Ruben Leenknecht, Emiel Vanspranghels,
Otto Meerschman, Staf Rys, Camille Louagie

Eindverslag

vrijdag 21 mei, 2021

Groep 1: Safety First

Inhoudsopgave

1 Inleiding

De groei en het verval van steden

Meer en meer zien we een groei van het verstedelijkt gebied. Met deze groei nemen ook de problemen toe: zoals onder meer criminaliteit [14], lawaai [3] en milieuvervuiling[11]. Als we terugkijken in de tijd, zien we een meermaals voorkomende situatie. Steden zitten in een bloeiperiode, bereiken een verzadingspunt en kunnen daarna de vraag niet meer aan. Dit heeft een negatief effect op de economie en de efficiëntie in een stad. Zo was er ook de val van Rome, nadat deze een hoogtepunt had bereikt. Oude steden hadden dan wel een kleiner bereik, maar de relatie tussen technologie en stedelijke groei blijft dezelfde [4] . We staan opnieuw voor een keerpunt, waarbij we moeten kiezen tussen groeien of blijven steken.

We moeten de technologie die we voor handen hebben kunnen gebruiken om dit vastlopen te voorkomen. Met andere woorden moeten we van onze steden zogezegde 'slimme steden' maken [1]. In ons dagdagelijkse leven worden we geconfronteerd met inefficiënte situaties die worden opgelost door een 'slimme' aanpak. Daarom is het nodig mensen bewust te maken van het nut van deze slimme steden.

Slimme steden

Maar wat zijn slimme steden nu juist? Met slim wordt de technologische innovatie bedoeld die opweegt tegen fysieke beperkingen. Een stad is een gebied van interacties en bijgevolg ook problemen en confrontaties [2]. In een kleine ruimte worden verscheidene dingen geconcentreerd samengebracht. Een stad heeft een veelheid aan functies en is pas doelmatig wanneer ze deze functies met succes volbrengt [15]. Een slimme stad wordt bereikt door stedelijke werking efficiënt te laten verlopen en door het vereenvoudigen van (openbare) diensten. Informatie- en communicatietechnologie (ICT), wordt gecombineerd met dagdagelijkse objecten om die interacties te verbeteren en stedelijke problemen te verminderen. Ondanks de vele vooruitgang van de afgelopen jaren, is het idee van een stad die volledig voorgeprogrammeerd gestuurd wordt, zonder menselijke tussenkomsten, voorlopig nog steeds een utopie. Nieuwe mogelijkheden en doorbraken zijn een drijfveer opdat dit ooit realiteit wordt [12].

Zelfrijdende auto's

Een van de middelen om de efficiëntie te verhogen in een stad is door het invoeren van zelfrijdende auto's die zelfstandig kunnen deelnemen aan het verkeer. Niet alleen wordt er extra tijd gecreëerd voor de bestuurder, die hij nuttiger kan spenderen, maar ook kunnen de auto's dichter op elkaar volgen en is het naleven van de verkeersregels verzekerd, mits de regels correct geïnterpreteerd worden. Dit kan ook voor een vermindering van het aantal verkeersongevallen zorgen en bijgevolg minder filedrukte. Het concept heeft niet alleen maar voordelen. Zo zouden hackers kunnen zorgen voor enorme verkeersproblemen. De verkeersgevallen die zich voordoen veroorzaken juridische conflicten in verband met aansprakelijkheid. Wie is verantwoordelijk wanneer iets misloopt? Is dit de bestuurder, de constructeur of de softwareontwikkelaar? Ook een inkomst van de overheid valt weg wanneer geen verkeersboetes of accijnzen op brandstof worden betaald. Daardoor zullen er meer belastingen op elektriciteit moeten geheven worden. Daarnaast zullen verschillende beroepen zoals buschauffeurs, rijinstructeurs en vrachtwagenchauffeurs overbodig worden, wat voor meer werkeloosheid onder minder geschoolden zal leiden [10].

Ondanks de Vlaamse overheid autonoom rijden stimuleert met projecten zoals Smart Highway en project CONCORDA, staat men in Vlaanderen sceptisch tegenover het idee [13]. Zo zou er een te grote psychologische drempel zijn [6]. Toch is er sprake van een grote marktpotentie. Bij de nieuwste auto's is er al sprake van een zeker mate van zelfstandigheid, zo kan men gealarmeerd worden bij het naderen van een andere bestuurder of een volle witte lijn. Bekende bedrijven zoals Google, Apple en Uber zijn volop bezig met de ontwikkeling van deze autonome wagens. Google werkt samen met verschillende autobouwers, waaronder Audi en Toyota en Apple heeft met hun project 'Titan' al meer dan 50 autonome wagens op de openbare weg rijden [9]. Er is geen ontkennen aan, zelfrijdende auto's hebben een toekomst. Om deze reden hebben wij besloten een miniatuurwagen te maken die in staat is om autonoom een voorgeprogrammeerde weg te volgen. Deze wagen laten we rijden in een miniatuur 'smart city' samen met andere wagentjes met hetzelfde doel: het parcours met succes beëindigen. De taak is volbracht als het wagentje de verkeersregels volgt, het parcours juist interpreteert en geen botsingen veroorzaakt. We maken van de gekregen vrijheid gebruik om het wagentje volledig (in de mate van de gekregen vrijheid) naar onze hand te zetten.

2 Klantenvereisten en ontwerpspecificaties

Een zelfrijdende miniatuurwagen die zich rondbeweegt in een Slimme Stad zal aan bepaalde vereisten moeten voldoen en bepaalde dingen kunnen om zich zonder problemen te kunnen voortbewegen. Deze vereisten hangen vast aan hoe de Slimme Stad er uitziet en wat er van de gebruikers verwacht wordt

2.1 Klantenvereisten

De miniatuur miniatuurwagen moet zich volgens een voorgeprogrammeerde route door een modelstad kunnen voortbewegen waarbij de modelstad bestaat uit negen identieke kruispunten verbonden door straten van één meter lang ihij al dan niet mag door rijden of afslaan. Ook moet de miniatuurwagen een voorligger of obstakel kunnen detectern een grid. Hierbij volgt de miniatuurwagen een zwarte, volle volglijn. Bij de kruispunten is er een stopstreep en hangen er stoplichten op 75 mm hoogte, die gemonteerd zijn op een tafelonderstel van 300 mm hoog. De miniatuurwagen moet stoppen bij de stopstreep en kunnen interpreteren wanneer en en op tijd kunnen stoppen om een botsing te vermijden indien nodig. Ook moet er een grafische gebruikersomgeving zijn waarmee relevante gegevens van de miniatuurwagen kunnen afgelezen worden en een manuele overname kunnen uitgevoerd worden via deze grafische gebruikersomgeving. De manuele overname moet in staat zijn een noodstop uit te voeren of de besturing van de miniatuurwagen over te nemen. De maximale kostprijs van de zelfrijdende miniatuur miniatuurwagen mag niet meer dan 3500 virtuele eenheden zijn.

2.2 Ontwerpspecificaties

De afmetingen van de miniatuurwagen worden beperkt tot 300 mm in de hoogte door de tafelonderstellen bij de kruispunten en 250 mm breedte door de breedte van de baan. De lengte van de miniatuurwagen valt vrij te kiezen, maar het moet wel haalbaar zijn om bijvoorbeeld bochten te nemen. De miniatuurwagen volgt een donkere volglijn op een lichte ondergrond van 25 mm breed. Aan een kruispunt interpreteert het wagentje een verkeerslicht dat zich bevindt op 75 mm boven de grond en knippert aan een frequentie van één Hertz. Het verkeerslicht is gemonteerd aan de voorkant van een tafelpoot waardoor het wagentje hem langs de rechterkant zal moeten detecteren. Indien het verkeerslicht rood is, stopt de miniatuurwagen bij de stopstreep. Deze stopstreep is 50 mm dik en 25 cm lang. Het miniatuurwagentje kan ook voorliggers detecteren via een afstandssensor. Indien het een voorligger detecteert, vertraagt het wagentje of stopt het om zo een botsing te vermijden. Het te volgen traject wordt een week op voorhand bekend gemaakt en kan dan geprogrammeerd worden. De componenten van het prototype mogen verbonden worden via een experimenteerbord. De definitieve versie van de miniatuurwagen moet wel via een printplaat kunnen functioneren. Voor de microcontroller dient gebruik te worden gemaakt van een NI myRIO of Raspberry Pi. Tussen de microcontroller en de motoren dient een motorshield te worden aangebracht, gezien dit een terugloopbeveiliging bevat die beschadiging van de microcontroller voorkomt. Er moet ook een draadloze informatieoverdracht via LabVIEW aanwezig zijn die zich zal uiten op een zelfgemaakte grafische gebruikersomgeving.

3 Ontwerp

3.1 Chassis

De ontwerpruimte werd sterk beperkt door de gelimiteerde materiaaldatabank. De belangrijkste keuzes betroffen de wielen, de motor, de microcontroller, het chassis, het type sensoren, de batterij en het experimenteerbord.

De wielen

Voor de wielen van de miniatuurwagen hebben we gekozen voor de kleinste wielen met een diameter van 32 mm. Het voordeel aan deze wielen is dat ze meer nauwkeurigheid bieden bij het roteren van de miniatuurwagen en minder kracht nodig hebben om rond gedraaid te worden. Daarnaast kunnen kleinere wielen ook sneller ronddraaien, waardoor de elektrische motoren die de wielen aandrijven ook sneller ronddraaien. Dit leidt tot een verhoogde efficiëntie. Het nadeel aan deze wielen is echter wel dat de topsnelheid lager ligt. De wielen komen aan de voorkant, zoals te zien op figuur 1, zodat dit

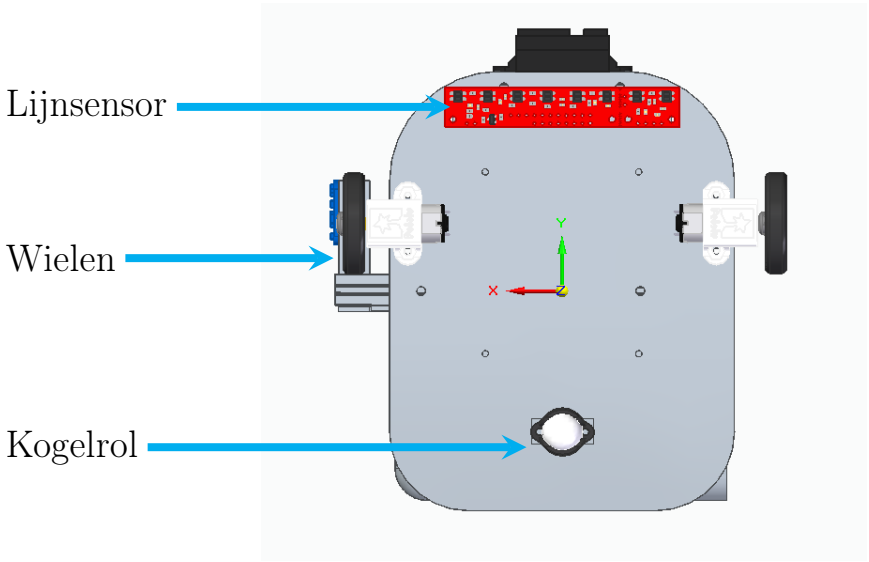
Motor	toeren/min	km/h
100:1 HP	310	0.78
100:1	130	1.87
50:1 HP	590	2.71
30:1	1000	3.56
30:1 HP	450	6.03

Tabel 1: Berekening snelheid voor elke motor voor wielen met diameter 32 mm

het besturen van de miniatuurwagen vergemakkelijkt. De kogelrol zullen we in het midden van achteren plaatsen. Deze zullen we op eenzelfde hoogte plaatsen als de wielen door er plastic plaatjes onder te bevestigen. We positioneren de kogelrol zodanig dat het chassis een minimale hoek maakt met de grond. Op deze manier verzekeren we dat de lijnsensor en de afstandssensor, respectievelijk, parallel en loodrecht staat met het grondoppervlak". De keuze voor deze wielen legt ook beperkingen op aan het type motoren.

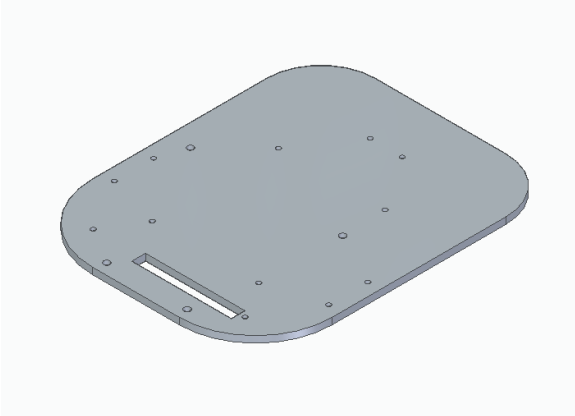
De motor

Wij kozen voor de 30:1 HP motor aangezien deze met een snelheid van ca. zes km per uur het meest compatibel zijn met de gekozen wielen . Deze motor zorgt voor de hoogste topsnelheid met de wielen die we gekozen hadden. Om dit te bepalen hebben we enkele berekeningen uitgevoerd met het aantal toeren per minuut dat de motor maximaal kan draaien en de omtrek van de wielen [5]. Deze resultaten zijn te vinden in tabel 1. Na een testrit bleken de motoren te traag te snel voor onze wagen, het koppel was te laag. Dit heeft ons doen kiezen voor motoren met meer koppel, namelijk de 100:1 motor. Deze wielen en motoren zullen we plaatsen aan de voorkant van onze wagen. De motoren worden aan de onderkant van het chassis bevestigd om zo ons chassis hoger van de grond te krijgen en zo kan op die manier ook de lijnsensor aan de onderkant van het chassis geplaatst worden. Dit alles is te zien op figuur 1.

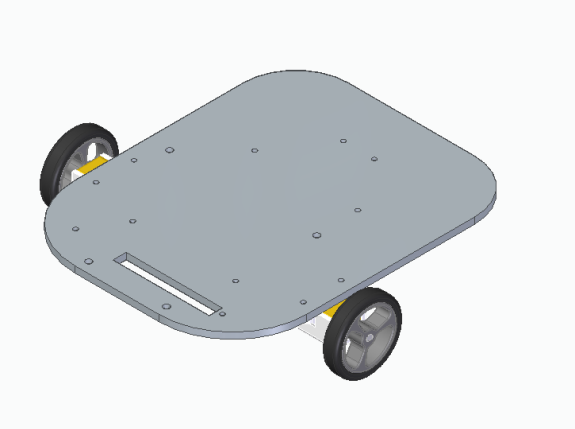


Figuur 1: De lijnsensor is bevestigd tussen de wielen, aan de onderkant van de wagen. Dit zorgt ervoor dat het wagentje nauwkeurig rijdt en sneller correcties kan uitvoeren. De figuur toont de onderkant van de wagen

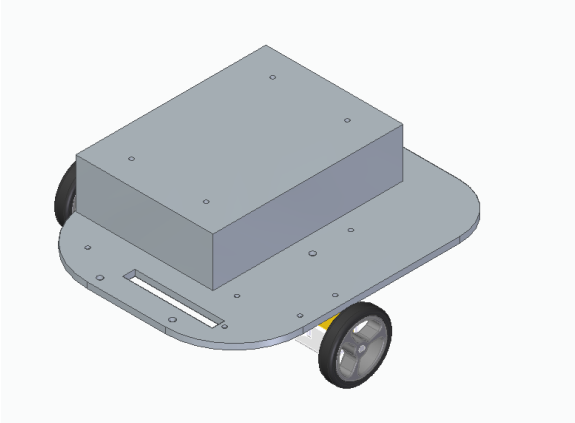
De microcontroller



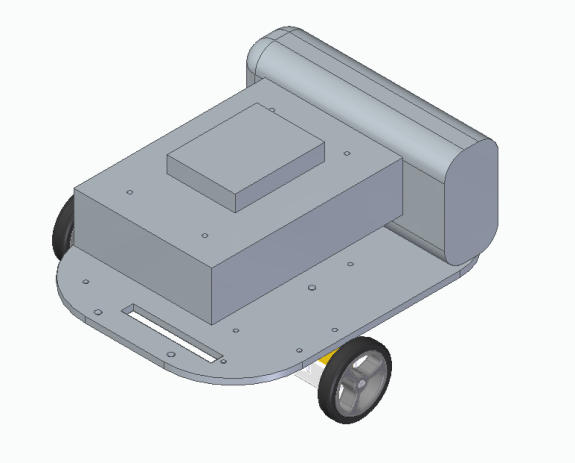
(a) Het zelfontworpen chassis



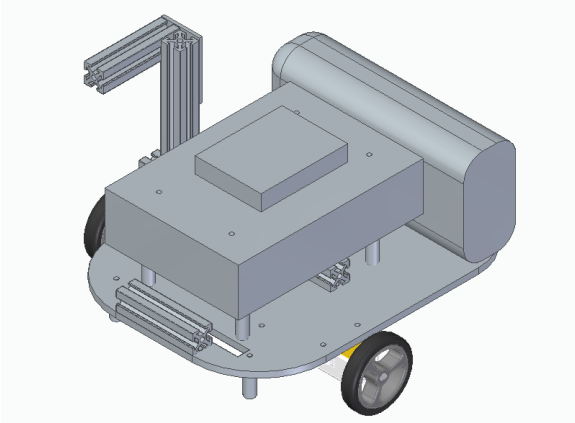
(b) Figuur 2a met wielen en motoren



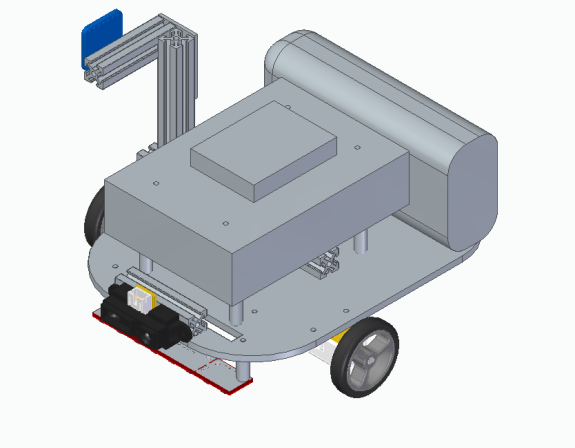
(c) Figuur 2b met de microcontroller, Raspberry Pi



(d) Figuur 2c met de powerbank en het experimenteerbord



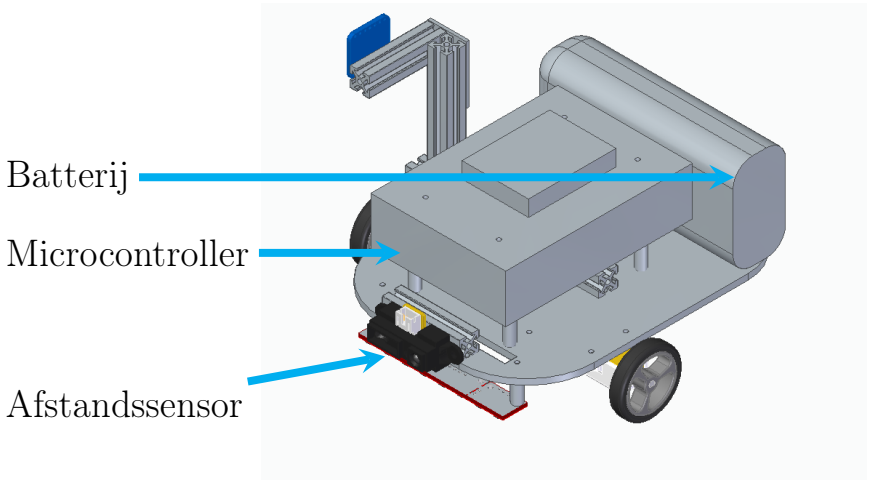
(e) Figuur 2d met de MakerBeam-constructie



(f) Figuur 2e met de sensoren: kleurensensor, afstandssensor en lijnsensor

Figuur 2: De deelfiguren tonen het bouwproces van de miniatuurwagen stap voor stap.

Een Raspberry Pi verbruikt zeer weinig stroom, wat een groot voordeel is wanneer je miniatuurwagentje op een batterij werkt. Daarnaast is het zeer makkelijk om informatie over de werking te vinden, aangezien deze microcontroller zoveel gebruikt wordt [7]. De Raspberry Pi wordt geplaatst op vier afstandsbussen van 15 mm hoog waardoor hij op een platform komt te staan. Dit is te zien op figuur 3. Op deze manier voorzien we voldoende ruimte voor de (plaatsing van de) sensoren, motoren, wielen en Maker Beams die nog op het chassis moeten komen.
Op de Raspberry Pi komen twee kleine experimenteerborden die aan elkaar gelinkt zijn. Achter de Raspberry Pi ter hoogte van onze kogelrol zullen we de powerbank op het chassis plaatsen. Deze past jammer genoeg juist niet onder de Raspberry waardoor hij niet onder het platform kan. De powerbank zorgt met zijn gewicht ervoor dat het massamiddelpunt van de wagen meer naar achteren en lager komt te liggen wat zorgt voor extra stabiliteit. De powerbank is ook makkelijk aan te sluiten op de Raspberry Pi en levert het juiste voltage voor de Raspberry Pi om te functioneren.



Figuur 3: De volledige miniatuurwagen

Het chassis

Het chassis is handmatig ontworpen naar de behoeften van het miniatuurwagentje en zal ge-3D-print worden. Door het chassis zelf te ontwerpen, kunnen we van het standaard model afstappen en ons wagentje er laten uitspringen. Dit chassis zal 140 mm lang, 110 mm breed en 3 mm dik zijn met afgeronde hoeken, zoals te zien op figuur 4. Hiermee bekomen we een chassis waarmee we genoeg ruimte hebben in de lengte om onze microcontroller en onze powerbank op te plaatsen. Vooraan het chassis zal er een gleuf voorzien worden van 6,5 mm breed en 45,8 mm lang voor de verbindingsdraden naar de lijnsensor die onder het chassis hangt, zie figuur 1. Er zullen ook gaten in het chassis bevinden die kunnen gebruikt worden voor het bevestigen van componenten met schroeven en moeren. Het chassis zal aan een dichtheid van 70% ge-3D-print worden zodat het genoeg steun kan bieden.

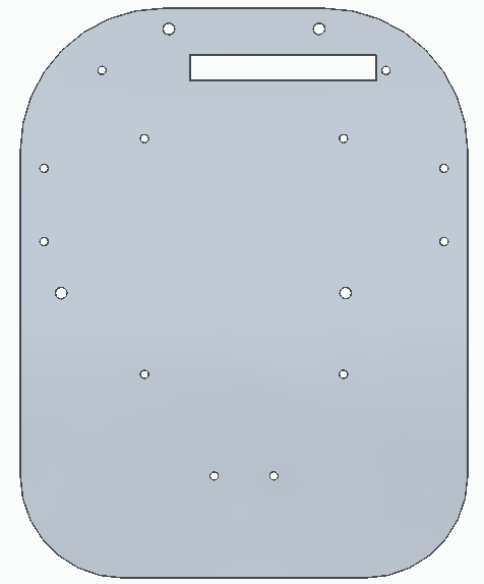
3.2 Sensoren

De kleurensensor

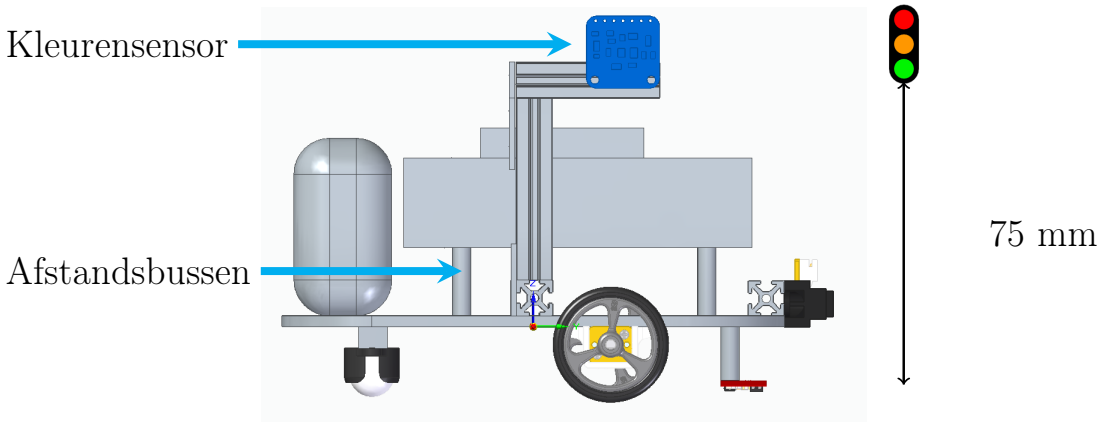
De kleurensensor zal bevestigt worden aan een MakerBeam-staaf op een hoogte van 75 mm van de grond zodat deze het verkeerslicht, die op dezelfde hoogte hangt, kan detecteren. Deze constructie laat toe om de positie van de kleurensensor eenvoudig (en handmatig) bij te stellen, dit is te zien op figuur 5. Deze constructie bestaat uit een MakerBeam die horizontaal in het midden van ons chassis ligt onder de Raspberry Pi met daarop een MakerBeam verticaal omhoog aan het rechter uiteinde. De kleurensensor wordt bevestigd aan de horizontale MakerBeam

De afstandssensor

Voor de afstandssensor hebben we gekozen voor een analoge sensor, opdat we nauwkeurig kunnen meten hoever het wagentje verwijderd is van een obstakel of een andere weggebruiker. Doordat we dit onderscheid kunnen maken moet onze miniatuur miniatuurwagen niet direct stoppen wanneer hij een voorligger detecteert, maar kan hij er voor zorgen dat hij eerst wat trager rijdt vooraleer volledig te stop-



Figuur 4: CAD-model van het zelf-ontworpen-chassis.



Figuur 5: De kleurensensor wordt bevestigd aan een MakerBeam-constructie die toelaat om de sensor optimaal te positioneren. De figuur toont het linkeraanzicht van het wagentje.

pen. De afstandssensor wordt aan de voorzijde van het chassis bevestigd met behulp van een MakerBeam.

De lijnsensor

De lijnsensor wordt gebruikt om de volglijn te detecteren. Hierbij hebben we een digitale sensor gekozen omdat deze compatibel is met de Raspberry Pi microcontroller en de sensor geen onderscheid moet kunnen maken tussen de lijnsoort. Deze wordt aan de voorkant onder het chassis bevestigd met behulp van afstandsbuizen zodat deze de volglijn als eerst detecteert en we zo sneller correcties kunnen uitvoeren. Aangezien de sensor dicht bij de wielen ligt, zal het wagentje ook nauwkeuriger bewegen.

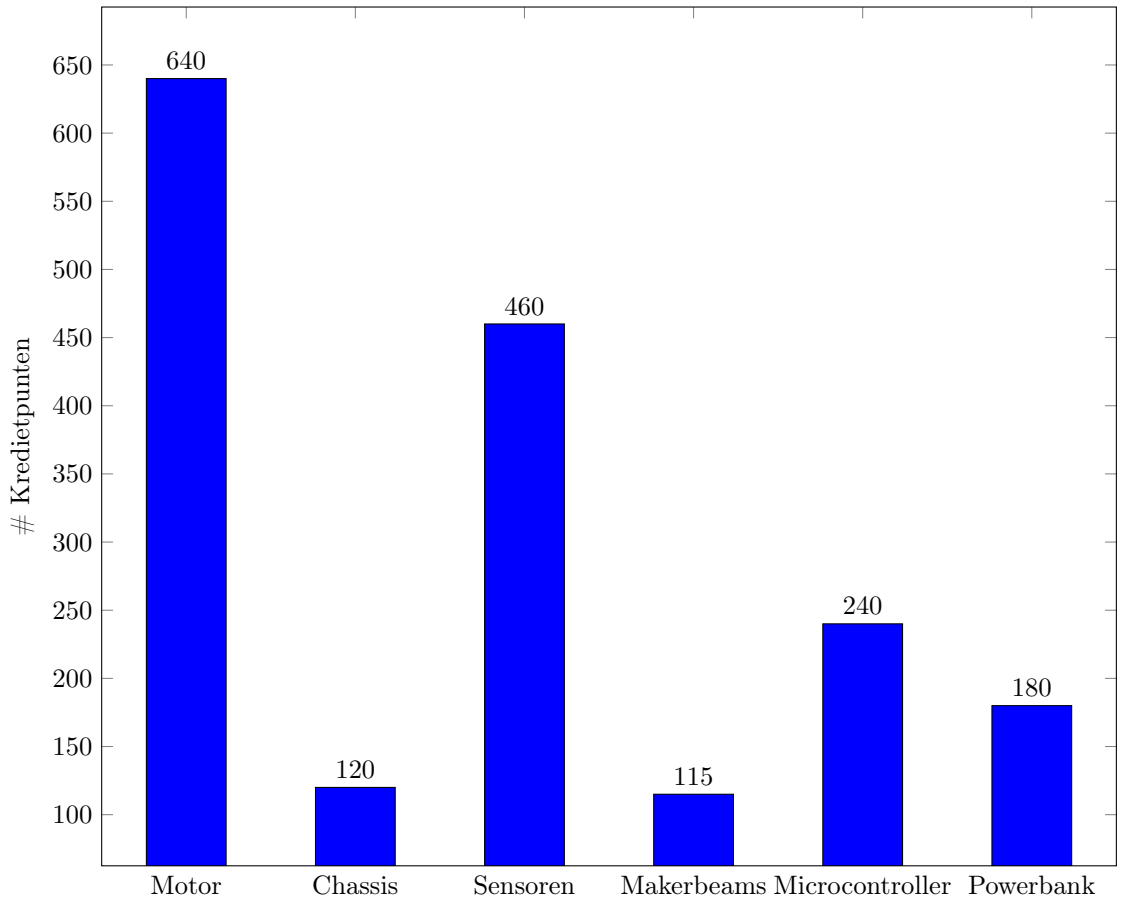
Het volledige ontwerp met de lijnsensor is te zien in figuur 3.

4 Evaluatie

4.1 Planning

Tot en met week 5 is alles verlopen zoals gepland en vastgelegd in de Gantt-grafiek. De documenten met betrekking tot de planning zijn afgewerkt net zoals het ontwerp van de chassis en het CAD-model. Het

bestellen van de onderdelen vond later plaats dan verwacht, vanwege de plaatsing bij de bieding. Dit heeft ons met een achterstand doen starten aan het programmeren en assembleren. Het team was van plan om te beginnen met de assemblage in week 6, echter wordt dit een week uitgesteld vanwege een probleem met de microcontroller. Het programmeren van de sensoren verliep vlot, enkel het programmeren in verband met de aansturing van het wagentje neemt meer tijd in dan ingerekend, namelijk twee weken. Dit wordt dan wel gecompenseerd met het maken van het CAD-model dat een week op voorhand klaar was, waardoor we nog altijd op schema zitten. Ook het schrijven van het verslag verloopt volgens schema. De tijd voorzien tijdens de lessen wordt efficiënt gebruikt, waardoor het werk buiten de lessen wordt beperkt tot een minimum.



Tabel 2: De kost van de motor is bijna 20% van het volledige budget doordat een tweede motor aangekocht moest worden. De tabel toont de absolute kosten van de grootste categorieën.

4.2 Financiën

Omdat we de laatste keuze hadden bij het bestellen, liepen we achter op onze planning, onze financiële situatie kende hier dan weer voordelen door. We zijn gestart met een budget van 3500 kredietpunten, dat werd gereduceerd tot een tegoed van 1768 na het plaatsen van een eerste bestelling. In deze order werden alle grote onderdelen en zaken die we zeker nodig hadden besteld. Aan onze eerste levering ontbraken nog Maker Beams en afstandsbussen. Onze tweede bestelling komt neer op 91 kredietpunten, waardoor we een budget van 1677 kredietpunten overhouden. Door de vele keuzemogelijkheden, waaronder vorm en dikte, werd ervoor gekozen het chassis zelf te ontwerpen en te 3D-printen. Dit is vermoedelijk een grote kost die werd geschat op 500 kredietpunten. Na het printen bleek het chassis slechts 120 kredietpunten te kosten. Samen met de kost voor de derde bestelling van 340 kredietpunten, brengt dit ons op een totaal van 1217 kredietpunten. Bij deze derde bestelling werden nieuwe 100:1 motoren gekocht na het vaststellen dat de 30:1 HP motoren niet genoeg koppel hadden. Dit zorgt ervoor dat de motoren de grootste kost

van de wagen zijn. Onze financiële toestand zorgt ervoor dat we in een comfortabele situatie zitten en kunnen we het ons veroorloven om fouten te maken en risico's te nemen. De kosten van de belangrijkste onderdelen zijn terug te vinden in tabel 2

4.3 Software

De manuele overname

De manuele overname is als volgt geïmplementeerd: De LabVIEW-code runt continu op een pc terwijl de Python-code dat ook doet op de Raspberry Pi. Wanneer de startknop wordt ingedrukt, wordt de overname ingeschakeld. Het programma komt in een lus terecht en controleert de berichten die binnenkomen. Afhankelijk van de binnenkomende berichten gaat het wagentje vooruit, achteruit, naar links of naar rechts. Indien er m.b.v. de overname kruispunten genomen worden, kan men handmatig bij het stopcommando het huidige kruispuntnummer meegeven. Dit is optioneel, dus als er geen nummer wordt meegegeven, wordt het kruispuntnummer niet gewijzigd. Indien men op geen enkele knop drukt blijft de overname actief, maar worden de motoren niet aangedreven. Indien men op de stopknop drukt, wordt de overname afgesloten tot het eventueel terug wordt opgestart.

De lijnsensor

Bij de lijnsensor wordt eerst de ruwe data van de acht sensoren ingelezen. Deze data wordt uitgedrukt in seconden. Bij weinig reflectie (een donker oppervlak) zijn deze waarden hoog, bij veel reflectie (een wit oppervlak) zijn deze waarden laag.

Omdat deze data niet praktisch is om mee te werken, wordt deze herschaald tot een waarde van 0 tot 1000, waarbij lagere waarden voor lichtere oppervlakken staan. Dit gebeurt aan de hand van een gekalibreerde minimum- en maximumwaarde van de seconden. Deze twee waarden worden op voorhand bepaald door de robotwagen over de lichte vloer te laten rijden en vervolgens over de zwarte volglijn, waarbij het minimum en maximum van de uitgelezen data genomen wordt. Nadat de data herschaald is, definiëren we de positie van het voertuig met volgende formule:

$$\frac{0 \cdot waarde_0 + 1000 \cdot waarde_1 + 7000 \cdot waarde_7}{waarde_0 + waarde_1 + .. + waarde_7}$$

Hierbij staat $waarde_0$voor de herschaalde waarde van de eerste, meest linkse sensor en $waarde_7$ voor de herschaalde waarde van de meest rechtse sensor. De breedte van de lijn is 25 mm, wat ongeveer overeenkomt met twee sensoren die een zwart oppervlak zien, terwijl de overige zes sensoren zich boven het lichte oppervlak bevinden. Aangezien we zouden willen dat de lijn zich onder het midden van de robotwagen bevindt, zouden we graag $waarde_3$ en $waarde_4$ gelijk hebben aan 1000 en de andere waarden gelijk aan nul. Daardoor wordt onze te bekomen positie dus

$$\frac{3000 \cdot 1000 + 4000 \cdot 1000}{1000 + 1000} = 3500.$$

We definiëren nu de fout als de huidige positie min de te bekomen positie, 3500. Stel dat in de huidige positie de lijn rechts onder het wagentje ligt, dan zal volgens bovenstaande formule de huidige positie een grotere waarde aannemen dan 3500. De fout is dus positief en de wagen moet iets meer naar rechts rijden om dit te corrigeren. Daarom corrigeren we de linkse motorsnelheid met een plus $c \cdot$ fout, en de rechtse motorsnelheid met min $c \cdot$ fout, waarbij c een constante is die door testen bepaald wordt.

Stel nu dat de lijn te links ligt, dan zal de huidige positie lager zijn dan 3500 en is de fout negatief. De linkse motor wordt gecorrigeerd met plus $c \cdot$ fout, maar de fout is negatief en resulteert dus in een netto negatieve correctie van de linkse motorsnelheid. De rechtse motorsnelheid zal dus een netto positieve correctie hebben. De wagen zal dus naar links rijden, wat precies is wat we wilden bereiken.

De afstandssensor Onze microcontroller, een Raspberry Pi, kan enkel digitale output uitlezen. Daarom was het noodzakelijk om een analoog-digitaalomzetter (ADC) in de schakeling te verwerken. Deze communiceert met de microcontroller door middel van de 'Serial Peripheral Interface' (SPI). Als we data van de sensor willen verkrijgen, sturen we een bericht naar de ADC, waarop deze ons een antwoord geeft. Dit antwoord is een digitaal antwoord bestaande uit 2 bytes. Door middel van volgende Python-code kunnen we uit dit bericht 1 enkele waarde bekomen:

```
# Antwoord (2 bytes) omzetten naar enkele waarde
adc = 0
for n in antwoord:
        adc = (adc << 8) + n

# Laatste bit uit antwoord is geen ADC-waarde
adc = adc >> 1
```

Uit deze waarde verkrijgen we dan op basis van onderstaande Python-code respectievelijk het output-voltage van de afstandssensor zelf en de afstand tot het voorliggende object in centimeter:

```
# Voltage berekenen op basis van de digitale waarde
voltage = (3.3 * adc)/1024

# De factor 27 door 2de grafiek van de bijgeleverde documenten van de afstandssensor
if voltage == 0:
        return 100
else:
        afstand = 27/voltage
        return afstand
```

Hierbij is `3.3` de spanning van de ADC. De deler in dezelfde lijn code, `1024`, is de resolutie van de ADC in bits, deze is 10 bit zo is $2^10 = 1024$. Het gevalonderscheid werd toegevoegd zodat ons programma niet zo crashen indien de afstandssensor 0 V zou teruggeven. We gebruikten deze code als functie in het hoofdprogramma (vandaar de return statements in de code hierboven), hierbij werd per keer dat het robotwagentje 20 keer de lijn volgde de afstandssensor uitgelezen en werd er gecontroleerd of de afstand niet kleiner was dan 15 cm. Indien dit toch het geval was, stopte het wagentje met rijden en werd de afstandssensor continu uitgelezen. Indien de afstand groter werd dan 20 cm begon het robotwagentje terug te rijden.

5 Besluit

Met een budget van 3500 kredietpunten hebben we alle onderdelen kunnen aankopen om een functionerende miniatuurwagen te kunnen bouwen. Van dit budget bleven 1217 kredietpunten over op het einde. Dit bedrag werd besteed aan voornamelijk wielen, motoren, een microcontroller, een powerbank, sensoren, een chassis en MakerBeam-onderdelen. Uit het financieel rapport bleek dat de motoren de grootste kost zijn en daarna de sensoren.

Uit dit project concluderen we dat er veel mogelijkheden zijn om een miniatuurwagen te maken en dat er veel keuzes gemaakt moeten worden. Dit project toont op kleine schaal aan dat een smart city en meer specifiek, zelfrijdende wagens minder utopie en meer werkelijkheid worden. Zelfrijdende wagens kennen zeker een toekomst en hebben hun nut al meermaals bewezen. De weg naar een wereld met autonome wagens is lang, maar wordt stilaan meer en meer bereden.

Referenties

[1] taalunie. https://taaladvies.net/taal/advies/vraag/1030/zogezegd_zogenaamd/.

[2] 3bplus. Wat is een smart city (slimme stad): een introductie. https://3bplus.nl/wat-een-smart-city-slimme-stad-een-introductie/, laatste wijziging 17 juni 2016.

[3] Amplifon. Geregistreerde criminaliteit. amplifon.com/nl-be/nieuws-en-blog/te-veel-lawaai-op-straat, laatste wijziging 4 maart 2020.

[4] Pieter Ballon. *Smart Cities*. Lannoo, 2016.

[5] Lulu's blog. https://lucidar.me/en/unit-converter/revolutions-per-minute-to-kilometers-per-hour/.

[6] TU Delft. Autonome auto's: de volgende grote technologische revolutie. https://www.tudelft.nl/ewi/actueel/nodes/autonome-autos-de-volgende-grote-technologische-revolutie.

[7] Micha den Heijer. Wat is een raspberry pi? hoe werkt het? https://prgrmmr.nl/wat-is-een-raspberry-pi-hoe-werkt-het.html: :text=Tenlaatste wijziging 29 februari 2020.

[8] Benjamin Maveau en Kevin Truyaert. Opgave teamopdracht probleemoplossen en ontwerpen 2. 2020-2021.

[9] Tom Van Gurp. Deze 30 bedrijven bestormen de markt voor zelfrijdende auto's. https://www.zelfrijdendvervoer.nl/mobiliteit/2016/04/26/dertig-fabrikanten-in-race-voor-marktaandeel-zelfrijdende-autos/?gdpr=accept, laatste wijziging 26 april 2016.

[10] kentekencheck.nu. Wat zijn de voor- en nadelen van zelfrijdende auto's? https://www.kentekencheck.nu/wat-zijn-de-voor-en-nadelen-van-zelfrijdende-autos/: :text=Zelfrijdendelaatste wijziging 11 december 2019.

[11] milieudefensie. Dit zijn de oplossingen voor luchtvervuiling. https://milieudefensie.nl/recht-op-gezonde-lucht/dit-zijn-de-oplossingen-voor-luchtvervuiling.

[12] Primestone. Advantages and disadvantages of smart cities. https://primestone.com/en/advantages-and-disadvantages-of-smart-cities/, laatste wijziging 2 april 2020.

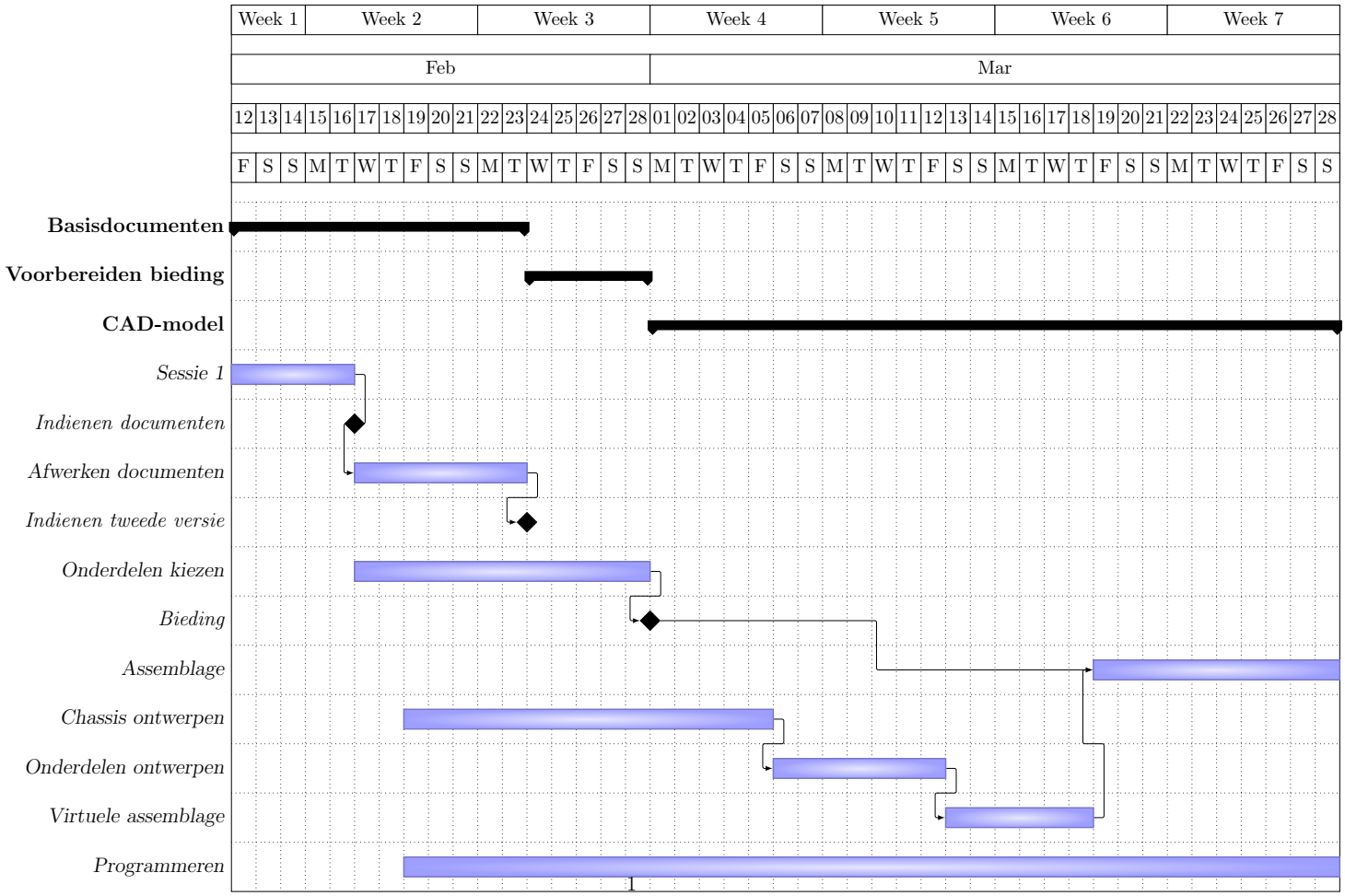
[13] Vlaamse Regering. Geconnecteerde en geautomatiseerde mobiliteit in vlaanderen. https://www.ewi-vlaanderen.be/sites/default/files/conceptnota_-_geconnecteerde_en_geautomatiseerde_mobiliteit_in_vlaanderen.pdf.

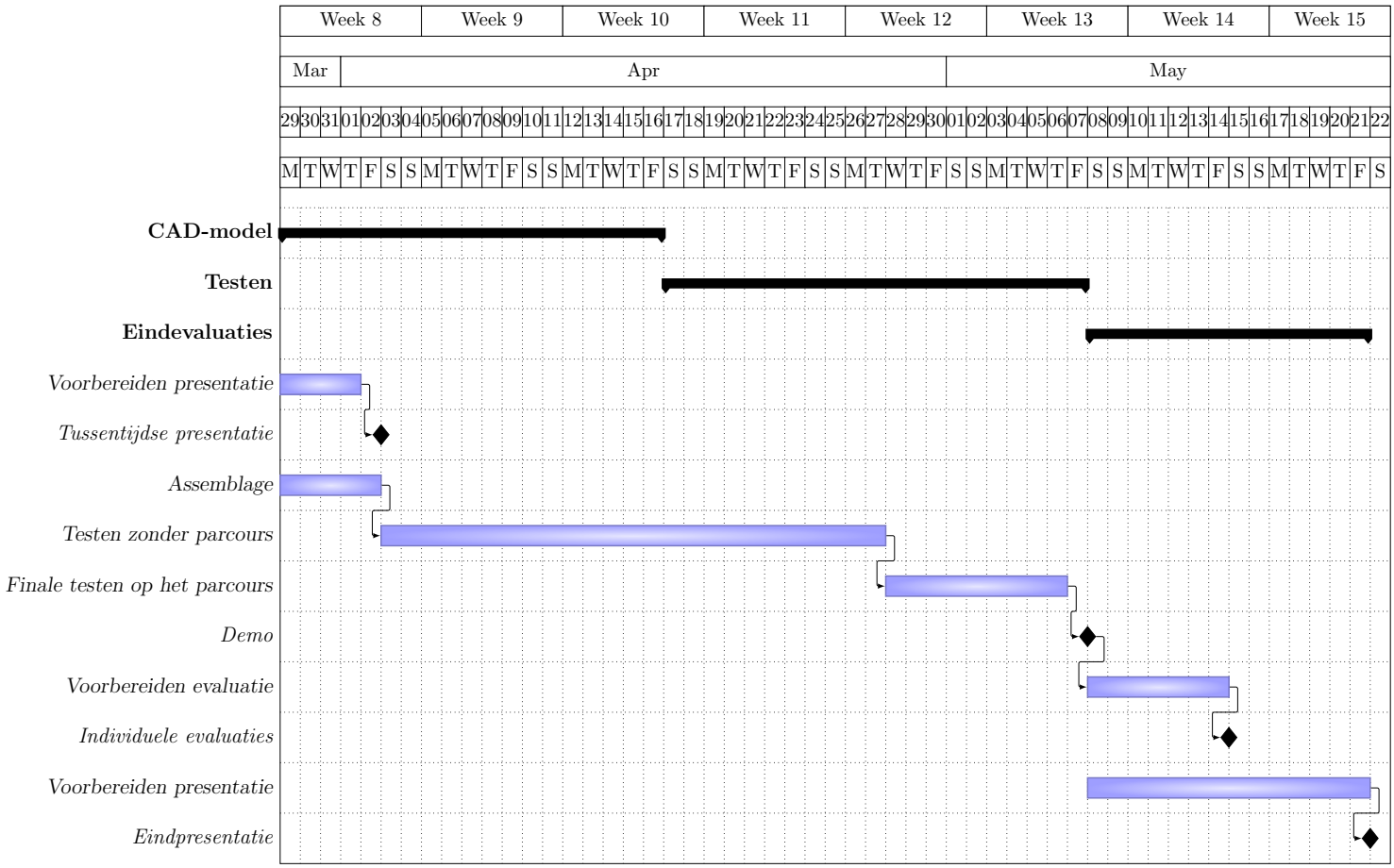
[14] StatistiekVlaanderen. Geregistreerde criminaliteit. https://www.statistiekvlaanderen.be/nl/geregistreerde-criminaliteit-0, laatste wijziging 27 augustus 2020.

[15] synoniemen.net. https://synoniemen.net/index.php?zoekterm=computerkraker.

Appendices

A Gantt-grafiek en taakstructuur


Week 1
Week 2
Week 3
Week 4
Week 5
Week 6
Week 7
Feb
Mar
12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S
Basisdocumenten
Voorbereiden bieding
CAD-model
Sessie 1
Indienen documenten
Afwerken documenten
Indienen tweede versie
Onderdelen kiezen
Bieding
Assemblage
Chassis ontwerpen
Onderdelen ontwerpen
Virtuele assemblage
Programmeren


Week 8
Week 9
Week 10
Week 11
Week 12
Week 13
Week 14
Week 15
Mar
Apr
May
29 30 31 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22
M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S
CAD-model
Testen
Eindevaluaties
Voorbereiden presentatie
Tussentijdse presentatie
Assemblage
Testen zonder parcours
Finale testen op het parcours
Demo
Voorbereiden evaluatie
Individuele evaluaties
Voorbereiden presentatie
Eindpresentatie

Tabel 1: Taakstructuur team Safety First

Code	Taak	Status
1	Inwerken	OK
1.1	Documenten op Toledo lezen	OK
1.2	Handleiding P&O2	niet OK
1.3	Overleggen en plannen	OK
1.4	Brainstorm	OK
1.5	Klantenvereisten	OK
1.6	Ontwerpspecificaties	OK
1.7	Verantwoordelijkheidsstructuur	OK
1.8	Teamkalender	OK
1.9	Gantt-grafiek	OK
2	CAD-model	OK
2.1	Chassisontwerp	OK
2.2	Basisonderdelen	OK
2.2.1	Wielen	OK
2.2.2	Powerbank	OK
2.2.3	Breadboard	OK
2.3	Extra onderdelen	OK
2.3.1	afstandsbus	OK
2.3.2	Makerbeam hoekverbinding	OK
2.3.3	Makerbeam verbindingen	OK
2.4	Assemblage	OK
2.5	Technische tekeningen	niet OK
2.6	Stuklijst	niet OK

Tabel 2: Taakstructuur team Safety First

Code	Taak	Status
3	Programmeren	OK
3.1	LabVIEW of Python	OK
3.1.1	Bocht naar rechts	OK
3.1.2	Bocht naar links	OK
3.1.3	verkeerslichtinterpretatie algoritme	OK
3.1.4	Lijnvolgalgoritme	OK
3.1.5	Afstandsinterpretatiealgoritme	OK
3.1.6	Motoraandrijving	OK
3.1.7	Starten	OK
3.1.8	Stoppen	OK
3.1.9	Grafische interface met besturingsmogelijkheden	OK
3.1.10	Noodstop	OK
4	Testen	OK
4.1	Testen zonder parcours	OK
4.1.1	Kleursensor	OK
4.1.2	Lijnsensor	OK
4.1.3	Bocht naar links	OK
4.1.4	Bocht naar rechts	OK
4.1.5	Kruispunt oversteken	OK
4.1.6	Stoppen	OK
4.1.7	Vertrekken	OK
4.2	Test op parcours	OK
4.2.1	Stoppen aan rood licht	OK
4.2.2	Bochten nemen	OK
4.2.3	Doorrijden aan groen licht	OK
4.2.4	Volledig uitgestippeld parcours afleggen	OK
5	Rapportering	OK
5.1	Tussentijds verslag	OK
5.2	Tussentijdse presentatie	OK
5.3	Eindpresentatie	OK
5.4	Eindverslag	OK

B Budgetmanagement

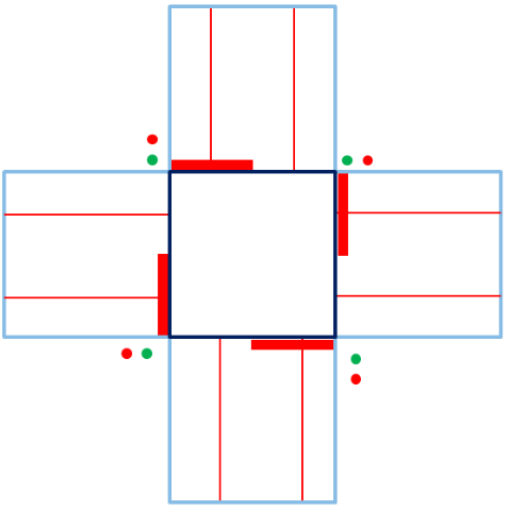
Onderdeel	**Aantal**	**Prijs**
Micro Metal Gear Motor 100:1	2	320
Micro Metal Gear Motor 30:1 HP	2	320
Raspberry Pi	1	240
Powerbank	1	180
Dual Drive DRV8833	1	70
Wiel 32x7 mm zwart	2	70
Ball Caster	1	60
Optische afstandsensor (analoog)	1	160
TCS34725 Kleur sensor BOB	1	15
QTR-8RC digitale reflectie sensor array	1	150
MakerBeam Hoekverbinding 90°	2	15
MakerBeam profiel 40 mm	1	20
MakerBeam profiel 60 mm	3	60
MakerBeam profiel 100 mm	1	20
Micro metal gear motor beugel	2	50
Breadboard Tiny	2	80
ADC MCP3002-I/P	1	40
Motorschield IC	1	70
Wire to board socket	4	20
Printplaat	1	50
Male headers 10	3	15
Afstandsbus 15x5 mm nylon	2	1
Afstandsbus 20x5 mm nylon	4	2

C Klantenvereisten

De klant wenst een miniatuurwagen die autonoom kan rondrijden volgens een voorgeprogrammeerde route in een modelstad. Hierbij volgt de wagen straten via een volglijn. Daarnaast kan het wagentje voorliggers of obstakels detecteren en stoppen indien nodig, om botsing te vermijden. De klant wenst ook dat het wagentje verkeerslichten kan interpreteren bij kruispunten. Indien het verkeerslicht rood is, dient de wagen bij de stopstreep te stoppen. Indien het verkeerslicht groen is, rijdt de wagen door. Het te volgen traject wordt vooraf geïmplementeerd. De wagen kan afslaan op een kruispunt indien dit in de vooraf beschreven route stond. De maximale kostprijs van dit miniatuurwagentje is 3500 virtuele eenheden. De klant wenst een grafische interface waarmee de relevante gegevens van dit wagentje kunnen uitgelezen worden vanop afstand. Een manuele overname van de wagen moet ook mogelijk zijn, zij het uitvoeren van een noodstop, zij de besturing overnemen.

D Ontwerpspecificaties

De klant wenst een zelfrijdend wagentje dat zich volgens een voorgeprogrammeerde route door een modelstad beweegt. De modelstad bestaat uit negen identieke kruispunten opgesteld in een vierkant die zich op een afstand van een meter van elkaar bevinden. Figuren 6 en 7 tonen een voorbeeld van een van die kruispunten.



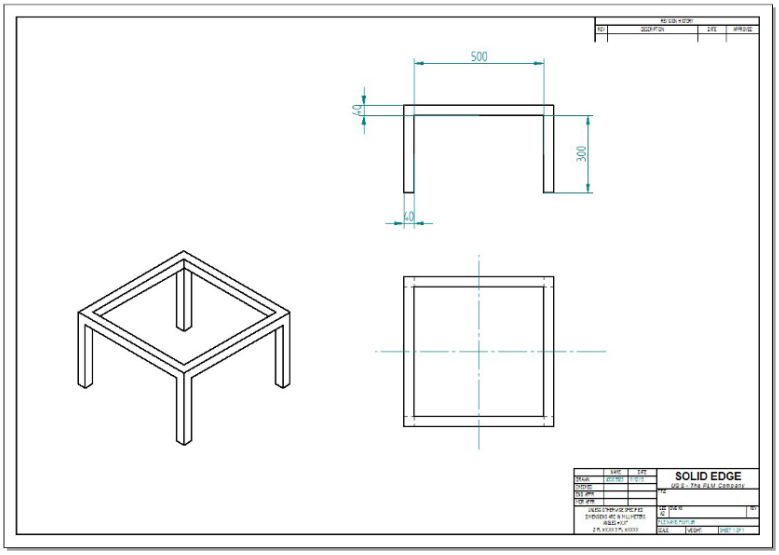
Figuur 6: Bovenaanzicht van een kruispunt (figuur ontleend aan [8])

Het wagentje volgt hierbij een lijn van 25 mm dik aan de hand van een reflectiesensor, dit zijn de dunne rode lijnen op figuur 6. Deze volglijn is een donkere lijn op een heldere ondergrond of een heldere lijn op een donkere ondergrond. De lengte van een te volgen straat is 1 meter en de breedte ervan bedraagt 0.5 meter. Op deze baan bewegen de miniatuurwagens zich rechts in de heenrichting en links in de terugrichting. De maximale, totale breedte van het wagentje bedraagt dus 25 cm. Aan een kruispunt interpreteert het wagentje een rood-groen verkeerslicht. De verkeerslichten zijn gemonteerd op een tafelonderstel (zie figuur 7). Aangezien de hoogte hiervan 300 mm is, is dit ook de maximale hoogte van het te bouwen wagentje. Het midden van het verkeerslicht bevindt zich op 7.5 cm boven de grond. Zoals te zien op de technische tekening **??** is dit ook de plaats waar de kleurled zich in het verkeerslicht bevindt. Het verkeerslicht is gemonteerd aan de voorkant van de tafelpoot, zodat de wagen het verkeerslicht langs de rechterkant moet detecteren, zoals ook figuur 6 suggereert. Indien het verkeerslicht rood is, stopt het miniatuurwagentje bij de stopstreep. Deze stopstreep is 50 mm dik en 25 cm lang, zoals we kunnen afleiden uit de rode dikke lijnen in figuur 6. Indien het verkeerslicht groen is, rijdt het wagentje door of slaat het af, naargelang de gevraagde voorgeprogrammeerde route.

Het wagentje kan ook voorliggers detecteren via een afstandssensor. Indien het een voorligger detecteert, vertraagt het wagentje of stopt het om zo een botsing te vermijden. Het te volgen traject wordt

een week op voorhand bekend gemaakt en kan dan al geprogrammeerd worden. De componenten van het prototype mogen verbonden worden via een breadboard. De definitieve versie van het wagentje moet wel via een printplaat kunnen functioneren. Voor de microcontroller dient gebruik te worden gemaakt van een NI myRIO of Raspberry Pi. Tussen de microcontroller en de motoren dient een motorshield te worden aangebracht, gezien dit een terugloopbeveiliging bevat die beschadiging van de microcontroller voorkomt. Het wagentje haalt zijn energie uit een batterij. De maximale kostprijs van het prototype bedraagt 3500 virtuele eenheden.

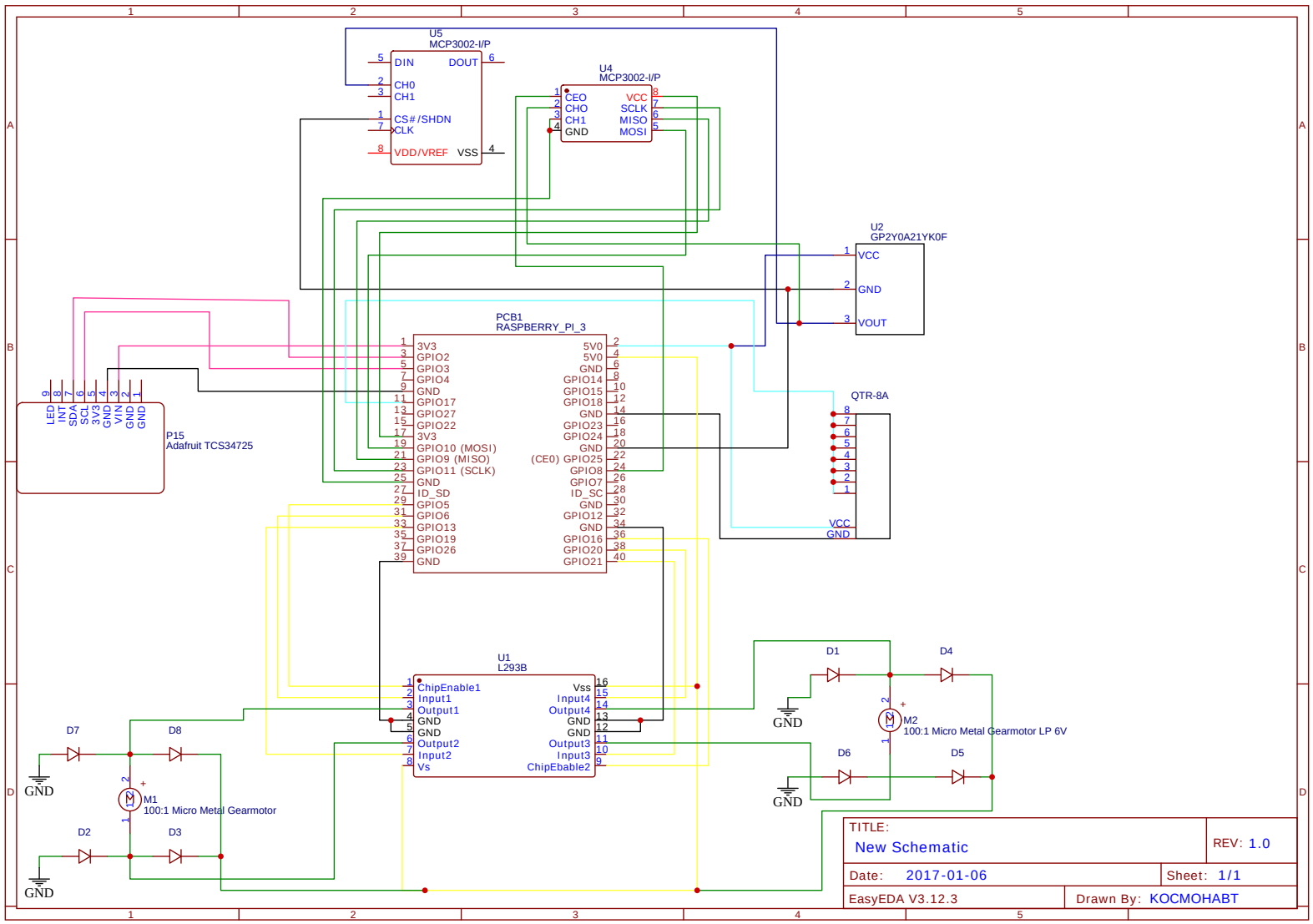
De klant wenst ook dat er een draadloze informatieoverdracht is tussen het wagentje en een computer op afstand. Deze draadloze informatieoverdracht verloopt via LabVIEW en er is ook een grafische interface beschikbaar. Via deze interface kan een noodstop worden uitgevoerd of de volledige besturing van het miniatuurwagentje overgenomen worden.



Figuur 7: Technische tekening van een kruispunt (figuur ontleend aan [8])

E Circuit diagram

Circuit diagram



TITLE: New Schematic		REV: 1.0
Date: 2017-01-06		Sheet: 1/1
EasyEDA V3.12.3	Drawn By: KOCMOHABT	