Er moet een verslag worden ingeleverd met een beschrijving van het project, de hardware en de software, welke problemen je hebt gehad, en hoe het uiteindelijke prototype afwijkt van de oorspronkelijke beschrijving.

Topverslag

# Beschrijving

In dit project is met behulp van een Arduino Uno een voertuig gemaakt waarmee een omgeving wordt verkend. De omgeving wordt in kaart gebracht door met dit voertuig naar de positie te gaan die het meest dichtbij is. Rijdt het voertuig tegen een obstakel aan, dan wordt dit waargenomen en opgeslagen in de interne kaart. Als de robot geen pad meer heeft om te verkennen dan zal hij stoppen met verkennen. De huidige kaart wordt dan naar het EEPROM van de arduino geschreven. Deze kaart kan daarna uitgelezen worden op de laptop.

## Hardware

Het voertuig bestaat uit een Arduino Uno in combinatie met een motorshield (DFRduino Motor Drive). Deze zijn gemonteerd op een constructie waarop 2 motoren en een zwenkwieltje zitten. Op de voorkant van het voertuig zit een bumper die gekoppeld is aan druksensoren. Rijdt het voertuig ergens tegen aan, dan worden deze sensoren ingedrukt en krijgt de Arduino een signaal.

## Software

De software is verdeeld in verschillende klassen, elk met een eigen functie. Hieronder staat een overzicht van de gebruikte klassen.

### Map

Deze klasse is verantwoordelijk voor het bijhouden van de kaart. Een kaart bestaat uit een tweedimensionale array van chunks. Elke chunk heeft twee waarden: de eerste geeft aan of de chunk al behandeld is, de tweede geeft aan of deze chunk een obstakel bevat.

Naast methoden om deze waarden in de kaart te veranderen, bevat deze klasse ook twee methoden om de kaart te schrijven naar en te lezen van de eeprom. Op deze manier kan het voertuig zijn werk zonder via een serieële kabel vebonden te zijn, terwijl na het verkennen de kaart uitgelezen kan worden

### Motor

Deze klasse zorgt voor het aansturen van de motor. Elke motor kan apart aangestuurd worden, maar ook tegelijk. Daarnaast bevat deze klasse methoden waarmee het voertuig een kwartslag linksom of rechtsom wordt gedraaid.

### Pathfinder

Met behulp van de map zoekt de Pathfinder-klasse vakjes op die nog verkend moeten worden. Door middel van deze klasse weet de robot welke kant hij op moet rijden om naar het volgende vakje toe te verplaatsen.

Het pad word bepaald door elke vakje te bepalen wat de afstand is tot dit vakje. De afstand is uitgedrukt in het aantal verplaatste vakjes plus het aantal draaiingen van 90 graden. Dit wordt bepaald door voor elke vakje te kijken wat de kosten zijn voor de buren en hier 1 en de kosten voor een eventuele draaiing bij op te tellen. Als het minimum hiervan minder is dan de oude waarde, worden de huidige kosten en richting na aankomst veranderd. Dit gaat door tot er niets meer veranderd is. Op dit moment zijn voor elk vakje de kosten berekend. Het vakje met de minimale kosten waarbij dit vakje nog niet is verkend, wordt gekozen als doel.

Aan de hand van de onthouden richtingen kan de route zelf worden bepaald. Als het voertuig na aankomst op een vakje naar het noorden wijst, weet je dat hij hier is gekomen vanuit het zuiden. De stap om hier te komen is dus om beweging naar het noorden te doen.

### Bumper

Deze klasse controleert of er een botsing is met een obstakel. Dit gebeurt door simpelweg de ingangssignalen van de twee druksensoren uit te lezen.

### Test

Bij een goede applicatie horen testen. De bedoeling van deze klasse is dan ook om te testen of verscheidene methoden in verschillende klassen werken naar verwachting. Met name voor het pathfinding-algoritme is dit van belang, aangezien dit redelijk foutgevoelig is. Daarnaast worden de functies van de Map klasse en het lezen en schrijven van de kaart naar het EEPROM getest.

# Problemen

### Afwijking

Omdat er een klein verschil zit tussen beide motoren en de wielen, heeft het voertuig bij gelijk vermogen op beide motoren een afwijking bij het vooruit rijden. Om dit probleem op te lossen hebben we een variabele *DEVIATION* toegevoegd, waarmee het vermogen van de linker motor naar boven of naar beneden kan worden afgesteld. Door deze variabele goed in te stellen, rijdt het voertuig recht vooruit.

Daarnaast werkte het zwenkwieltje niet optimaal. Deze is af en toe een beetje stroef, waardoor de rotaties niet goed worden uitgevoerd.

### Rotatie

De motor is gekalibreerd om een rotatie van om en nabij 90° te kunnen maken. Dit is echter gedaan door beide motoren gedurende een constante tijd te laten draaien. De ene motor vooruit, de andere achteruit. Helaas is dit niet altijd een perfecte hoek van 90 graden. Beide motoren draaien even snel en de wieltjes zijn niet perfect rond wat er voor zorgt dat er een kleine afwijking zit in de besturing. De snelheid waarmee de motoren draaien is ook afhankelijk van het vermogen dat nog in de batterij zit. Als het programma te lang draait dan zal de afwijking te groot worden waardoor de robot niet meer precies is waar hij denkt dat hij is.

### Voeding

Omdat we batterijen als voeding gebruiken, neemt het vermogen af naarmate de tijd verstrijkt. Zoals hierboven al is beschreven levert dit problemen op bij het kalibreren en bij het draaien van 90 graden. Dit is opgelost door elke keer voor het uitvoeren van het programma eerst de kalibratie handmatig uit te voeren. Dit is natuurlijk niet optimaal. In de aanbevelingen wordt dan ook omschreven hoe dit verbeterd kan worden.

# Aanbevelingen

Naar aanleiding van de problemen en het uiteindelijke resultaat worden hieronder enkele aanbevelingen en mogelijke verbeteringen omschreven.

## Stappenteller/Stappenmotor

Aangezien de huidige bewegingen, zoals rotaties, een constante tijd worden uitgevoerd, is dit vaak niet accuraat. Als verbetering zou een stappenteller kunnen helpen. Hiermee is het mogelijk om per wiel te meten hoe vaak deze is rondgedraaid. Zo werken de verplaatsingen nog steeds correct als het vermogen van de batterij lager is.

Een andere oplossing is het gebruik van stappenmotoren. Met deze motoren kun je precies zeggen hoeveel rotaties de motor uit moet voeren. Ook dit zorgt ervoor dat het aantal rotaties nog steeds correct is ondanks een lager batterijvermogen.

Dit is echter nog steeds niet ideaal. Het kan zijn dat de banden bijvoorbeeld niet voldoende grip hebben, waardoor de uitgevoerde rotatie nog steeds niet correct is. Betere banden en vooral een beter zwenkwieltje kunnen verbetering brengen.

## Batterij

Het vermogen van de door ons gebruikte batterij verminderde al snel. Hierdoor was het ook lastig om de motoren te kalibreren. Een andere aanbeveling is dan ook om betere batterijen of een betere accu te gebruiken. Een tweede oplossing is om de batterijen tijdens het rijden op te laden, bijvoorbeeld met zonnepanelen.

## Beter pathfinding-algoritme

Het door ons gebruikte pathfinding-algoritme is redelijk intensief. In deze situatie is dit nog niet zo erg, maar wordt de kaart groter dan kan dit best wat tijd schelen. In elke iteratie wordt namelijk voor elk vakje opnieuw de kosten berekend, ook al heeft deze al lang de minimale waarde bereikt. Een aanbeveling is dan ook om een ander algoritme, bijvoorbeeld een gebaseerd op het Dijkstra-algoritme, te gebruiken. Dit zorgt er ook voor dat je tussenresultaten kunt bijhouden, wat zorgt voor minder berekeningen. In plaats van elke keer voor elk veld opnieuw de kosten uit te rekenen, wordt dit gedaan als de waarde van één van de buren veranderd.

# Resultaat

We hebben naast de omschrijving zoals hierboven een extra knopje toegevoegd. Bij het opstarten wordt eerst de kaart via de seriële kabel geprint, zodat deze eventueel uitgelezen kan worden. Pas nadat het knopje is ingedrukt, begint het voertuig met verkennen.