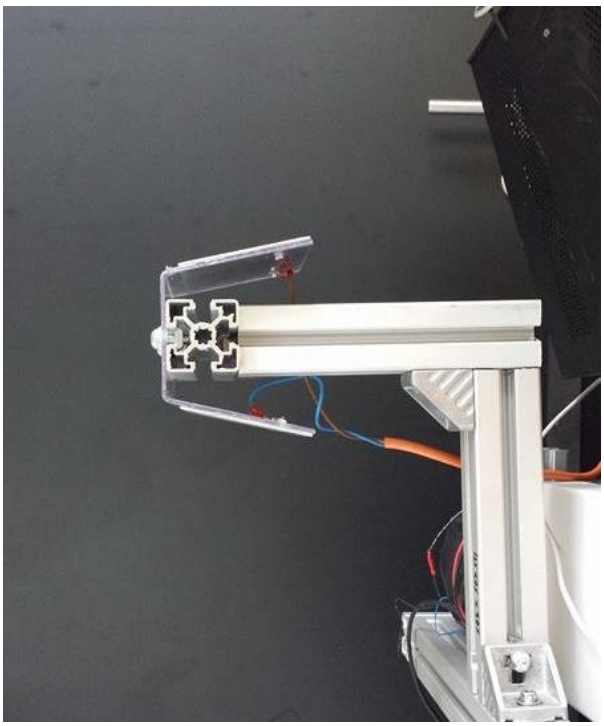
Na de ontwikkeling van het laadstation kon het geconstrueerd worden. De aluminium oplaadstrips zijn gemonteerd op een kunststof huis, dit zodat de oplaadstrips elektrisch gescheiden zijn, zie Figuur 13. Het kunststof huis is in feite een kunststof plaat die onder de

juiste hoeken geplooid is, voor de finale uitvoering kan dit huis in een stevig frame geplaatst worden te versteviging. Onderstaande afbeeldingen verduidelijken dit. Het laadstation wordt door middel van een connector verbonden met de bestaande oplader. Zoals vermeld is deze oplader behouden omdat de fabrikant dit type oplader voorschrijft voor dit type rolstoelen. Met dit ontwerp kan er dus ook nog steeds voor gekozen worden om manueel met deze lader te werken.

Aan de kant van de rolstoel komt 'het mannelijke gedeelte' van het laadstation, zie Figuur 14. Hiervoor wordt aan de achterkant van de rolstoel een houder uit kunststof voorzien. Deze is achteraan geplaatst omdat er onder de rolstoel weinig plaats is, omdat anders de wielen en de voetenplankjes in de weg zouden zitten bij het aanmeren. Op de kunststofhouder zijn aan de boven- en onderkant aluminium strips voorzien. Deze strips zullen contact maken met de oplaadstrips van het laadstation waarna de batterij wordt opgeladen. Op figuur wordt de werkingspositie van het laadstation zichtbaar. Het huis is hier manueel over 'het mannelijke gedeelte' geschoven om het principe duidelijk te maken. Het huis zal achteraf aan de muur of een stevig frame bevestigd worden.

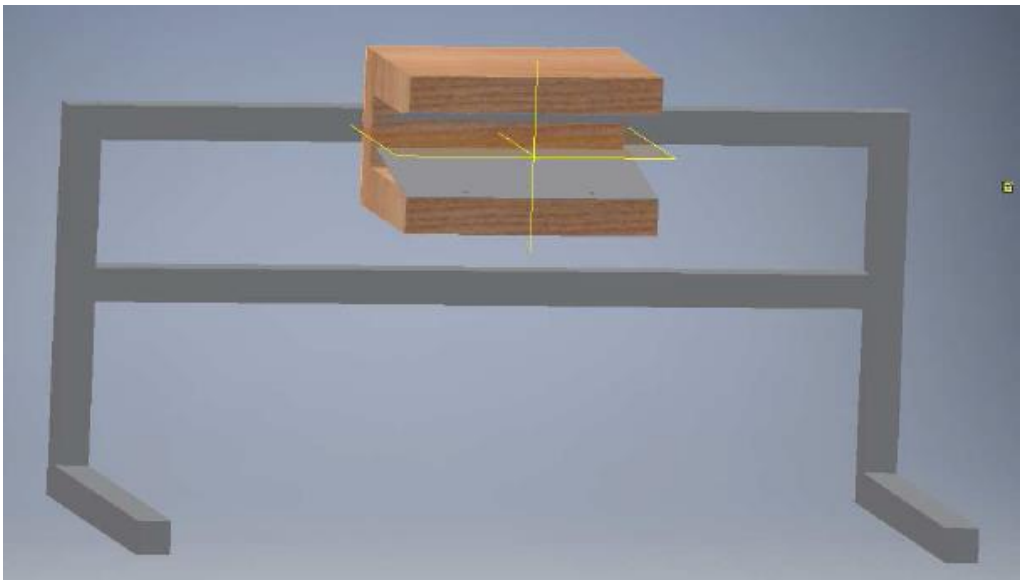


Figuur 13: Laadstation in werkingspositie.



Figuur 14: Laadsysteem rolstoelzijde.

Het onderstel zelf, waar het laadstation tegen komt is nog niet opgebouwd. Hiervoor zijn wel twee mogelijkheden voorzien, het laadstation kan gemonteerd worden tegen een muur, ofwel kan een aluminium frame het laadstation dragen. Ook zal nog een plaats voorzien worden voor de huidige lader in te plaatsen, een vrouwelijke XLR-connector zal aan het huis van het laadstation voorzien worden waar de huidige lader dan eenvoudig ingeplugd kan worden. Voor het aanmeeralgoritme te vergemakkelijken kan het frame ook voorzien worden van een duidelijke vorm die gebruikt kan worden als landmark, zie sectie 5.3 voor meer info.



Figuur 15: Principeschets prototype laadstation.

Eenmaal het laadstation is geconstrueerd kan het beoordeeld worden. Hiervoor wordt er terugverwezen naar de vereisten resultaten. Deze zijn terug te vinden in paragraaf 2.1 Ontwerpvereisten.

- Het laadstation is elektrisch veilig omdat het laadproces werkt op een veiligheidsspanning van 24V op de geleidingstrips;
- mechanisch kunnen er nog enkele verbeteringen aangebracht worden op gebied van veiligheid. Zo zouden de scherpe hoeken nog afgerond kunnen worden. Op mechanische veiligheid is niet expliciet gelet bij dit ontwerp omdat het nu slechts een prototype is;
- doordat er gekozen is voor brede en diepe geleidingstrips ontstaat er een horizontale en verticale marge die mogelijke onnauwkeurigheden van de rolstoel kan opvangen;
- aangezien het laadsysteem aan de rolstoelzijde beperkte afmetingen heeft (10x11x12cm) is het toepasbaar op verschillende types van rolstoelen. Het laadstation is in de deze vorm echter alleen maar toepasselijk voor opladers met een XLR-connectie. Het aanpassen naar een andere connector is slechts een kleine aanpassing;
- door te kiezen voor een aluminium onderstel is het laadstation makkelijk te integreren in elke omgeving;
- voorlopig houdt het systeem nog geen rekening met een slechte positionering van het laadstation in de omgeving, hier wordt wel op teruggekomen in sectie 5.3;
- er is momenteel nog geen kostenberekening gemaakt omdat het nog slechts een prototype is, in het ontwerp is wel rekening gehouden met de lage kostprijs van de gebruikte materialen;
- verdere testen moeten de levensduur van het laadstation nog bepalen. Dit was nog niet mogelijk omdat de rolstoel het pad nog niet uitvoert;
- het laadstation neemt relatief weinig plaats in, de uitstekende poten van het onderstel zijn noodzakelijk voor de stabiliteit, tenzij het laadstation rechtstreeks tegen de muur bevestigd word;
- er wordt nog geen sein gegeven wanneer de rolstoel is opgeladen;
- het pad naar het laadstation word nog niet uitgevoerd, mogelijke verplaatsingen van het laadstation zijn nog niet getest maar kleine verplaatsingen kunnen opgevangen worden via een flexibele programmatie, zie sectie 5.3 voor meer uitleg.

## 2.4 Experimentele resultaten

Om het correct opladen te garanderen is het belangrijk dat de spanningsval door implementatie van het laadstation niet te hoog is. Een te hoge spanningsval over het laadstation zou de oplaadcyclus negatief beïnvloeden. Om er zeker van te zijn dat dit niet het geval is, moesten er een aantal testmetingen op het ontworpen laadstation gebeuren. Hierbij werd de bronspanning, stroom en spanningsval over het laadstation gemeten. Deze metingen werden onder verschillende aanmeerhoeken uitgevoerd zodat er geconcludeerd kan worden wat de maximale aanmeerhoek is. Onderstaande tabel geeft de uitslagen van stroom- en spanningsmetingen. Hieruit werd daarna de contactweerstand berekend.

| Aanmeerhoek | 0° | 15° | 30° | 45° | 60° |
|---|---|---|---|---|---|
| Bronspanning (V) | 28,70 | 28,80 | 28,56 | 28,62 | 28,68 |
| Stroomsterkte (A) | 1,50 | 1,50 | 1,50 | 1,50 | 1,40 |
| Spanningsval over het laadstation (mV) | 45,6 | 62,3 | 60,0 | 40,0 | 170,0 |
| Weerstand van het laadstation (Ω) | 0,0304 | 0,041533 | 0,04 | 0,026667 | 0,121429 |

Tabel 1: Resultaten van de metingen voor het bepalen van de contactweerstand en de spanningsval bij gebruik van het laadstation.

De testen tonen aan dat er slechts een kleine spanningsval is over het laadstation bij aanmeerhoeken tussen de 0° en de 45°. Bij aanmeerhoeken groter dan 45° is de spanningsval iets groter, maar nog altijd niet te hoog. Door implementatie van het laadstation zal er dus geen negatieve invloed zijn op het laadproces van de batterijen.

## 2.5 Besluit

De literatuurstudie naar zowel commerciële als niet commerciële laadstations hebben geen concreet ontwerp naar voren gebracht, maar hebben wel geleid tot enkele conceptueel zeer goede ideeën voor het ontwerp van het laadstation. Aan de hand van deze ideeën is een laadstation ontworpen en getekend met behulp van Autodesk Inventor. De constructie gebeurde aan de hand van een polycarbonaten plaat die onder de juiste hoeken geplooid werd. Hierop zijn de oplaadstrips gemonteerd. Het gebruik van het laadstation mag natuurlijk geen negatieve invloed hebben op het laadproces van de batterijen. Om er zeker van te zijn dat dit niet gebeurt zijn er enkele testen uitgevoerd. Hierbij is vooral gekeken naar de spanningsval over het laadstation. Deze lag bij hoeken onder de 45° rond de 60mV. Hieruit kon dan ook besloten worden dat er zich nauwelijks negatieve invloeden op het laadproces voordoen. Wanneer er duidelijkheid is over de precisie van het autonoom aanmeren kan de finale versie van het laadstation opgebouwd worden. Hoofdstukken 4 en 5 analyseren deze navigatienauwkeurigheid verder.

# 3 Het gebruikte voertuigplatform en sensoren

## 3.1 Elektrische rolstoel

Het basis mobiel platform is een commercieel beschikbare elektrische rolstoel van het merk Invacare, het type is typhoon. Onderstaande afbeelding toont de rolstoel. Deze werd al gebruikt bij eerder onderzoek aan onderzoeksgroepen van de KU Leuven (ACRO en PMA) [13]. Veel hardware was dus al voorzien. Zo hangt er achteraan de rolstoel een computer die voor programmatie gebruikt wordt, de wielen waren voorzien van encoders en tevens was er een laserscanner beschikbaar. De belangrijkste onderdelen worden hieronder nog verder toegelicht.

Figuur 16: De rolstoel die gebruikt werd voor dit project.

## 3.2 Laserscanner

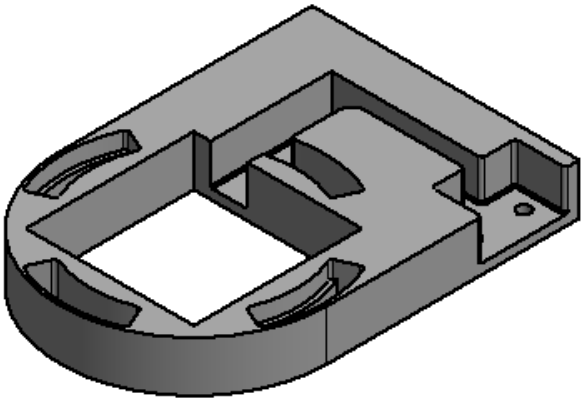
Eén van de belangrijkste componenten van deze masterproef is de laserscanner. De gebruikte laserscanner is een Hokuyo URG-04LX. Dit is een 2D laserscanner met een meethoek van 240°. De maximale afstand bedraagt vier meter. Hiermee is deze laserscanner dus zeer goed geschikt voor binnenshuis. De datasheet is terug te vinden in de bijlage achteraan.


Figuur 17: Hokuyo URG-04LX laserscanner [14].

In vorige onderzoeken met deze rolstoel werd gebruik gemaakt van twee laserscanners. Er werd dan zowel voor- als achteraan een laserscanner voorzien. Voor deze masterproef was echter nog maar één laserscanner beschikbaar. Aangezien het laadsysteem achteraan de rolstoel wordt bevestigd is het logisch om ook de laserscanner achteraan te bevestigen. Zo zal het aanmeermaneuvre preciezer en betrouwbaarder kunnen gebeuren. Om praktische redenen werd ervoor gekozen de laserscanner ondersteboven te monteren, dit om meer plaats te laten tussen de laserscanner en het laadsysteem.

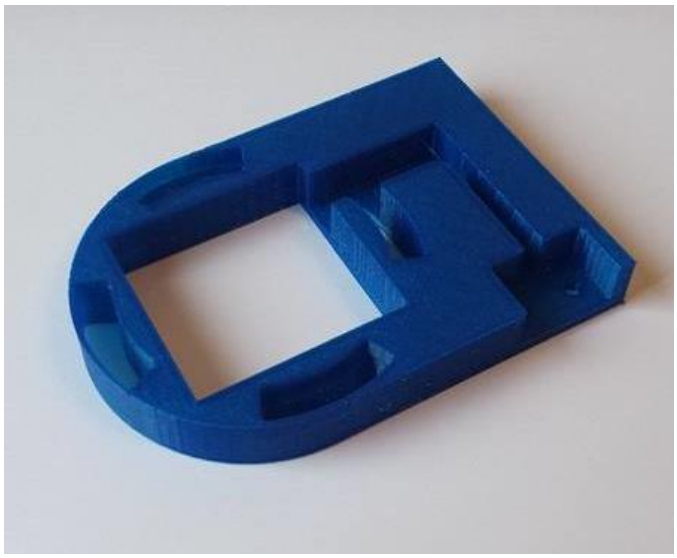
Figuur 18: De laserscanner werd ondersteboven gemonteerd aan de achterzijde van de rolstoel.

De reden dat er nog slechts één laserscanner beschikbaar is, is dat bij vorig onderzoek de USB-connector van de andere laserscanner verschillende keren beschadigd werd, en deze is moeilijk te herstellen. De connector bij deze Hokuyo scanners is zeer fragiel uitgevoerd waardoor bij beweging van de USB-stekker snel slecht contact ontstaat wat natuurlijk nefast is voor een vlotte werking. Om dit probleem op te lossen werd een versteviging ontworpen in Autodesk Inventor en 3D-geprint. De figuur hieronder geeft het ontwerp van deze versteviging weer. De scanner zal in het vierkant gat komen. De USB-connector en de kabel komen in de uitsparing te liggen. Er zijn twee gaatjes aan de rechterzijde voorzien zodat hier de kabel geklemd kan worden met een trekbandje, dit zodat er nooit meer aan de connector zelf getrokken kan worden.



Figuur 19: Inventor tekening van het beschermblokje dat ontworpen werd ter beveiliging van de laserscanner.

Onderstaande afbeelding toont het werkelijke beschermblokje. In Figuur 18 is de bevestiging van het beschermblokje al zichtbaar.



Figuur 20: 3D geprint beschermblokje dat ontworpen werd ter beveiliging van de laserscanner..

Voor de programmatie is het belangrijk om de positie van de laserscanner ten opzichte van het rolstoelframe te kennen. Hiervoor ontwikkelde Mr. E. Demeester in het verleden een MATLAB applicatie. Door de rolstoel loodrecht te positioneren t.o.v. een rechte hoek en de afstand in lengte- en breedterichting te meten kan exact de juiste positie bepaald worden.

## 3.3 Encoders

Om de afgelegde weg van de aangedreven wielen van de robot te kennen zijn er encoders voorzien; de stand van de vier zwenkwielen wordt niet opgemeten. De encoders bevinden zich op de assen van de wielen. Dit zijn magnetische encoders van het merk Baumer met typenaam MDFK 10. Hier is in het verleden door de vorige projectmedewerkers voor gekozen. Het zijn encoders van het magnetoresistive sensor type. Deze werkt op het principe van

elektrische weerstand van het meetelement dat typisch verandert afhankelijk van het aangelegde magnetisch veld.


Figuur 21: Magneetencoder MDFK 10 [13]

## 3.4 Laptop

De computer die bevestigd is op de rolstoel draait nog op Ubuntu 10.04. Hier komen, via NIDAQ data-acquisitiekaarten, de encoderdata binnen. Deze data worden ingelezen en verwerkt met software in C++, waarna de hoekstanden van de wielen bekend zijn. De oorspronkelijke bedoeling was om de computer te updaten naar Ubuntu 14.04, zo kon nog gebruik gemaakt worden van de vroegere programmering in Eclipse en kon de ook de kaartopbouw, lokalisatie en padplanning ook rechtstreeks vanaf deze computer gebeuren. Wegens een te oude Ubuntu versie was updaten echter niet meer mogelijk. Een andere optie was een volledige herinstallatie van Ubuntu 14.04 met als nadeel dat achteraf in Eclipse alle juiste bibliotheken weer gezocht en geïnstalleerd moesten worden, wat een zeer tijdrovend proces kon worden om alles opnieuw op punt te krijgen.

Als alternatief is er daarom voor gekozen het besturingssysteem zo te laten en de nodige encoderdata vanuit Eclipse via een ethernet socket te versturen naar een laptop die wel draait op Ubuntu 14.04. Dit heeft als bijkomend voordeel dat de opstelling flexibeler is naar programmatie toe aangezien er zo niet telkens een scherm aan de computer van de rolstoel gekoppeld dient te worden.

### 3.4.1 Socket

Zoals in vorige paragraaf al beschreven staat moet er gebruik gemaakt worden van een socketverbinding tussen een laptop waarop alle navigatiesoftware draait en de rolstoelPC die de encoderdata inleest. Deze verbinding bestaat uit twee delen, een server en een cliënt. De server zal wachten op een verzoek van de cliënt. Om het verzoek te kunnen doen, moet er aan de server een IP adres gegeven worden zodat de cliënt de server kan vinden. Als het verzoek gedaan is zal de server dit verwerken en zal een antwoord op het verzoek terug sturen naar de cliënt. Deze weet zo dat er al dan niet een goede verbinding is gemaakt. Eénmaal er verbinding is kunnen er data in twee richtingen doorgestuurd worden.

Er bestaan vier soorten sockets: stream, datagram, raw en sequenced packet sockets. De eerste twee worden meestal gebruikt, de laatste twee daarentegen worden zelden tot nooit toegepast. Van daardat alleen de stream en datagram sockets hier kort toegelicht worden.

- Stream sockets: bij dit type wordt de aflevering van de data gegarandeerd. Ook zal de volgorde van de data hetzelfde blijven. Als er bijvoorbeeld data van het volgende formaat gestuurd zouden worden, 'A,B,C', dan zal de ontvanger die data ook in deze volgorde binnen krijgen. Als de data overdracht onmogelijk is, dan zal de afzender een error terug krijgen en dus weten dat de data niet correct verzonden is. Deze sockets maken gebruik van TCP (Transmission Control Protocol) voor data overdracht.

- Datagram sockets: hierbij wordt de aflevering van de data, in tegenstelling tot de stream sockets niet gegarandeerd. De afzender stuurt in dit geval zijn data 'blind' op de verbindingslijn zonder er rekening mee te houden of de ontvanger ook effectief de data binnen krijgt. Er is dus minder zekerheid, maar net doordat er geen controles moeten uitgevoerd worden kan de verzender veel meer data sturen op dezelfde tijd. Datagram sockets maken gebruik van UDP (User Datagram Protocol).

Omdat er in dit geval zeer veel data doorgestuurd moet worden, en dit liefst ook nog zo snel mogelijk, is er gekozen voor een UDP socket. Bovendien is de kans dat er data verloren zal gaan zeer klein bij een rechtstreekse verbinding tussen twee computers.

## 3.5 Besluit

Dit hoofdstuk beschreef kort de relevante onderdelen van het gebruikte rolstoelplatform en de hierop aanwezige sensoren (laserscanner en encoders).

De USB-connector van de laserscanner is erg kwetsbaar. Om in de toekomst te vermijden dat deze USB-connector afbreekt, is er een beschermblokje ontworpen en geproduceerd. Hiervoor is een 3D-printer gebruikt. Ook is de nodige hardware uitgebreid met een extra laptop. Dit was nodig omdat de PC, die verbonden is met de rolstoel, nog op Ubuntu 10.04 werkt. De gebruikte robotsoftwarebibliotheek ROS (zie Hoofdstuk 0) is hiermee niet compatibel. Omdat Ubuntu 10.04 niet meer ondersteund wordt, kan deze ook niet worden geüpdatet. Hierdoor is er een extra laptop toegevoegd waarop ROS wel compatibel is. Voor de communicatie tussen de laptop en de PC is er gekozen voor een UDP verbinding, dit omdat deze zeer veel data op korte tijd doorgeeft. Via deze verbinding zullen enerzijds de encoderdata van de wielen worden doorgestuurd naar de laptop. Anderzijds zullen de benodigde lineaire- en hoeksnelheid voor padplanning worden teruggestuurd van de laptop naar de PC.

# 4 Kaartopbouw en lokalisatie

Zoals al besproken werd in de literatuurstudie bestaan er verschillende manieren voor een robot om het laadstation terug te vinden. De snelste en meest effectieve manier is om op voorhand een kaart op te bouwen van de omgeving waarin de robot zal functioneren. Op het moment dat de robot zich naar het laadstation moet begeven moet deze zich enkel kunnen lokaliseren binnen deze kaart waarna rechtstreeks het beste pad naar het laadstation gepland kan worden. Binnen dit project is er dus voor gekozen om geen gebruik te maken van infraroodsignalen, of andere detectiesystemen, om het laadstation terug te vinden. Het laadstation zal standaard op dezelfde plaats in de kamer moeten staan, deze plaats zal dan bij eerste ingebruikname worden meegegeven aan het padplanningsalgoritme.

Binnen dit hoofdstuk zal in sectie 4.1 eerst de gebruikte software besproken worden. Daarna volgt in sectie 4.2. de uitleg over de kaartopbouw. Eerst wordt hier algemeen de werking van SLAM uitgelegd waarna de twee geteste algoritmes, Hector Slam en Gmapping, uitgebreid besproken en geëvalueerd worden. In sectie 4.3 volgt de werking van het gebruikte algoritme voor de lokalisatie binnen een vooraf opgebouwde kaart, AMCL. Ook de nauwkeurigheid van de lokalisatie wordt hier besproken waarna in sectie 4.4 een algemeen besluit over dit hoofdstuk volgt.

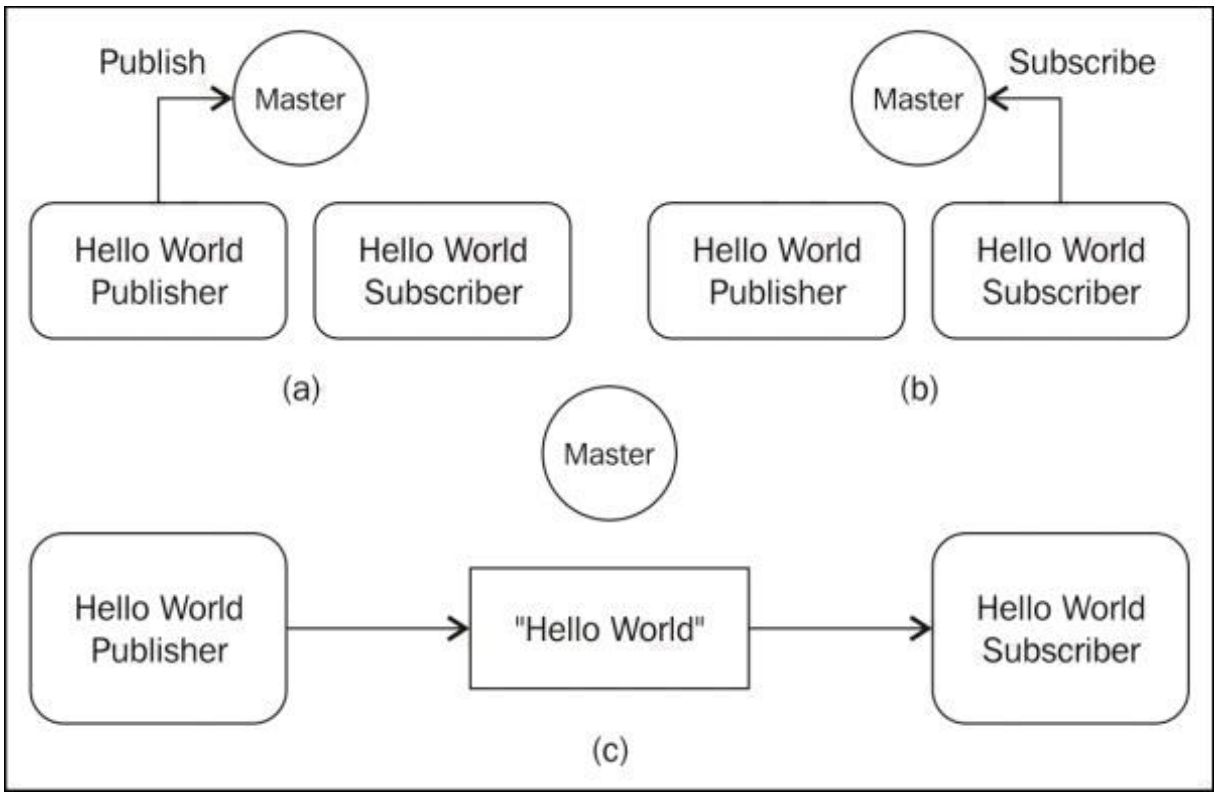
## 4.1 Gebruikte software

Voor de programmatie van deze masterproef is gekozen voor ROS, 'Robot Operating System'. Dit is een verzameling van softwarebibliotheken met nuttige functionaliteit voor het ontwikkelen van robots, zoals drivers om sensoren in te lezen of algoritmes voor padplanning. Hierin bestaat zeer veel open source software die geschikt is voor projecten als deze. De programmering binnen ROS kan geschreven worden in zowel C++ als Python. Binnen deze masterproef werd wegens eerdere kennismaking met C++ ook hiervoor gekozen.

Deze softwarebibliotheken worden enkel ondersteund voor Ubuntu Linux. Experimenteel wordt het echter ook al gebruikt met zowel Microsoft Windows als Mac OS X. Voor de hand ligt dus wel dat bij deze programmatie gebruik werd gemaakt van Ubuntu. De gebruikte versie is Ubuntu 14.04 waarbij gekozen is voor ROS Indigo Igloo, deze release van ROS wordt aangeraden omwille van zijn stabiliteit en is ook ontwikkeld voor Ubuntu 14.04. Zoals al eerder vermeld zal dit op een aparte laptop gebeuren, en dus niet op de PC die verbonden is met de rolstoel.

Om de werking van de verschillende algoritmes die gebruikt zijn binnen deze masterproef te kunnen begrijpen is het belangrijk om de basisprincipes van de programmering binnen ROS te kennen. Binnen deze masterproef gebeurde de programmering in C++. Wanneer deze C++-files gecompileerd worden tot ROS nodes, kunnen deze gestart worden binnen ROS, na het starten van de ROS Master, die de communicatie tussen de nodes verzorgt.

Onderstaande figuur beschrijft visueel de werking van de communicatie via ROS nodes. Op moment (a) begint een publisher met het zenden van data op een communicatiekanaal. Binnen ROS wordt gesproken over 'topics' in dit geval het topic 'Hello World'. Op dit moment krijgt de master de details over het topic en de node. De master zal zoeken naar andere nodes die geabonneerd zijn op dit topic. Op moment (b) start de subscriber node, deze wil zich abonneren op het topic "Hello World". De master krijgt ook deze details en zal nu (c) de communicatie regelen tussen deze twee nodes. Vanaf nu verloopt de communicatie automatisch zonder tussenkomst van de master. In dit voorbeeld wordt de string "Hello World" gepubliceerd op en gelezen van het topic.




Figuur 22: Communicatie tussen publisher en subscriber in ROS [15].

Het publiceren op een topic gebeurt niet lukraak maar altijd via een message waarvan de opbouw van de gepubliceerde data precies gekend is. Voor veelgebruikte toepassingen bestaan standaard messages. Deze werden in deze masterproef bijvoorbeeld gebruikt voor het publiceren van de odometriedata.

ROS nodes kunnen via het terminal venster apart gestart worden. Makkelijker is echter om gebruik te maken van een launch file. Hiermee kunnen voor de verschillende toepassingen meerdere nodes samen gestart worden om het gebruiksgemak te verhogen. Ook kunnen hier meteen parameters worden ingesteld.

De gebruikte algortimes zullen uitgelegd worden aan de hand van een RQT-graph. Om ook dit principe even te verklaren geeft de onderstaande figuur de RQT-graph weer van het bovenstaande voorbeeld.


Figuur 23: Hello world publisher en subscriber.

De grotere rechthoeken met daarin de ovalen visualiseren de verschillende nodes. Zo is links de Hello World publisher te zien en rechts de Hello World subscriber. De kleinere rechthoeken geven de topics weer. Een pijl naar een topic betekent dat er data op dit topic wordt gepubliceerd. Een pijl die vertrekt vanaf een topic betekent dat er data vanaf dit topic wordt gelezen. RQT-graphs zijn dus zeer goed geschikt om op een visuele manier de communicatie tussen de verschillende nodes weer te geven. [15]

## 4.2 Kaartopbouw

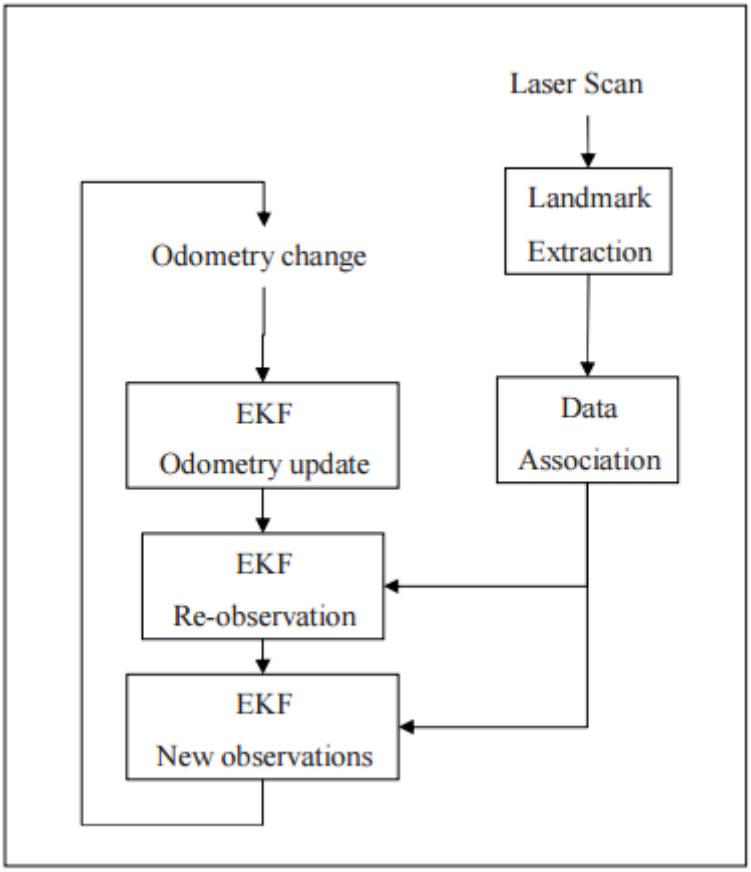
### 4.2.1 Algemene werking

Wanneer een mobiele robot zich in een omgeving van punt A naar punt B moet verplaatsen zijn er verschillende vragen die de robot zich zal moeten stellen. De eerste duidelijke vraag is: "Hoe ziet de wereld eruit?". De tweede belangrijke vraag is: "Waar bevind ik mij in deze wereld?". Om deze problemen op de lossen wordt in de praktijk gebruikt gemaakt van een kaartopbouw- en lokalisatie algoritme. Voor kaartopbouw, in het Engels "Simultaneous Localisation And Mapping (SLAM)" genoemd, bestaan verschillende algoritmen en software. Hiervoor gebruikt de robot één of meerdere sensoren, . meestal 2D-laserscanners. Omdat deze laserscanners nooit perfect zijn, men krijgt te maken met zowel ruis als meetfouten, zal dit niet altijd even gemakkelijk zijn. Ook bij metingen van glazen oppervlaken zullen grote meetfouten optreden. Wanneer de kaart van de omgeving wordt opgebouwd zal de robot zichzelf hierin tegelijkertijd ook moeten kunnen lokaliseren. Hiervoor wordt meestal naast de laserscandata ook gebruik gemaakt van odometriedata. Aan de hand van de rotatie van de wielen, die gemeten wordt met wielencoders, wordt de verplaatsing van de robot berekend. Ook deze berekeningen zullen echter enkel een benadering van de precieze verplaatsing van de robot geven, onder andere het slippen van de wielen kan snel grote fouten geven als het algoritme niet optimaal is ingesteld. De afwijking op de odometrie is moeilijk te voorspellen en zal dan ook groter en groter worden wanneer deze niet gecorrigeerd wordt. Het SLAM-algoritme is in staat de sensordata en odometriedata samen te verwerken om een kaart op te bouwen die zo precies mogelijk is. [16]

### 4.2.2 SLAM proces

Het SLAM proces bestaat uit een aantal stappen. Het doel ervan is om de positie van de robot te achterhalen aan de hand van de omgeving. Om de positie te achterhalen zal het SLAM proces eerst enkel de odometrie gebruiken. Aangezien dat er vaak een fout op de odometrie zit, bijvoorbeeld door het slippen van de wielen, is het niet voldoende om hier alleen rekening mee te houden. Een laserscanner lost dit probleem op, deze zal namelijk referentiepunten in

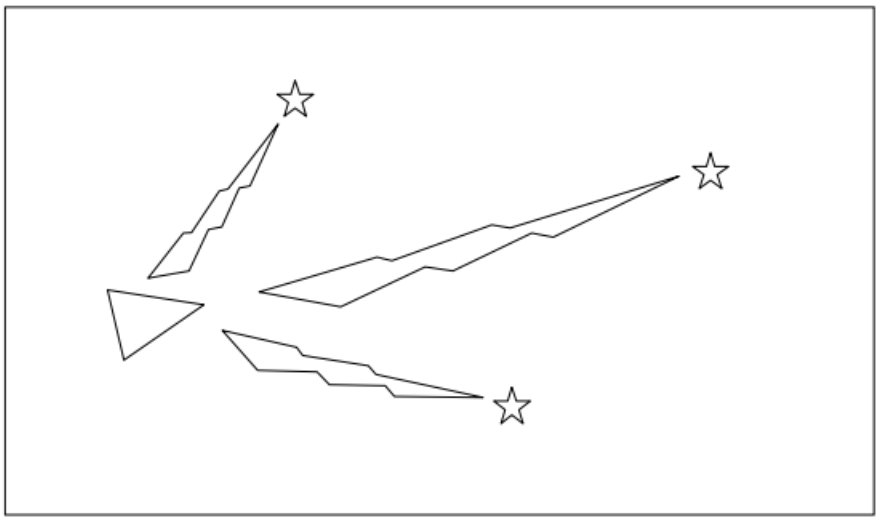
de omgeving binnen lezen. Als de robot zich vervolgens verplaatst zal er opnieuw naar deze referentiepunten gekeken worden. Deze referentiepunten worden landmarks genoemd. Een EKF (Extended Kalman Filter) verwerkt de data van de laserscanners. De EKF is verantwoordelijk voor het updaten van de robotpositie. Deze zal rekening houden met een schatting van de onzekerheid van de robot positie, maar ook van de onzekerheid van de landmarks in de omgeving. Onderstaande afbeelding geeft de algemene structuur van het SLAM proces.




Figuur 24: Algemene structuur SLAM proces [17].

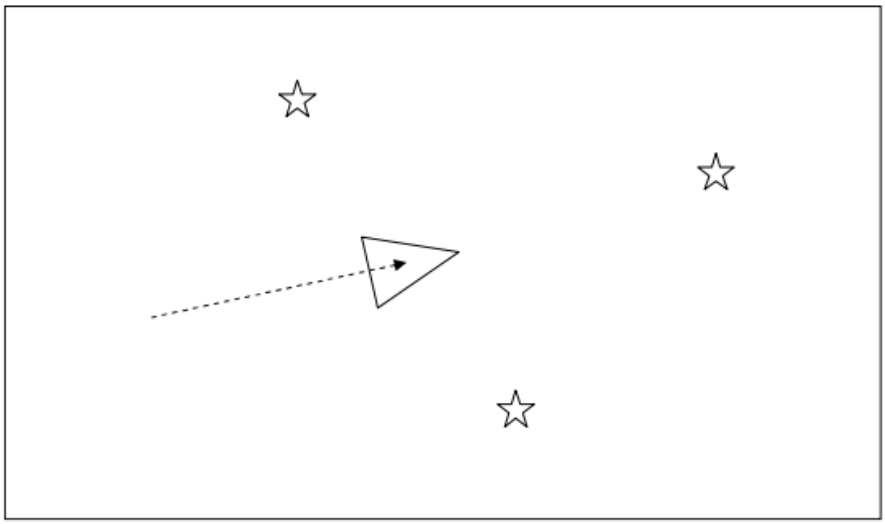
Als de robot van positie verandert zal de odometrie natuurlijk ook wijzigen. De onzekerheid van de nieuwe robot positie zal vervolgens worden geüpdatet in de EKF aan de hand van deze odometriedata. Dan kijkt het SLAM proces vanuit zijn nieuwe positie naar landmarks. Er wordt geprobeerd om deze te koppelen aan vorige landmarks, als dit gelukt is kan vervolgens de nieuwe positie van de robot bepaald worden in de EKF, deze stap noemt re-observation. Als er landmarks gezien worden die nog niet eerder geobserveerd zijn zullen deze toegevoegd worden aan de EKF. Deze kunnen later dan gebruikt worden voor re-observation. Hieronder volgt een verdere uitleg van het SLAM proces aan de hand van afbeeldingen.

De driehoek in onderstaande afbeelding stelt de robot voor en de sterren stellen landmarks voor. Dit is de eerste stap van het SLAM proces. De robot zal om te beginnen een eerste keer via zijn sensoren naar landmarks kijken en deze opslaan.



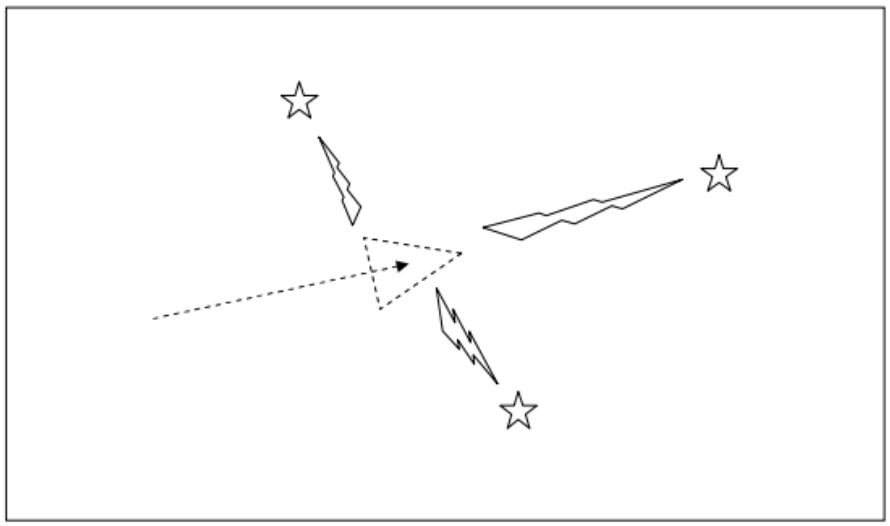
Figuur 25: Eerste observatie van landmarks [17].

Vervolgens zal de robot naar een nieuwe positie rijden. De gestreepte lijn geeft deze beweging aan. De verplaatsing, en dus ook de verwachte positie wordt hier nog enkel berekend door de odometrie.



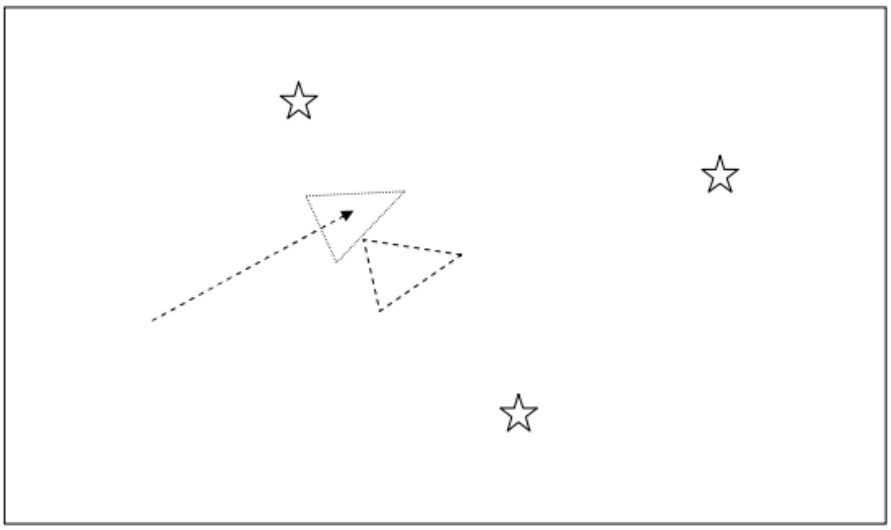
Figuur 26: Beweging van robot naar nieuwe positie [17].

Op de nieuwe positie voorspelt de robot de locaties van de in de eerste stap geobserveerde landmarks. Deze voorspelde locaties zullen meestal echter niet exact overeen komen met de werkelijke landmarklocaties, dit komt omdat er vaak een fout op de odometrie zit. De robot is dus niet op de positie waar hij denkt dat hij is, deze wordt nog altijd weergegeven in onderstaand afbeelding door de gestreepte lijn. Vervolgens neemt de robot opnieuw de landmarks waar met de laserscanner vanuit zijn positie.



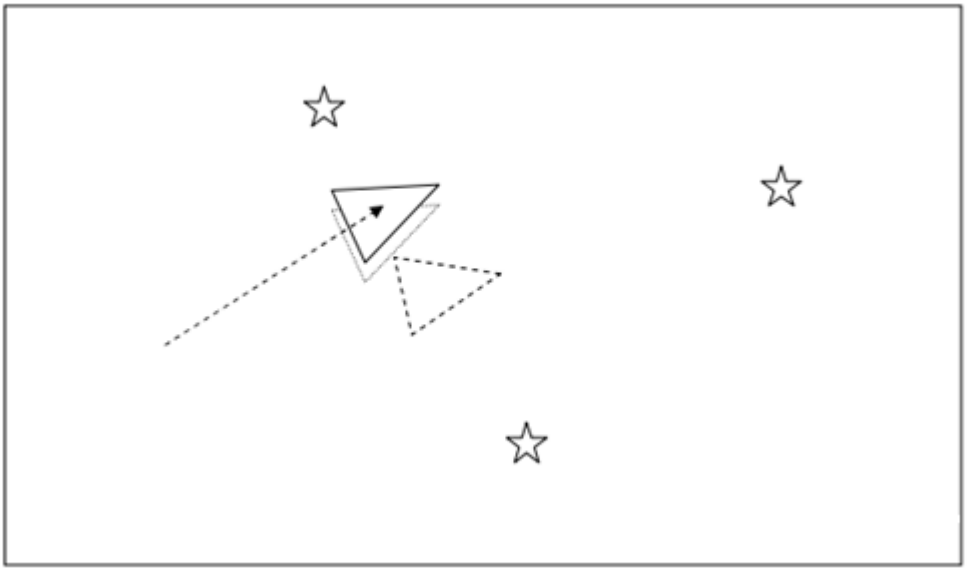
Figuur 27: Observatie van de landmarks vanuit nieuwe robot positie [17].

Aangezien de robot zijn laserscans meer vertrouwt dan de odometrie data, zal de EKF de veronderstelde positie aanpassen. Dit is de re-observation stap waarbij de huidige landmarks vergeleken worden met de vorige.



Figuur 28: Robotpositie via odometrie en anderzijds via laserscans [17].

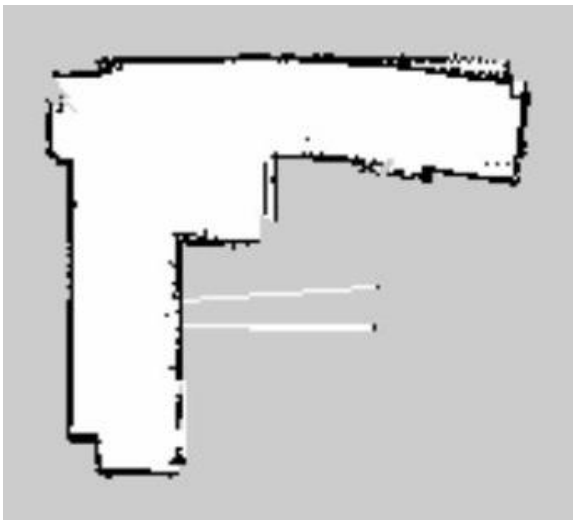
De veronderstelling die de robot maakt van de positie komt nooit perfect overeen met de werkelijkheid. Dit komt omdat er altijd nog wel ergens kleine afwijkingen zijn. Onderstaande afbeelding geeft dit weer. Hierbij geeft de driehoek met volle lijnen de werkelijke (maar ongekende) positie van de robot, de grijze lijn de geschatte positie aan de hand van de laserscans en de odometrie. De gestreepte lijn geeft de positie die bekomen wordt als er enkel gebruik gemaakt word van de odometrie. [17]



Figuur 29: Overzicht van werkelijke en geschatte robotposities [17].

### 4.2.3 Kaartopbouw zonder odometrie: Hector SLAM

Om kennis te maken met de verschillende SLAM-algoritmes binnen ROS, werd de eerste test uitgevoerd met Hector SLAM. Dit is een algoritme dat enkel gebruik maakt van een laserscanner zonder dat hiervoor odometrie nodig is. Dit algoritme kon dus al getest worden nog voor de communicatie met de computer in orde was en de odometriedata beschikbaar waren. Door de laserscanner eenvoudig te koppelen aan een laptop kon hiermee al een kaart gemaakt worden. Voor de eerste testen werd een kaart opgebouwd met de sensor in de hand. Om de testen daarna secuur te laten verlopen werd de sensor natuurlijk wel al bevestigd op de rolstoel. De testen gebeurden in de trappenhal gelegen achter ACRO. De figuur hieronder geeft het resultaat hiervan weer.

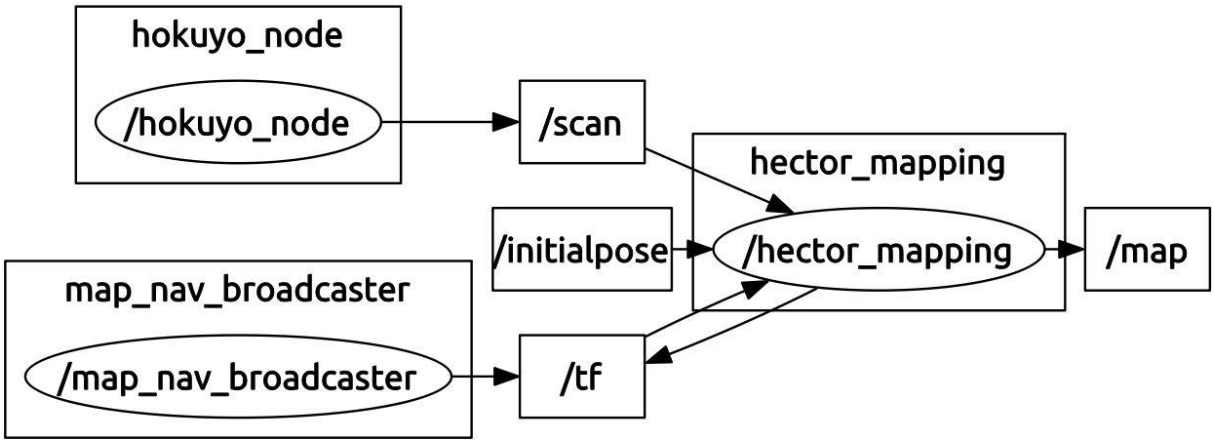


Figuur 30: Kaart trappenhal a.d.h.v. Hector SLAM.

Voor eenvoudige ruimtes zoals deze trappenhal konden al snel goede resultaten behaald worden. Er moest echter zeer traag gereden worden. Bij snelle verplaatsing raakt het algoritme snel zijn locatie kwijt waarna de kaartopbouw stopt of grove fouten maakt. Zo is duidelijk te zien dat bij het iets te snel draaien rechtsboven in de kaart de gang scheef wordt afgebeeld, deze gang is in werkelijkheid wel recht. Omdat er binnen dit algoritme geen controle over de locatie mogelijk is met behulp van de odometrie is het mogelijk dat het algoritme niet exact in staat is de huidige locatie van de scanner te bepalen. Ook bij lange rechte muren of glazen wanden treden er al snel problemen op. Dit bleek duidelijk wanneer met Hector SLAM getest werd in grotere ruimtes. Een laserscanner met een grotere range zou dit probleem al kunnen verminderen, maar de grootste verbeterfactor is natuurlijk het kiezen voor een algoritme dat wel gebruik maakt van odometrie.

#### 4.2.3.1 RQT-graph Hector slam

Onderstaande figuur geeft de RQT-graph van Hector mapping weer. De belangrijkste node hierin is natuurlijk de node Hector_mapping. Hierin gebeurt het rekenwerk en wordt gezorgd voor de opbouw van de kaart. Hector_mapping is geabonneerd op verschillende topics. Zo is er het topic /scan, op dit topic wordt geschreven door de Hokuyo node, dit is de publisher die de data komende van de laserscanner verwerkt. Daarnaast is er nog het zeer belangrijke topic /tf. Het topic /tf houdt de plaats en de verplaatsing van de verschillende assenstelsels bij. Zo onderscheiden we binnen deze masterproef onder andere het basisassenstelsel van de rolstoel alsook het assenstelsel van de laserscanner. Indien de plaats van de laserscanner verschilt van het basisassenstelsel van de rolstoel moet dit dus ook nog meegegeven worden. Dit gebeurt via de node map_nav_broadcaster. Duidelijk te zien is dat Hector_mapping zowel publiceert als geabonneerd is op het topic /tf. Dit omdat Hector_mapping natuurlijk telkens eerst de vorige locatie moet kennen en achteraf ook de nieuwe locatie moet kunnen meegeven. Binnen hector_mapping bestaat ook de mogelijkheid om de startlocatie binnen de kaart mee te geven. Dit is niet noodzakelijk, maar indien gewenst kan deze initiële pose worden meegeven via het topic /initial_pose. Hector_mapping zal, aan de hand van al deze data, in real time de kaart opbouwen en deze data publiceren op het topic /map. Deze map kan dan gemakkelijk via het programma Rviz gevisualiseerd worden.

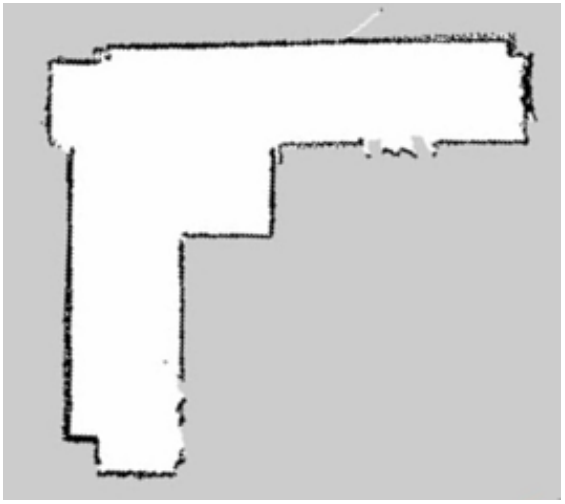



Figuur 31: RQT-graph Hector slam.

### 4.2.4 Kaartopbouw met odometrie: Gmapping

Het meest gebruikte algoritme binnen ROS is Gmapping, dit is een SLAM algoritme dat gebruik maakt van zowel sensordata als odometriedata. De sensordata konden met behulp van de Hokuyo node rechtstreeks gebruikt worden. Voor de odometriedata moest in deze masterproef eerst een node worden ontwikkeld die precies de juiste informatie doorgeeft aan Gmapping. Binnen deze node gebeuren eerst de instellingen voor de UDP socket. Vanaf het starten van de node zullen de encoderdata, komende van de PC aan de rolstoel binnenkomen. Deze data bestaan telkens uit een stringstream waar naast een ID en timestamp de huidige hoekpositie van de twee aangedreven wielen worden meegegeven. Aan de hand van deze posities wordt de lineaire en de hoeksnelheid van de rolstoel berekend. Daarna kan de verplaatsing in x- en y-richting berekend worden alsook de oriëntatie van de rolstoel. Gmapping zal deze data zoeken op het topic /tf volgens het juiste protocol. De data worden dus in de juiste vorm gepubliceerd op dit topic. Het volledige bestand, geschreven in C++ en voorzien van commentaar, is terug te vinden in bijlage 1.

De eerste testen met Gmapping gaven meteen een behoorlijk resultaat. Wanneer het algoritme nog extra werd geoptimaliseerd werd het resultaat opmerkelijk beter. Op de onderstaande figuur is de opgebouwde kaart te zien van de trappenhal binnen het onderzoekscentrum. Dit is duidelijk een veel betere kaart dan de kaart die bekomen werd met Hector SLAM.



Figuur 32: Kaart trappenhal a.d.h.v. Gmapping.

#### 4.2.4.1 Evaluatie

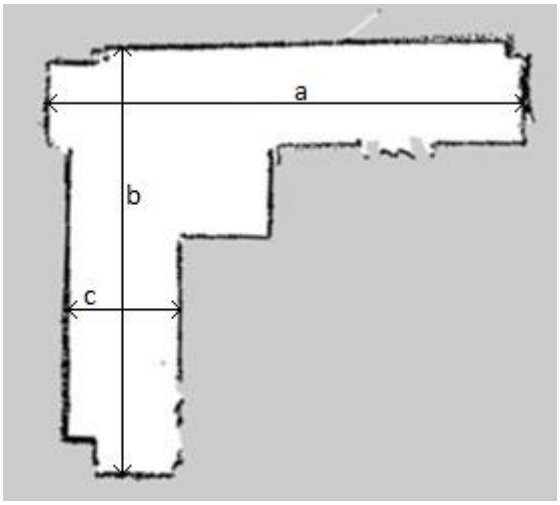
Voor een hoge nauwkeurigheid bij het uitvoeren van het geplande pad en de lokalisatie binnen een kaart is het natuurlijk belangrijk om te beginnen met een kaart die zelf erg nauwkeurig is. Tijdens het testen werden al enkele parameters van het algoritme aangepast. De parameter die hier de grootste verbetering gaf was '~iterations'. Deze parameter bepaalt het aantal iteraties van de scanmatcher. Standaard stond deze ingesteld op 5. Een verhoging naar 20 gaf al een aanzienlijk beter resultaat. Om de kaartopbouw nog beter te optimaliseren is hiervoor zeker

nog ruimte bij het veranderen van verscheidene parameters. Dit is een proces dat bestaat uit telkens aanpassen en testen tot de optimale parametrering bereikt is.

Om de juistheid van de opgebouwde kaart te controleren kunnen via RVIZ met de tool 'measure' de verschillende afstanden gemeten worden. Deze gemeten afstanden worden in de tabel hieronder vergeleken met de werkelijke afstanden. De figuur eronder geeft duidelijkheid over de plaats van de metingen. Te zien is dat het verschil met de werkelijkheid maximaal 2 cm is. De tool 'measure' zelf heeft een veel hogere resolutie dan de kaart, dit geeft nog een kleine marge over de afstand afhankelijk van de exacte plaats waar geklikt wordt binnen de kaart. Uit deze metingen wordt geconcludeerd dat de kaartopbouw nauwkeurig genoeg is voor deze toepassing. Aangezien binnen het ontwerp van het laadstation rekening werd gehouden met een veel grotere marge zal een verschil van maximaal 2 cm geen probleem geven.

| | Werkelijke afstand (m) | Afstand in kaart (m) |
|---|---|---|
| a | 6,50 | 6,48 |
| b | 5,84 | 5,84 |
| c | 1,56 | 1,55 |

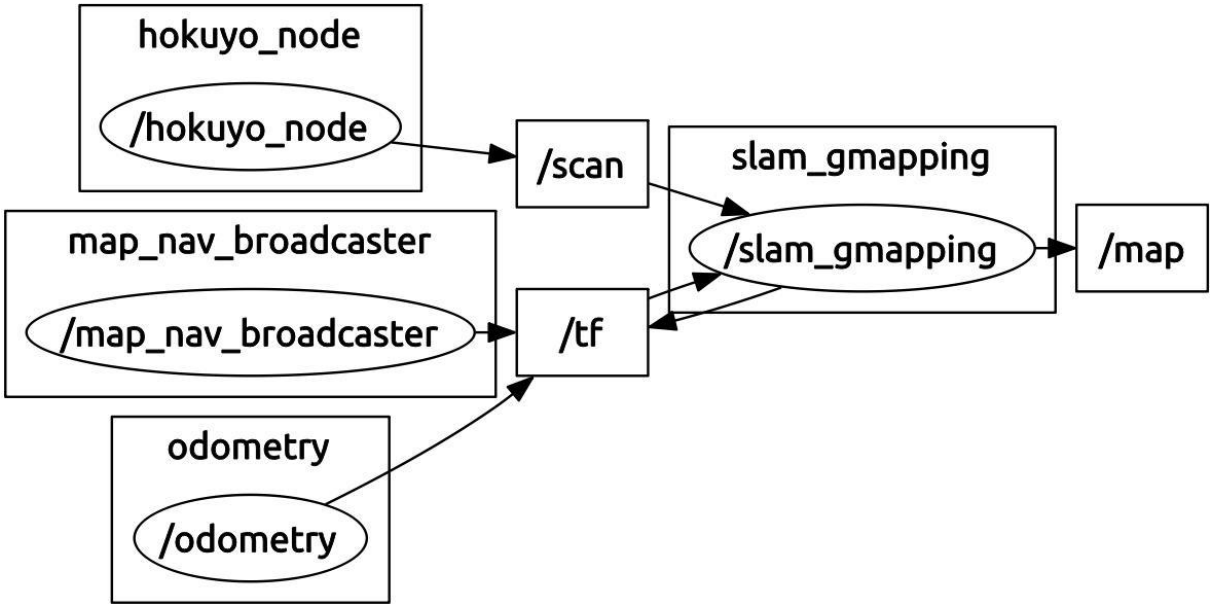
Tabel 2: Nauwkeurigheid Gmapping. Enkele reële afstanden worden vergeleken met de afstanden binnen Gmapping.




Figuur 33: Legende voor tabel 2.

#### 4.2.4.2 RQT-graph Gmapping

Onderstaande figuur geeft de RQT-graph van Gmapping weer. Wanneer deze RQT-graph vergeleken wordt met die van Hector Mapping wordt al snel duidelijk dat deze zeer gelijkaardig zijn. Het grote verschil is dat er een odometrienode bijkomt. Deze node geeft aan het topic /tf rechtstreeks de berekende plaats van de rolstoel door op basis van de encoderdata die via de socketverbinding verkregen worden. Op deze manier is er een eerste indicatie van de locatie van de robot in de kaart. Zo zal er in een omgeving met weinig goede landmarks toch een redelijk goede kaart opgebouwd kunnen worden. Voor meer uitleg over de verwerking van de odometrie, zie hoofdstuk 4.2.2 SLAM proces.

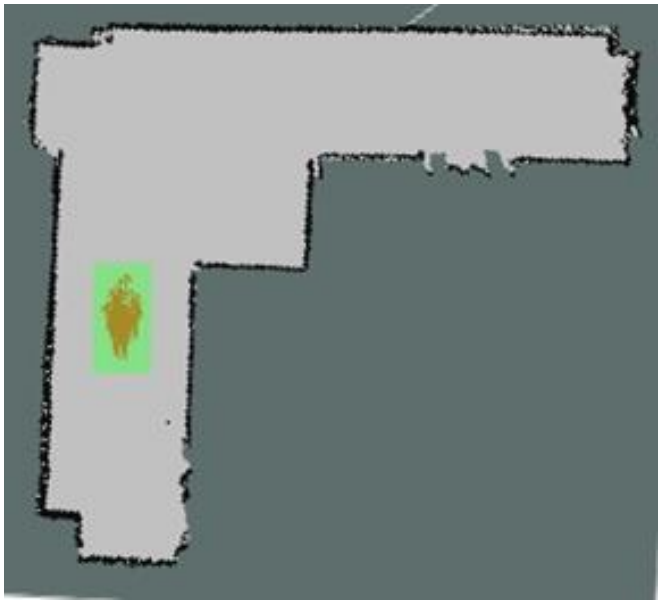



Figuur 34: RQT-graph Gmapping.

## 4.3 Lokalisatie in een vooraf opgebouwde kaart

Voor de lokalisatie in de kaart wordt binnen ROS gebruik gemaakt van het AMCL algoritme of voluit 'Adaptive Monte Carlo Localisation'. Dit algoritme berekent de plaats en de oriëntering van de robot in een vooraf opgebouwde kaart. De Monte Carlo lokalisatie is een algoritme voor robots dat gebruik maakt van 'particle filters'. Initieel heeft de robot geen enkel idee waar hij zicht bevindt binnen de kaart. Bij de start van de lokalisatie worden dus binnen de opgebouwde kaart op willekeurige plaatsen deeltjes of "partikels" gegenereerd. Elk partikel stelt een mogelijke pose van de robot voor, deze pose bevat zowel de (x, y) positie als de oriëntering van de robot en wordt binnen Rviz weergegeven door een pijltje. Aan de hand van de gemaakte scan wordt telkens van elk partikel de waarschijnlijkheid van de pose berekend worden. Poses met een te lage waarschijnlijkheid vallen hierbij weg. Wanneer de robot verplaatst zal elk partikel op basis van de odometriedata binnen de kaart verplaatsen gelijk met de robot. Opnieuw wordt de waarschijnlijkheid van alle partikels berekend aan de hand van de nieuwe scans. Uiteindelijk zullen de partikels zo convergeren tot de actuele plaats van de robot. Er zullen ook telkens nieuwe partikels in de buurt van deze actuele plaats worden aangemaakt om op deze manier ook de afwijkingen op de odometriedata op te vangen. Om het lokalisatieproces te versnellen kan er bij de start een geschatte pose worden meegegeven aan het algoritme. Zo zullen de partikels bij de start zich al rond deze geschatte pose genereren en zal de actuele plaats sneller gevonden worden. Onderstaande afbeeldingen tonen begin- en eindsituatie binnen ROS voor een test in de opgebouwde kaart. [18]

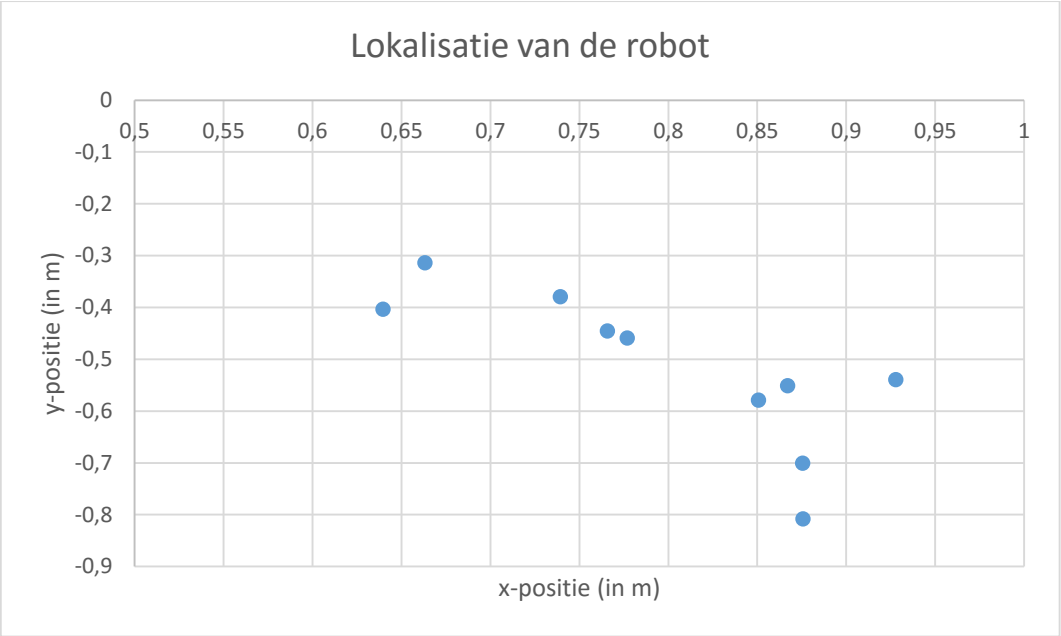
Figuur 35: Begintoestand AMCL, de rechthoek stelt de geschatte pose van de rolstoel weer met de grootste waarschijnlijkheid. De rode pijltjes stellen de andere geschatte poses voor.



Figuur 36: Eindtoestand AMCL, de rechthoek stelt de geschatte pose van de rolstoel weer met de grootste waarschijnlijkheid. De rode pijltjes stellen de andere geschatte poses voor.

### 4.3.1 Evaluatie en optimalisatie

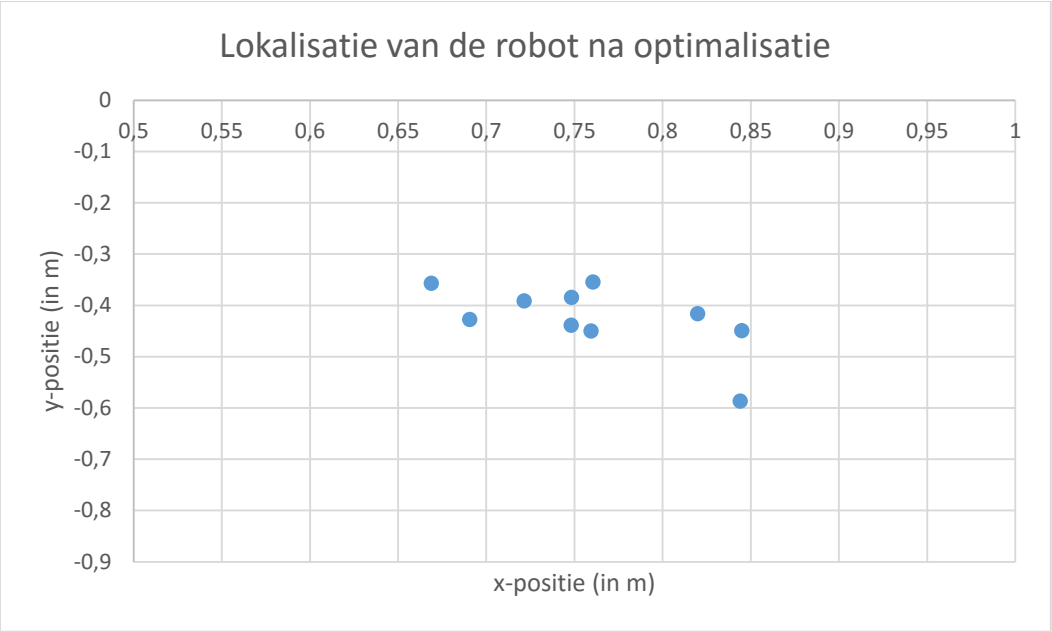
Voor correct aanmeren aan het laadstation is het belangrijk om te weten hoe precies de lokalisatie gebeurt. Hiervoor is er een evaluatie van het AMCL algoritme nodig. De precisie hiervan bepaalt namelijk of het achteraf mogelijk gaat zijn om de rolstoel correct in zijn laadstation te positioneren. Om dit te testen is een reeks van metingen gebeurd waarbij hetzelfde punt in de kaart 10 keer na elkaar werd aangevaren. Vervolgens werd de positie opgevraagd die geschat werd door het AMCL algoritme. Onderstaande grafiek geeft het resultaat van deze testen. Hierbij zijn de x- en y-posities de posities in de kaart.




Figuur 37: Grafiek die de spreiding van het AMCL algoritme weergeeft zonder aangepaste parametrering.

Hieruit blijkt dat dat er een spreiding op zowel de x-positie als op de y-positie zit. Deze ligt voor de x-positie tussen de 0,64 m en 0,93 m. Deze lokalisatie gebeurt dus met een nauwkeurigheid van rond de 30 cm. Op de y-positie zit een spreiding die tussen -0,8 cm en -0.3 cm ligt. De lokalisatie gebeurt hier dus met een nauwkeurigheid van rond de 50 cm. De nauwkeurigheid is duidelijk zeer laag!

Om deze nauwkeurigheid te vergroten zijn er enkele parameters aangepast met het oog op een betere lokalisatie. De parameter met het meeste invloed was '~odom_alpha1'. Hierbij wordt een maat van de verwachte storing van de rotatiebeweging van de wielen, ingegeven. Het is als het ware een maat die aangeeft in welke mate het AMCL algoritme de odometrie vertrouwt. Deze waarde is nu ingesteld op 10, standaard stond deze op 0,2. Dit zorgt er voor dat bij rotatie van de robot het algoritme meer partikels zal genereren waarvan de rotatie afwijkt van de rotatie volgens de odometrie. Op deze manier worden onnauwkeurigheden van de odometrie weggewerkt. Vervolgens is dezelfde test opnieuw uitgevoerd. Onderstaande grafiek geeft de resultaten van deze test.

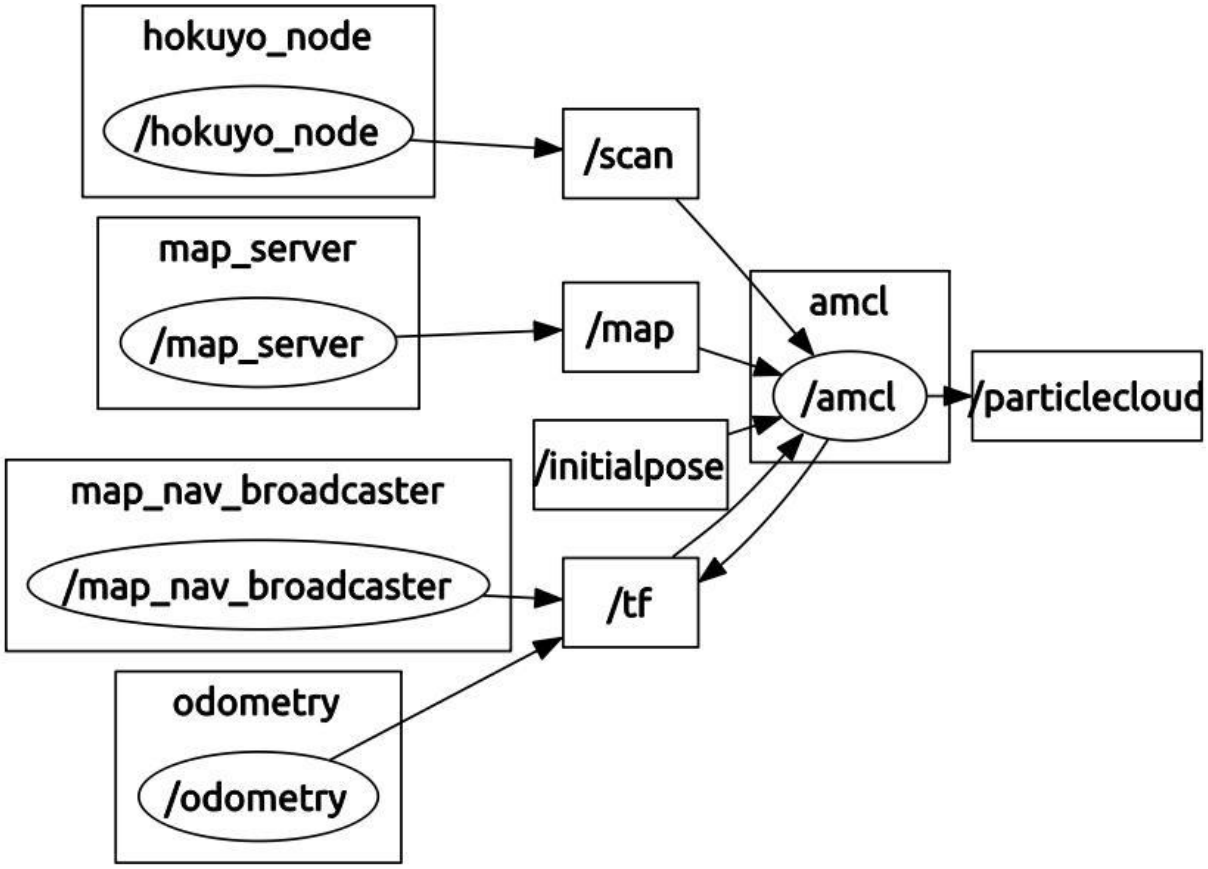



Figuur 38: Grafiek die de spreiding van het AMCL algoritme weergeeft na een eerste parametrering.

Er is een duidelijke verbetering zichtbaar. Zo is de spreiding van de x-positie verkleind tot iets minder dan 20 cm. Die van de y-positie is gedaald tot 25 cm. Hiermee is duidelijk aangetoond dat het aanpassen van de parameters invloed heeft op de precisie van het AMCL algoritme. Verder optimalisatie is echter nog mogelijk.

### 4.3.2 RQT-graph AMCL

De figuur hieronder geeft de RQT-graph van het AMCL algoritme weer. Duidelijk is dat de opbouw zeer gelijkaardig is aan die van Gmapping. Het grootste gedeelte komt terug en de uitleg hierover kan dus bij de vorige paragrafen teruggevonden worden. Verschillend hierbij is dat de kaart niet meer door het algoritme wordt opgebouwd. Het is dan ook de bedoeling om te lokaliseren in een vooraf opgebouwde kaart. Deze kaart werd opgebouwd met Gmapping en daarna met behulp van de map server, een tool binnen ROS, opgeslagen. Via deze map server wordt nu ook een kaart van de kamer of gebouw waarin de rolstoel zich bevindt gepubliceerd op het topic /map. Zoals in de uitleg over AMCL al vermeld werd, werkt dit algoritme met een particle filter. Het algoritme publiceert deze partikels dan ook op een topic "/particlecloud". Deze kunnen ook binnen RVIZ worden weergegeven, elk partikel wordt daar gevisualiseerd door een rood pijltje. De pose met de hoogste waarschijnlijkheid wordt doorgegeven aan het topic /tf. Indien de rolstoel bij het starten van de lokalisatie telkens op een vaste plaats zal komen te staan, kan deze locatie rechtstreeks aan het algoritme worden meegegeven via het topic /initialpose. Dit zal het lokalisatieproces aanzienlijk versnellen.




Figuur 39: RQT-graph AMCL.

## 4.4 Besluit

In dit hoofdstuk werden twee kaartopbouw algoritmes besproken, Hector SLAM en Gmapping. Gmapping gaf duidelijk een betere kaart. Dit komt omdat Gmapping, in tegenstelling tot Hector SLAM, wel gebruik maakt van de odometrie. Hierdoor is de kaartopbouw veel nauwkeuriger en betrouwbaarder. Dit is de reden waarom al in eerste instantie gekozen is om verder te gaan met Gmapping. Als eerste optimalisatie werd hier de parameter '~iterations' aangepast wat de kaart nog aanzienlijk verbeterde. Metingen via RVIZ toonden hierna aan dat de foutmarge van een kamer slechts twee centimeter was op een volledige lengte van bijna zeven meter. Deze foutmarge valt voor het volledige proces te verwaarlozen.

Vervolgens is het AMCL algoritme toegevoegd. Dit zorgt voor de lokalisatie van de rolstoel binnen de opgebouwde map. Om na te gaan of de precisie van het AMCL algoritme hoog genoeg is zijn er enkele testen uitgevoerd. Uit een eerste test blijkt dat er een relatief grote spreiding aanwezig is. In de x-positie hebben we te maken met een nauwkeurigheid van rond 30 cm, in de y-positie ligt deze rond de 50 cm. Om de nauwkeurigheid te vergroten is er een eerste optimalisatie gebeurd door aanpassing van de '~odom_alpha1' parameter. Dit resulteerde in een verbetering van de nauwkeurigheid. Voor de x-positie ligt deze nu op iets minder dan 20 cm en in de y-positie op 25 cm.

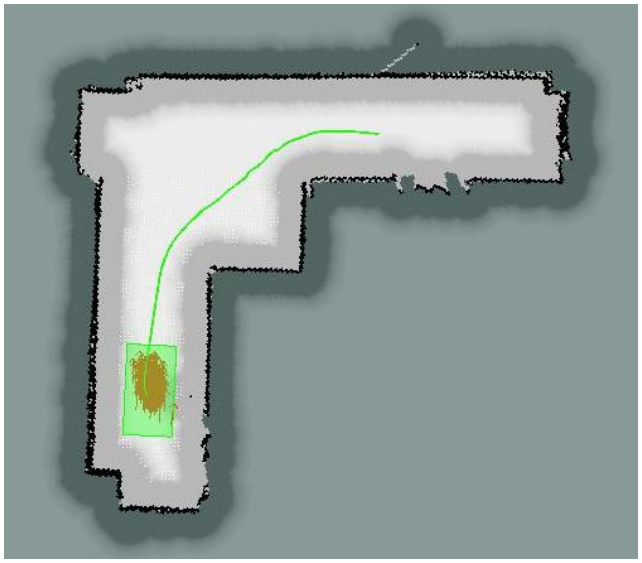
Als na implementatie van het padplanningsalgoritme blijkt dat de nauwkeurigheid nog steeds te laag is, dan is verdere optimalisatie mogelijk. Dit kan zowel voor Gmapping als voor het AMCL algoritme. Optimalisatie gebeurt in dit geval door aanpassing van de parameters van de algoritmes.

Als achteraf blijkt dat optimalisatie van de parameters nog niet voldoende nauwkeurigheid biedt, kan er nog altijd overwogen worden om een tweede laserscanner te gebruiken. Deze wordt het best aan de voorkant van de rolstoel geplaatst. Hierdoor kan de rolstoel zowel voor zich als achter zich landmarks observeren, wat de nauwkeurigheid aanzienlijk zal verhogen.

# 5 Padplanning en aanmeermanoeuvre

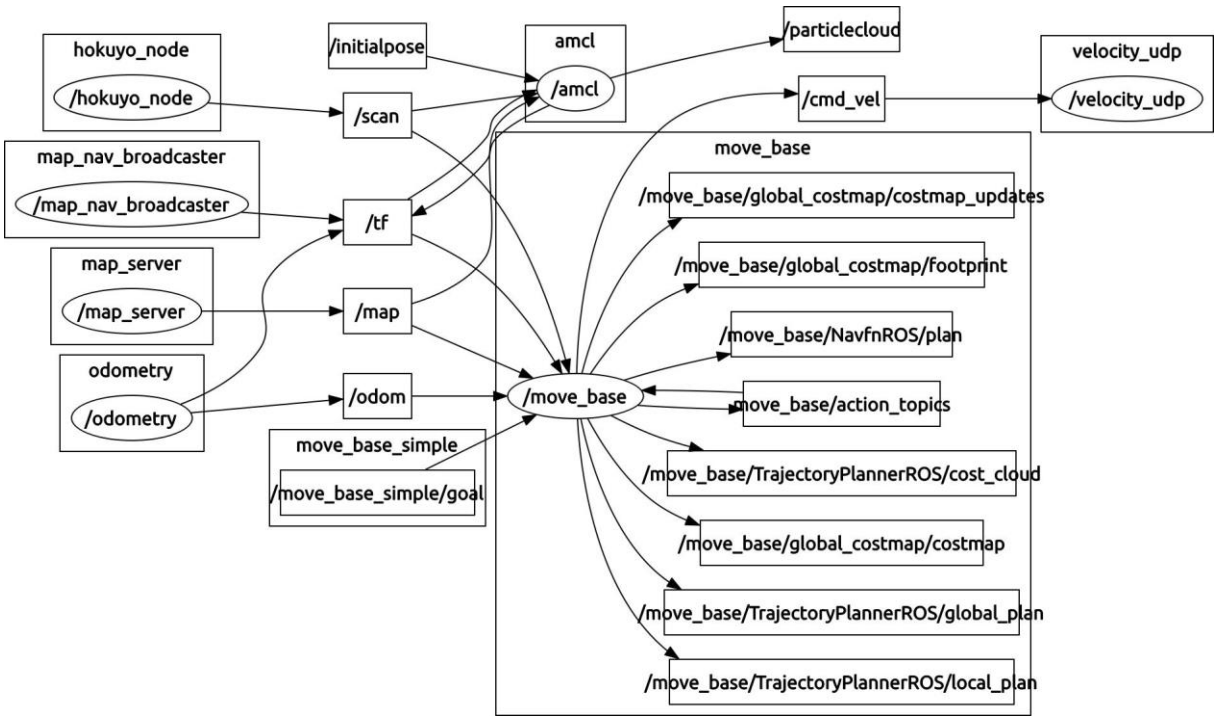
## 5.1 Padplanning

Wanneer de robot juist gelokaliseerd is binnen de opgebouwde kaart kan een pad gepland worden naar de doellocatie. Hier bestaan vele verschillende algoritmes voor. Omdat het binnen deze masterproef geen grote meerwaarde biedt om hier zeer diep op in te gaan zullen hier enkel de basisprincipes verklaard worden. Dit is voldoende om een globaal beeld te krijgen van de werking van deze algoritmes. Om een pad te kunnen plannen zal de robot zeer goed moeten weten hoe de omgeving eruit ziet. Hiervoor zal zowel globaal als lokaal een "cost map" van de omgeving gemaakt worden. Een cost map geeft voor elke locatie binnen de kaart de mogelijkheid weer om de locatie te betreden. Zo krijgt de robot een beeld van de toegankelijke locaties. De globale costmap zal vertrekken van de gebruikte kaart voor de lokalisatie. Binnen deze kaart worden alle muren en andere obstakels 'opgeblazen', dit wil zeggen dat er een bepaalde veiligheidsmarge wordt toegevoegd aan de muren wat dan visueel ook het beeld geeft dat deze worden opgeblazen. In de onderstaande figuur is deze globale cost map gevisualiseerd boven de gebruikte kaart voor de lokalisatie. De donkere delen van deze costmap worden door het algoritme beschouwd als niet toegankelijk. Zo zullen botsingen met de kamer vermeden worden. Hiernaast is het natuurlijk ook belangrijk om tijdelijke obstakels te kunnen herkennen. Hiervoor wordt ook nog een lokale costmap gemaakt. Deze tijdelijke obstakels worden herkend met behulp van de laserscanner. Ook deze obstakels worden opgeblazen. Bij het plannen van het pad wordt rekening gehouden met deze twee cost maps en zal het algoritme de makkelijkste weg naar de doelpositie berekenen. Er zal zowel een globaal als een lokaal plan berekend worden die dan samengevoegd worden tot het uiteindelijke plan. Hiervoor zijn ook gegevens over de grootte van de robot noodzakelijk. Een grote robot zal immers meer plaats nodig hebben om te manoeuvreren. Deze 'robot footprint' wordt in onderstaande figuur gevisualiseerd door de groene rechthoek die de uiterste afmetingen van de rolstoel zichtbaar maakt. Het algoritme zal dan afhankelijk van het type robot zowel de lineaire als de hoeksnelheid berekenen en doorsturen naar de robot. Via parameters kan zowel de maximale snelheid als versnelling ingesteld worden.



## 5.2 RQT-graph padplanning

Om het pad naar het laadstation te plannen is het natuurlijk belangrijk om de huidige plaats binnen de kaart te blijven kennen. Hiervoor blijft het volledige proces van de lokalisatie continu lopen. Hiernaast zal het padplanningsalgoritme uitgevoerd worden. Dit wordt weer duidelijk in onderstaande figuur die de RQT-graph van het volledige proces weergeeft. De node move_base verzorgt hier de padplanning. Via de node /move_base_simple/goal kan een doelpositie worden meegeven. Wanneer het laadstation op een vaste plaats in de kamer staat kan deze plaats hier worden ingegeven. Tijdens de testfase kan in RVIZ ook rechtstreeks een doelpositie worden gekozen. De padplanner berekent een pad en publiceert de benodigde lineaire en hoeksnelheid van de rolstoel op het topic /cmd_vel. Aangezien dit algoritme hier op een laptop wordt uitgevoerd moeten deze commando's nog via de UDP-socket worden verzonden naar de rolstoel. De communicatie hiervoor gebeurt via de node velocity_udp. Deze node werd in deze masterproef ook geïmplementeerd.




Figuur 40: RQT-graph padplanning.

## 5.3 Voorstel aanmeeralgoritme

Om op een juiste manier aan te meren aan het laadstation is het belangrijk om eerst naar een punt gelegen recht voor het laadstation te rijden. De juiste doelpositie voor de padplanning zal zich precies voor het laadstation moeten bevinden met de juiste oriëntatie, best is om de rolstoel zo te oriënteren dat deze evenwijdig met het laadstation staat. Op deze manier zal de rolstoel na het uitvoeren van het geplande pad enkel nog loodrecht achteruit moeten rijden.

Hiervoor kan een extra node worden geschreven die wordt uitgevoerd wanneer de doelpositie bereikt is. Eventueel kan het laadstation nog voorzien worden van een duidelijke en goed herkenbare landmark zodat de rolstoel bij een minder nauwkeurige lokalisatie en uitvoering van het geplande pad zich toch nog goed kan positioneren ten opzicht van het laadstation. Op deze manier zal ook het aanmeren vlotter en preciezer kunnen gebeuren.

## 5.4 Besluit

Momenteel kan enkel de weg naar een doelpositie gepland worden. De benodigde lineaire en hoeksnelheid worden al via de UDP-socket verzonden naar de PC van de rolstoel. Op deze PC moet nu nog de juiste koppeling naar de motoren gebeuren zodat deze aangestuurd worden. Vanaf dan kan de optimalisatie van het volledige proces gebeuren. Wel kan al gezegd worden dat het plaatsen van een tweede laserscanner aan de voorkant noodzakelijk is voor het zien van nieuwe obstakels binnen de kamer. Aangezien de robot logischerwijs het geplande pad vooruit zal afleggen en de enige laserscanner zich momenteel achteraan bevindt is dit dus niet ideaal.

# 6 Besluit

## 6.1 Overzicht

Tijdens deze masterproef zijn verschillende stappen gezet om een elektrische rolstoel autonoom te laten opladen. Om te beginnen moest er een laadstation ontworpen worden. Hiervoor werd een literatuurstudie naar zowel commerciële als niet commerciële laadstations uitgevoerd. Deze hebben echter geen concreet, passend ontwerp voor dit project naar voren gebracht. Ze hebben wel geleden tot enkele conceptueel zeer goede ideeën voor het ontwerp van het laadstation. Aan de hand van deze ideeën is een laadstation ontworpen in Inventor. De constructie gebeurde aan de hand van een polycarbonaten plaat die onder de juiste hoeken geplooid werd. Hierop zijn de oplaadstrips gemonteerd. Het gebruik van het laadstation mag natuurlijk geen negatieve invloed hebben op het laadproces van de batterijen. Om er zeker van te zijn dat dit niet het geval is zijn er enkele testen uitgevoerd. Hierbij is vooral gekeken naar de spanningsval over het laadstation. Deze lag bij hoeken van minder dan de 45° rond de 60mV. Hieruit kan besloten worden dat er zich geen negatieve invloeden op het laadproces voordoen.

De gebruikte rolstoel was bij aanvang van deze masterproef al beschikbaar. Deze was reeds uitgerust met onder andere een PC, een laserscanner en encoders. De encoderdata werd al ingelezen in de PC. Ondanks al het voorgaande werk waren er toch nog enkele aanpassingen nodig aan het voertuigplatform. Zo is in het verleden de USB-connector van eenzelfde laserscanner afgebroken. Om dit in de toekomst te vermijden is er een beschermblokje ontworpen en geproduceerd. Hiervoor is een 3D-printer gebruikt. Ook zijn er aanpassingen gebeurt aan de hardware van de rolstoel. Dit was nodig omdat de PC, die verbonden is met de rolstoel, nog op Ubuntu 10.04 werkt. De gebruikte robot softwarebibliotheek ROS is niet compatibel met deze versie van Ubuntu. Omdat Ubuntu 10.04 niet meer ondersteund wordt, kan deze ook niet worden geüpdatet. Daarom is er een aparte laptop gebruikt voor de programmering, deze laptop werkt wel op Ubuntu 14.04. Voor de communicatie tussen de laptop en de PC is er gekozen voor een UDP socket. Dit omdat deze zeer veel data op korte tijd kan doorgeven. Via deze verbinding zullen enerzijds de encoderdata van de wielen worden doorgestuurd naar de laptop. Anderzijds zullen de benodigde lineaire- en hoeksnelheid voor de uitvoering van de padplanning worden teruggestuurd van de laptop naar de PC.

Voor de kaartopbouw zijn er twee algoritmes besproken, Hector SLAM en Gmapping. Gmapping gaf duidelijk een betere kaart. Dit komt omdat Gmapping, in tegenstelling tot Hector SLAM, wel gebruik maakt van de odometrie. Hierdoor is de kaartopbouw veel nauwkeuriger en betrouwbaarder. Dit is de reden waarom al in eerste instantie gekozen is om verder te gaan met Gmapping. Als eerste optimalisatie van Gmapping werd de parameter '~iterations' aangepast wat de kaart nog aanzienlijk verbeterde. Metingen via RVIZ toonden hierna aan dat de foutmarge van een kamer slechts twee centimeter was op een volledige lengte van bijna zeven meter. Deze foutmarge valt voor het volledige proces te verwaarlozen.

Vervolgens is het AMCL algoritme toegevoegd. Dit zorgt voor de lokalisatie van de rolstoel binnen de opgebouwde kaart. Om na te gaan of de precisie van het AMCL algoritme hoog genoeg is zijn er enkele testen uitgevoerd. Uit een eerste test blijkt dat er een relatief grote spreiding aanwezig is. Voor de x-positie was er met een nauwkeurigheid van rond 30cm, voor de y-positie lag deze rond de 50cm. Om de nauwkeurigheid te vergroten is er een éérste optimalisatie gebeurt door aanpassing van de '~odom_alpha1' parameter. Dit resulteerde in een verbetering van de nauwkeurigheid. Voor de x-positie ligt deze nu op iets minder dan 20cm en in de y-positie op 25cm. Verdere optimalisatie door middel van een betere parametrering is uiteraard nog mogelijk.

Voor de beweging van de rolstoel naar zijn laadstation moet er een pad gepland worden. Momenteel kan enkel de weg naar een doelpositie gepland worden. Het berekende pad wordt echter nog niet uitgevoerd. De benodigde lineaire en hoeksnelheid worden wel al via de UDP-socket verzonden naar de PC van de rolstoel. Op deze PC moet enkel nog de juiste koppeling naar de motoren gebeuren zodat deze aangestuurd worden.

## 6.2 Bijdragen

Deze masterproef onderzocht de mogelijkheden om het laadproces van elektrische rolstoelen te automatiseren voor andersvalide personen. Hierbij werd nauwlettend de nauwkeurigheid van het volledige proces in de gaten gehouden, alsook de mogelijkheid tot eenvoudige implementatie in verschillende omgevingen. Op deze manier kan in de toekomst hopelijk bijgedragen worden tot een makkelijkere hulpverlening in de zorgsector en met daarbij ook een grotere zelfstandigheid van de hulpbehoevenden.

Vroeger moest, bij het opladen, de verbinding tussen het laadstation en de rolstoel manueel gebeuren. Dit gebeurde via een XLR-connector. Het inpluggen van deze connector vormde voor andersvaliden voor een groot probleem. Om dit op te lossen is de XLR-koppeling vervangen door een laadstation. Aan het ontworpen laadstation kan de rolstoel zich, mits de juiste programmatie, volledig autonoom opladen. Dit zorgt voor een zeer grote verbetering voor de andersvaliden!

## 6.3 Toekomstig werk

Er zijn nog enkele stappen nodig om dit project te voltooien. Zo zou het berekende pad nog moeten uitgevoerd worden. Hiervoor worden wel al de juiste data doorgestuurd naar de PC, maar deze verwerkt deze alleen nog niet. Vanaf dan kan de optimalisatie van het volledige proces gebeuren. Vervolgens moet gecontroleerd worden of de nauwkeurigheid van het hele proces groot genoeg is om correct te kunnen aanmeren. Indien dit niet het geval is moet er een tweede laserscanner geplaatst worden. Zo kunnen zowel aan de voorkant als aan de achterkant van de rolstoel landmarks geobserveerd worden. Ook de procedure voor correct aanmeren zal nog uitgewerkt moeten worden. Daarna kan het prototype van het laadstation verder geëvalueerd worden op basis van de nauwkeurigheid van het volledige proces. Indien nodig kunnen ook de nodige verstevigingen hieraan nog worden aangebracht.

Verder kunnen er nog verschillende maatregelen getroffen worden voor verdere in bedrijf name. Zo is het belangrijk dat er zo weinig mogelijk batterijcapaciteit verloren gaat. Hierdoor is het wenselijk dat de PC niet continu aan staat. Dit valt te vermijden door implementatie van een spanningsdetectie op de batterijen van de rolstoel, die bij een te lage spanning de PC aanschakelt. Deze moet vervolgens automatisch de nodige programma's starten om zijn weg naar het laadstation te vinden en deze uit te voeren. Na voltooiing van het laadproces mag de PC terug uitschakelen.

Wanneer deze rolstoel effectief zijn weg vindt naar de zorgsector is het ook noodzakelijk dat er door de eindgebruikers ingesteld kan worden op welke momenten de rolstoel mag opladen. Zo kan deze opgeladen worden op momenten dat de gebruiker de rolstoel niet nodig heeft. Op deze manier ondervindt hij geen hinder van het laadproces en blijft zijn mobiliteit zoveel mogelijk behouden. Het laden van de rolstoel kan dan 's nachts gebeuren en eventueel zo worden ingesteld dat de rolstoel na het laden terugkeert naar de plaats van vertrek.

# Literatuurlijst


[1] „Netwerk Duurzame Mobiliteit," [Online]. Available: http://www.duurzame-mobiliteit.be/standpunt/de-zelfrijdende-auto. [Geopend 1 Juni 2016].

[2] C. Sight, „Mainpress," [Online]. Available: www.mainpress.com/nederlands/dossier_automation/pdf/robot3D.pdf. [Geopend 1 Juni 2016].

[3] H. A. P. E. S. J. Demeester E., „ML, MAP and greedy POMDP shared control: comparison of wheelchair navigation assistance for switch interfaces," Heverlee, 2014.

[4] J. Layton, „How Robotic Vacuums Work," HowStuffWorks, 2006.

[5] „Robovac weebly," [Online]. Available: http://www.robovac.weebly.com. [Geopend Oktober 2015].

[6] S. Vandevoordt, „livios," [Online]. Available: http://www.livios.be/nl/bouwinformatie/tuin/tuinaanleg/hoe-werkt-een-robotmaaier/. [Geopend oktober 2015].

[7] „robotstofzuigerinfo," [Online]. Available: http://www.robotstofzuigerinfo.nl/over-robotstofzuigers/hoe-een-robot-stofzuiger-werkt/. [Geopend oktober 2015].

[8] „allonrobots," [Online]. Available: http://www.allonrobots.com/robot-vacuum.html. [Geopend oktober 2015].

[9] „Probotics," Probotics, 2004. [Online]. Available: http://www.probotics.com/rm200-rm400/rm200-rm400-robomow.htm. [Geopend Oktober 2015].

[10] G. Song, „Automatic Docking System for Recharging," *IEEE Transactions on Consumer Electronics,* nr. Vol. 57, No. 2, p. 8, 2011.

[11] C. S. &. M. Serrao, „R&D of a loading and charging dock for an Autonomous Indoor Vehicle (AIV)," Limburg Catholic University College, Belgium, 2013.

[12] D. N. B. J. a. G. S. S. Milo C. Silverman, „Staying Alive: A Docking Station for Autonomous Robot Recharging," Raytheon Electronic Systems, El Segundo,Californië, 2002.

[13] G. J. Pieter Gijsen, „Zelfkalibratie van laserscannerpose en odometrieparameters voor mobiele robots met magnetische wiel-encoders," Uhasselt, Diepenbeek, 2014-2015.

[14] „Hokuyo official website," Hokuyo automatic co, 2012. [Online]. Available: http://www.hokuyo-aut.jp/02sensor/07scanner/download/products/urg-04lx-ug01/. [Geopend 20 Mei 2016].

[15] L. Joseph, Mastering ROS for Robotics Programming, Packt Publishing ltd, 2015.

[16] N. gekend, „Wikipedia," [Online]. Available: https://nl.wikipedia.org/wiki/Simultaneous_localization_and_mapping. [Geopend 12 Mei 2016].

[17] S. R. e. M. R. Blas, „SLAM for Dummies".

[18] „Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Monte_Carlo_localization. [Geopend 26 Mei 2016].

[19] D. Scocco, „Programming Logic," 25 04 2014. [Online]. Available: http://www.programminglogic.com/sockets-programming-in-c-using-udp-datagrams/. [Geopend 04 2016].

[20] D. Mangus, „ROS Answers," 16 11 2011. [Online]. Available: http://answers.ros.org/question/11973/gathering-wheel-encoder-data-and-publishing-on-the-odom-topic/. [Geopend 02 2016].

[21] N. Kumar, „ROS Answers," 20 04 2015. [Online]. Available: http://answers.ros.org/question/207392/generating-odom-message-from-encoder-ticks-for-robot_pose_ekf/. [Geopend 02 2016].

# Bijlagen

## Bijlage 1: De odometrienode die gebruikt werd bij Gmapping, AMCL, en het padplanningsalgoritme

Deze node werd ontwikkeld uit verschillende bestaande programma's die werden samengevoegd en bijgewerkt tot onderstaand eindresultaat. [19] [20] [21]

```
#include "ros/ros.h"
#include <geometry_msgs/Vector3.h>
#include <tf/transform_broadcaster.h>
#include <nav_msgs/Odometry.h>
#include <iostream>
#include <stdio.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
#include <arpa/inet.h>
#include <string>

using namespace std;

/*Initialisation of the variables*/
double _PreviousLeftEncoderAngle = 0;
double _PreviousRightEncoderAngle = 0;
ros::Time current_time_encoder, last_time_encoder;
double diam=0.3533;
double DistancePerRadian=diam/2;
double x=0;
double y=0;
double th=0;
double W=0;
double V=0;
double vx=0;
double vy=0;
double vth=0;
double deltaLeft=0;
double deltaRight=0;
double radius=0;
double d=0.588;
double type;
double id;
double angle_left;
double angle_right;
double timestamp;

int main(int argc, char **argv)
{
  ros::init(argc, argv, "odometry");

  /******UDP SERVER CODE*******/
  int udpSocket, nBytes;
  char buffer[1024];
  struct sockaddr_in serverAddr, clientAddr;
```

```
  struct sockaddr_storage serverStorage;
  socklen_t addr_size, client_addr_size;
  int i;

  /*Create UDP socket*/
  udpSocket = socket(PF_INET, SOCK_DGRAM, 0);

  /*Configure settings in address struct*/
  serverAddr.sin_family = AF_INET;
  serverAddr.sin_port = htons(2000);
  serverAddr.sin_addr.s_addr = inet_addr("192.168.0.2");
  memset(serverAddr.sin_zero, '\0', sizeof serverAddr.sin_zero);

  /*Bind socket with address struct*/
  bind(udpSocket, (struct sockaddr *) &serverAddr, sizeof(serverAddr));

  /*Initialize size variable to be used later on*/
  addr_size = sizeof serverStorage;

  /*****END UDP SERVER CODE*****/

  /*ROS initialisation*/
  ros::NodeHandle n;
  ros::Publisher odom_pub = n.advertise<nav_msgs::Odometry>("odom",
50);
  tf::TransformBroadcaster odom_broadcaster;
  ros::Rate r(200.0);

  /*Set previous encoder time*/
  last_time_encoder = ros::Time::now();

  while(n.ok())
  {
    /***** UDP SERVER CODE *****/

    /* Try to receive any incoming UDP datagram. Address and port of
      requesting client will be stored on serverStorage variable */
    nBytes = recvfrom(udpSocket,buffer,1024,0,(struct sockaddr
*)&serverStorage, &addr_size);

    /*Convert message received to uppercase*/
    for(i=0;i<nBytes-1;i++)
    buffer[i] = toupper(buffer[i]);

    /*Send uppercase message back to client, using serverStorage as the
address*/
    sendto(udpSocket,buffer,nBytes,0,(struct sockaddr
*)&serverStorage,addr_size);

    /*Generate stringstream from the incoming message*/
    std::stringstream message (buffer);
    std::vector<int> data;

    /*Split stringstream into variables*/
    message >> type;
    message >> id;
    message >> angle_left;
    message >> angle_right;
    message >> timestamp;

    /*Set the initial angles (only the first time)*/
```

```
    static double angleLeftInit = angle_left;
    static double angleRightInit = angle_right;

    /*Calculate the difference of the angle in case the initial angle
wasn't zero*/
    angle_left -= angleLeftInit;
    angle_right -= angleRightInit;

    /***** END UDP SERVER CODE *****/

    /*Set the current time*/
    current_time_encoder = ros::Time::now();

    /*Calculate the difference from the previous angle*/
    deltaLeft = angle_left - _PreviousLeftEncoderAngle;
    deltaRight = angle_right - _PreviousRightEncoderAngle;

    /*Calculate the velocities of the left and right wheel, vx and vy*/
    vx = deltaLeft * DistancePerRadian / (current_time_encoder -
last_time_encoder).toSec();
    vy = deltaRight * DistancePerRadian / (current_time_encoder -
last_time_encoder).toSec();

    /*If the velocities are the same, the robot moves right forward. If
not, calculate the angular and linear velocity*/
    if (fabs(vx-vy)<0.000001)
    {
      V = vx;
         W = 0;
    }
     else
    {
        radius = (vx * d) / (vy - vx); // Anti Clockwise is positive
      if (fabs(radius)<0.000001)
      {
            W = vy/d;
      }
      else
      {
            W = vx/radius; // Rotational velocity of the robot
      }
        V = W * (radius + d/2); // Translation velocity of the robot
    }

    /*vth = angular velocity*/
    vth= W;

    /*Set the previous encoder angle*/
    _PreviousLeftEncoderAngle = angle_left;
    _PreviousRightEncoderAngle = angle_right;

    /*compute odometry in a typical way given the velocities of the
robot*/
    double dt = (current_time_encoder - last_time_encoder).toSec();
    double delta_th = vth * dt;
    th += delta_th;
    double delta_x = V * cos(th) * dt;
    double delta_y = V * sin(th) * dt;

    x += delta_x;
    y += delta_y;
```

```
    /*since all odometry is 6DOF we'll need a quaternion created from
yaw*/
    geometry_msgs::Quaternion odom_quat =
tf::createQuaternionMsgFromYaw(th);

    /*first, we'll publish the transform over tf*/
    geometry_msgs::TransformStamped odom_trans;
    odom_trans.header.stamp = current_time_encoder;
    odom_trans.header.frame_id = "odom";
    odom_trans.child_frame_id = "base_link";

    odom_trans.transform.translation.x = x;
    odom_trans.transform.translation.y = y;
    odom_trans.transform.translation.z = 0.0;
    odom_trans.transform.rotation = odom_quat;

    /*send the transform*/
    odom_broadcaster.sendTransform(odom_trans);

    /*next, we'll publish the odometry message over ROS*/
    nav_msgs::Odometry odom;
    odom.header.stamp = current_time_encoder;
    odom.header.frame_id = "odom";

    /*set the position*/
    odom.pose.pose.position.x = x;
    odom.pose.pose.position.y = y;
    odom.pose.pose.position.z = 0.0;
    odom.pose.pose.orientation = odom_quat;

    /*set the velocity*/
    odom.child_frame_id = "base_link";
    odom.twist.twist.linear.x = V;
    odom.twist.twist.linear.y = 0;
    odom.twist.twist.angular.z =W;

    /*publish the message*/
    odom_pub.publish(odom);
    last_time_encoder = current_time_encoder;
    ros::spinOnce();
    r.sleep();
  }
}
```

# Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
**Integratie van kaartopbouw, lokalisatie en padplanning voor het autonoom aanmeren van een elektrische rolstoel aan een ontworpen laadstation**

Richting: **master in de industriële wetenschappen: energie-automatisering**
Jaar: **2016**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

**Drijkoningen, Bert** **Rubens, Lukas**

Datum: **7/06/2016**