2018•2019

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: energie

Masterthesis

Open source software voor objecttracking met behulp van markers: een vergelijkende studie

PROMOTOR :

Prof. dr. ir. Eric DEMEESTER

BEGELEIDER :

ing. Martijn CRAMER

BEGELEIDER :

ing. Peter AERTS

Andries Kelchtermans

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: energie, afstudeerrichting automatisering

Gezamenlijke opleiding UHasselt en KU Leuven

2018•2019

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: energie

Masterthesis

Open source software voor objecttracking met behulp van markers: een vergelijkende studie

PROMOTOR :

Prof. dr. ir. Eric DEMEESTER

BEGELEIDER :

ing. Martijn CRAMER

BEGELEIDER :

ing. Peter AERTS

Andries Kelchtermans

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: energie, afstudeerrichting automatisering

UHASSELT KU LEUVEN

Woord vooraf

Voor de opleiding industrieel ingenieur is het zeer belangrijk om over veel kennis te beschikken, maar is het nog belangrijker om deze kennis in de praktijk om te zetten. Dit aspect van de opleiding zit voor een groot deel verwerkt in de masterproef. Hierbij zullen de studenten al hun vergaarde kennis toepassen op één groot project. Dit project is volledig vrij om te kiezen zolang het aan bepaalde eisen voldoet. Als student met een bachelor diploma industrieel ingenieur elektromechanica en master afstudeerrichting energie-automatisatie is er gekozen voor een thema betreffende robotica. Met als onderwerp het robuust tracken van objecten en personen voor drie casussen. Dit onderwerp focust zich voornamelijk op computervisie, dat op dit moment een grote groei kent in de industrie. Door middel van deze masterproef zal ik als student meer inzicht verwerven in computervisie en zijn toepassingen.

Aangezien deze masterproef deel uitmaakt van de opleiding, is er voor dit onderzoek nog ondersteuning door verschillende personen. De eerste persoon die ik voor de vele ondersteuning wil bedanken is mijn promotor prof. dr. ir. Eric Demeester. Hij hielp met het zetten en afbakenen van het doel van deze masterproef en hielp bij moeilijke momenten om knopen door te hakken en verstandige keuzes te maken op knelpuntmomenten. Ook wil ik graag mijn twee begeleiders ing. Martijn Cramer en ing. Peter Aerts bedanken. Enerzijds voor al hun inzichten en anderzijds voor mij telkens te helpen bij allerlei soorten uitdagingen gaande van softwareproblemen tot praktische problemen. Als laatste zou ik ook graag alle andere medewerkers van het onderzoekscentrum ACRO bedanken voor hun hulp en advies.

Inhoudsopgave

Lijst van tabellen

Lijst van figuren

Verklarende woordenlijst

Tracking: het volgen van het traject van een object en mogelijk hiervan de toekomstige locatie(s) voorspellen.

Opensource: (of open bron) betekent dat er vrije toegang is tot het bronmateriaal.

Features: de features in een frame/afbeelding hebben betrekking op de kenmerken en eigenschappen van de objecten die een duidelijke onderscheidende waarde hebben.

Real-time: een programma is real-time wanneer de uitvoertijd van het programma sneller is dan de toegelaten tijd. In het geval voor visie zijn er 40 beelden per seconde gewenst om voor de mens de frames te zien als een video. Dus in dit geval dient de reactie- en uitvoertijd korter te zijn dan 1/40ste seconde.

Ground truth: verwijst naar de informatie die rechtstreeks uit observatie(s) is/zijn behaald. Bij detectie kan dit bijvoorbeeld zijn dat er 20 ballen in een doos liggen, dit is dan de ground truth, en kan gebruikt worden om te vergelijken met hoeveel de detector er detecteert.

Kalibreren/ kalibratie: het vergelijken van een systeem (bv. een camera) met de standaard om zo het systeem te kunnen afstellen zodat die overeenkomstige resultaten bekomt. Deze standaard zal een kalibratieplaat zijn die gekend is. Door deze te detecteren en het beeld te vergelijken met de gekende afmetingen van de plaat is het mogelijk om de intrinsieke en extrinsieke parameters te schatten.

Keypoints: keypoints zijn punten die een essentiële rol spelen voor de detectie aangezien zij ten opzichte van hun omgeving een contrast vertonen en omdat zij op robuuste wijze detecteerbaar zijn in diverse afbeeldingen zoals bijvoorbeeld hoeken.

Permutatiematrix: is een vierkante matrix die precies één waarde in elke rij en elke kolom heeft en waarbij alle overige elementen gelijk zijn aan nul.

Padplanning (bij mobielen robots): is het proces dat men gebruikt om een pad te construeren van een beginpunt tot een eindpunt aan de hand van een volledige, gedeeltelijke of dynamische kaart.

Cropwindow(s): een venster op basis waarvan een afbeelding wordt bijgesneden om zo overbodige gebieden te verwijderen en enkel de essentie over te houden.

Abstract

Vandaag de dag kent tracking, het volgen van objecten of personen in beeldmateriaal, een sterke opmars met talloze toepassingen. Deze toepassingen bezitten elke hun eigenaardigheden, waardoor geen algemene oplossing geboden kan worden voor ieder trackingsprobleem. Deze masterthesis focust zich daarom op het oplossen van drie casussen binnen de ACRO-onderzoeksgroep, namelijk de kalibratie van het bewegingsmodel van een gidsrobot, de tracking van objecten tijdens mens-robotsamenwerking en de tracking van objecten en personen door een mobiele robot.

Allereerst werd een uitgebreide literatuurstudie uitgevoerd naar principes en algoritmes voor objectdetectie, -tracking en data-associatie. Hieruit is gebleken dat detectie een computationeel intensief proces is en daarom is besloten dat hiervoor een benadering nodig is. Daarom werd voor zowel casus één als twee markerdetectie van OpenCV toegepast. Hiervan is een vergelijkende nauwkeurigheidsstudie uitgevoerd.

Voor casus één is dit van toepassing omdat de exacte pose en locatie nodig is, uit de studie blijkt dat het ChArUco-paneel hiervoor het meest geschikt is. Bij casus twee worden enkel objecten getrackt, maar omdat de detectie te intensief is, wordt ook hier markerdetectie met behulp van de ArUco markers toegepast. Ten laatste is het doel van casus drie om objecten te detecteren en tracken in 3D met een mobiele camera. Hiervoor zijn enkel voorstellen van aanpak voor het trackingsprobleem gemaakt.

Abstract in English

Nowadays tracking, the following of objects or persons in visual material, is on the rise with numerous applications. These applications each have their own peculiarities, which means that no general solution can be offered for every tracking problem. This master thesis therefore focuses on solving three cases within the ACRO research group, namely the calibration of the movement model of a guide robot, the tracking of objects during human-robot collaboration and the tracking of objects and persons by a mobile robot.

First of all, an extensive literature study was carried out into principles and algorithms for object detection, tracking and data association. This has shown that detection is a computationally intensive process and therefore it has been decided that an approach is needed. Therefore, the OpenCV mark detection was used for both case one and two. A comparative accuracy study has been carried out on this subject.

The study shows that the ChArUco panel is the most suitable panel for case one because the exact pose and location is needed. In case two, only objects are traced, but because the detection is too intensive, marker detection with the help of the ArUco-markers is also used here. Finally, the purpose of case three is to detect and track objects in 3D with a mobile camera. Only proposals for approach on the tracking problem have been made for this case.

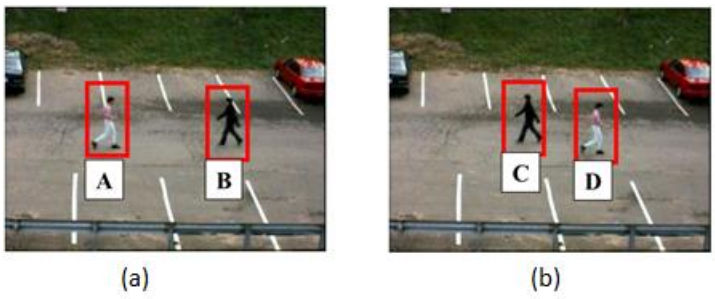
1 Inleiding

1.1 Situering

Deze masterproef wordt uitgevoerd in samenwerking met de onderzoeksgroep ACRO, behorende tot het Departement Werktuigkunde van KU Leuven en gelegen in het Technologiecentrum op Campus Diepenbeek. ACRO focust zich op de vakgebieden: automatisatie, computervisie en robotica [1].

Vandaag kent het gebruik van visiesystemen een sterke opmars in sectoren zoals de maakindustrie, beveiligingssector en culturele sector. Deze systemen worden onder andere ingezet om mensen, dieren en voorwerpen in diverse omgevingen te *tracken*. Deze trend zet zich eveneens voort in de roboticasector waar systemen worden ontwikkeld om camerabeelden om te vormen naar bruikbare data. Een voorbeeld hiervan is *padplanning* waarbij een map geconstrueerd kan worden door de data uit camerabeelden. Ook zal met deze data het mogelijk zijn dat een mobiele robot zich kan lokaliseren en navigeren doorheen de ruimte. Tegelijk kan het systeem ook andere voorwerpen of personen herkennen en lokaliseren waardoor de mobiele robot botsingen kan vermijden.

Echter, een belangrijk aspect in deze systemen en ook het onderwerp van deze masterproef is data-associatie. Hierbij associeert men, zoals de naam het aangeeft, data met data. Deze data kan bestaan uit mensen, dieren, objecten en allerlei verschillende vormen. De associatie van deze data gebeurt voor opeenvolgende beelden. Zo is bijvoorbeeld een persoon in opeenvolgende beelden gevolgd door de data-associatie, vervolgens zorgt dit voor het vormen van het afgelegde traject. Deze stap gebeurt voor elk beeld opnieuw. In Figuur 1 is er data-associatie gebruikt om persoon A en B te tracken. Dit wil zeggen dat in het volgende beeld (Figuur 1.b) persoon D geassocieerd is met persoon A omdat dit dezelfde persoon is en analoog voor persoon C met B. Hoe dit gebeurt is verder in deze masterproef duidelijk uitgelegd. In Figuur 1 is dit duidelijk niet gedaan waardoor er vier trajecten zijn in plaats van twee. Hoe data-associatie gebeurt is verder in deze masterproef duidelijk uitgelegd.



Figuur 1: (a) foto 1 met object A en B, (b) foto 2 met object C en D [2, p. 21]

1.2 Probleemstelling

Om personen en voorwerpen een eigen traject toe te kennen zal de data-associatie gebruik maken van een algoritme dat ervoor zorgt dat een bepaald persoon of voorwerp gevolgd wordt in opeenvolgende beelden. Dit wil zeggen dat men een traject aan elk voorwerp of persoon kan toekennen en dat men voor elke waarneming dit traject kan updaten. Hiervoor is er een gamma aan algoritmes en filters aanwezig. Dit gamma aan data-associatie is bruikbaar omdat elk algoritme/filter verschillende eigenschappen heeft. Maar ook maakt dit het moeilijk om een keuze te kunnen maken. Daarom heeft deze masterproef tot doel om een overzicht te scheppen in de beschikbare algoritmes en filters, om vervolgens voor drie casussen uit lopende doctoraatonderzoeken en een masterproef een geschikte oplossing voor te stellen.

Casussen:

De eerste casus heeft betrekking op het kalibreren van het bewegingsmodel van een gidsrobot. Deze gidsrobot bezit intern een afwijking waardoor beide aandrijfwielen niet met dezelfde snelheid roteren bij eenzelfde stuursignaal. Dit leidt tot afwijkingen van bijvoorbeeld een rechtlijnig pad bij een voorwaarts stuursignaal. Om dit te compenseren dient de relatie tussen de werkelijke poses van de mobiele robot en de stuursignalen bepaald te worden. Dit laat toe om een correctiefactor aan de sturing toe te voegen, dat deze afwijkingen compenseert. Voor deze casus is er dus nood aan een detector die de positie en oriëntatie van de mobiele robot kan bepalen.

De tweede casus kadert binnen mens-robotsamenwerking, waarbij de robot de mens superviseert tijdens de uitvoering van een assemblagetaak, met name: de assemblage van een kogellager. Doordat tijdens de assemblage bepaalde objecten tijdelijk verborgen kunnen worden, bv. door het verplaatsen van een kogel in de handpalm van de mens, kan de tracking van dit object verloren gaan. Om daarom een correcte supervisie van deze assemblagetaak uit te kunnen voeren, is de implementatie van data-associatiesoftware noodzakelijk.

De derde casus bestaat uit het detecteren en tracken van personen tijdens de navigatie van een mobiele robot. Een mobiele robot, uitgerust met camera's en bijkomende sensoren, dient zich een weg te banen in een dynamische omgeving met bewegende personen. Hiervoor is data-associatie belangrijk. Indien de mobiele robot een persoon ziet verdwijnen achter een obstakel, bv. een kast, is de kans reëel dat de persoon langs de andere zijde opnieuw verschijnt. Deze informatie stelt de robot in staat om tijdig te anticiperen om zo een botsingsvrije navigatie te garanderen.

1.3 Doelstellingen

Deze masterproef bestaat uit vijf grote uitdagingen. De eerste uitdaging is het gevolg van het steeds groeiende gamma aan detectoren en trackers. Hierin een selectie maken vereist een uitgebreide kennis, dat de keuze voor een geschikte oplossing bemoeilijkt. Het eerste onderdeel van deze masterproef bestaat daarom uit een literatuurstudie van de beschikbare detectoren en trackers met de focus op open-source software.

De tweede uitdaging volgt uit het gebrek aan data-associatie in de huidige open-source technieken. Data-associatie kan beschouwd worden als de link tussen twee opeenvolgende frames en heeft dus een groot effect op de kwaliteit en nauwkeurigheid van het volledige trackingsysteem. Ook hiervoor bestaan reeds diverse mogelijkheden die geïntegreerd kunnen worden in een trackingsysteem. Analoog aan de vorige uitdaging bestaat de oplossing hiervoor uit een uitvoerige literatuurstudie waarin deze verschillende technieken nader worden toegelicht.

De derde uitdaging is het voorstellen van een geschikte oplossing voor de eerdergenoemde casussen. Er is op dit moment geen trackingsysteem beschikbaar, dat een oplossing kan bieden voor elke toepassing daar elke methode zijn specifieke voor- en nadelen bezit. Daarom is het belangrijk om deze voor-en nadelen af te wegen voor elke toepassing. Dit kan leiden tot het gebruik van verschillende technieken voor de verschillende casussen.

De vierde is meer een persoonlijke uitdaging, zijnde het verwerven van een grote hoeveelheid kennis. Dit komt omdat computervisie een onderwerp is waar op dit moment zeer veel onderzoek naar wordt gedaan en ook zeer uitgebreid is. Ook zijn de algoritmes en principes van trackers, detectoren en data-associatie zeer wiskundig uitgelegd in papers, wat het moeilijk maakt om ze snel te begrijpen.

De laatste uitdaging is in de vorm van een probleem, zijnde de kwaliteit van de apparatuur waarmee deze masterproef verwezenlijkt wordt. Aangezien dit onderzoek zich voornamelijk toespitst op de kennis van de verschillende technieken eerder dan de uitwerking ervan is al het werk op een standaard laptop uitgevoerd. Dit heeft als nadeel dat het geen topkwaliteit processor heeft, dit botst uiteraard met het toepassen van deze algoritmes en technieken aangezien ze zeer vermogen- en geheugenintensief zijn.

1.4 Methodiek

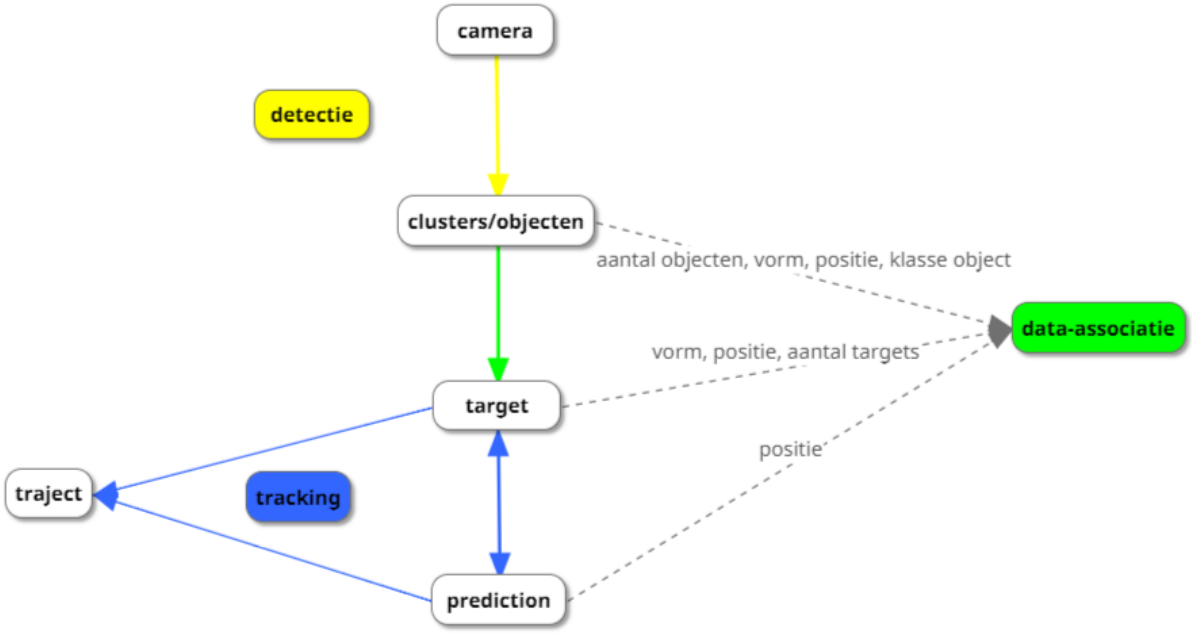
Deze masterproef start met een uitgebreide literatuurstudie, waarin de drie voornaamste onderdelen van het trackingsproces alsook het beschikbare gamma aan detectoren en trackers in verder detail worden beschreven. Aangezien deze masterproef meer gefocust is op het zoeken naar een oplossing dan het volledig oplossen van de casussen zal de literatuurstudie zeer uitgebreid zijn. Hierdoor kan dit hoofdstuk dienstdoen als naslagwerk voor toekomstige onderzoekers of studenten in dit domein.

Vervolgens zal Hoofdstuk 3 zich toespitsen op markertracking om zo de detectie te vereenvoudigen en de focus meer op tracking en detectie te leggen. De markers zorgen op hun beurt voor een goede benadering van de detectie. In Hoofdstuk 3 wordt er gekozen voor de ArUco-module en zal deze verder behandeld worden om toe te passen voor de casussen. Ook zal er een nauwkeurigheidsstudie worden gedaan om te achterhalen welk markertype het beste is en wat hun eigenschappen en struikelblokken zijn.

In Hoofdstuk 4 worden de drie casussen die in de probleemstelling zijn aangekaart behandeld. De eerste casus (de gidsrobot) is volledig uitgewerkt met in Hoofdstuk 3 uitleg over de gebruikte techniek, de nodige aanpassingen en een nauwkeurigheidsstudie. Voor de tweede casus (mens-robotsamenwerking) zal wegens de laatste uitdaging van de doelstellingen een benadering worden uitgewerkt. Ten slotte is er voor de derde casus (de mobiele robot), ook wegens de laatste uitdaging, enkel een aanpak uitgeschreven omdat het de uitwerking hiervan nood had aan financiële middelen die niet ter beschikking waren tijdens deze masterproef.

2 Literatuurstudie

Voor de bronnenstudie van deze masterproef wordt het volledige trackingsproces opgedeeld in drie onderdelen. Als eerste de detectie, als tweede het tracken en ten slotte de data-associatie, deze stap is de link tussen de twee voorgaande stappen en is verantwoordelijk voor het correct linken van de data doorheen de tijd. Voor elk van deze onderdelen worden er algoritmen, methodieken en filters besproken. In Figuur 2 zijn de drie onderdelen van het trackingsproces duidelijk zichtbaar. Hun functie en de relatie tussen de onderdelen is aangeduid met pijlen. Startend van bovenaan zorgt de detectie ervoor dat uit de camerabeelden verschillende objecten worden gedetecteerd en dat hun positie in het beeldscherm gekend is. Daarbij zal ook informatie over de vorm van het object bemachtigd worden, dit is mogelijk door een vierkant (*bounding box*), eclips of op andere vorm rond het object te plaatsen. Het tweede onderdeel dat zich centraal bevindt is de data-associatie. De data-associatie is verantwoordelijk voor het associëren van een object bij opeenvolgende beelden. Hiervoor zal de data-associatiestap nood hebben aan veel informatie, dit is geïllustreerd met de stippellijnen, maar welke informatie het gebruikt is volledig afhankelijk van het type algoritme. Ten slotte het onderdeel tracking. Het doel van dit onderdeel is om aan de hand van de huidige locatie van een object/target en informatie uit een vorige tijdsstap, zijn volgende locatie te voorspellen en om de juistheid van de huidige locatie te bevestigen of zelfs aan te passen. De informatie (aantal objecten, vorm, locatie, etc.) van het target wordt over de verschillende beelden bijgehouden en vormt dan het traject van het target.



Figuur 2: Schematisch overzicht van de 3 onderdelen van het trackingsproces

2.1 Detectie

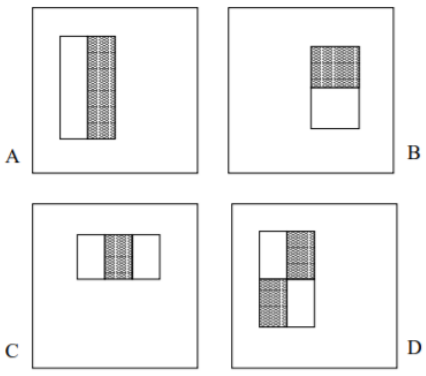
Dit hoofdstuk beschrijft de werking van verscheidene detectieprogramma's en -methodes op conceptueel niveau. Dit omdat de focus van deze masterproef eerder ligt op de overige twee stappen in het trackingsproces. In dit hoofdstuk wordt een onderscheid gemaakt tussen twee markertypes, namelijk marker- en objectdetectie. Markerdetectie kan beschouwd worden als een subcategorie van objectdetectie, waarbij de detectie wordt vereenvoudigd doordat het te detecteren object exact gekend is. Dit in tegenstelling tot de detectie van objecten zoals gezichten waarin een grote variatie bestaat.

2.1.1 Gezichtsdetectie

Voor het hoofdstuk detectie zal er gestart worden met gezichtsdetectie aangezien dit de eerste niet-marker gerelateerde detectie die real-time gerealiseerd is. Uiteraard starten we ook met een van de bekendste onderzoeken in gezichtsdetectie, namelijk die van Viola en Jones [3]. Deze heeft tot de dag van vandaag nog zeer veel toepassingen en was ook het eerste soort herkenningsalgoritmes die real-time mogelijkheden had.

2.1.1.1 Viola-Jones algoritme

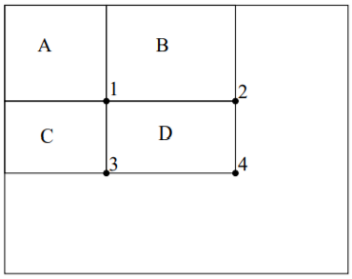
In het Viola-Jones algoritme wordt er gebruik gemaakt van een serie van simpele classificatoren die elk instaan voor de detectie van een bepaald gezichtsfeature. Het Viola-Jones algoritme detecteert niet ogen maar detecteert het verschil tussen de ogen en de neus of voorhoofd. De eerste stap vertrekt bij zeer simpele filters, dit door gebruik te maken van "*Haar basis functions*". Deze functies bestaan uit simpele rechthoeken en worden in Figuur 3 afgebeeld. Deze functies of filters zorgen voor features, deze is voor A en B een rechthoekige feature bestaande uit twee segmenten, voor C is dit een rechthoekige feature bestaande uit drie segmenten en voor D is dit een rechthoekige feature bestaande uit vier segmenten. Deze features berekenen het verschil tussen de som van de pixels in het zwarte segment(en) en de som van de pixels in het witte segment(en) [3].



Figuur 3: Rechthoekige features van Viola-Jones [3, p. 2]

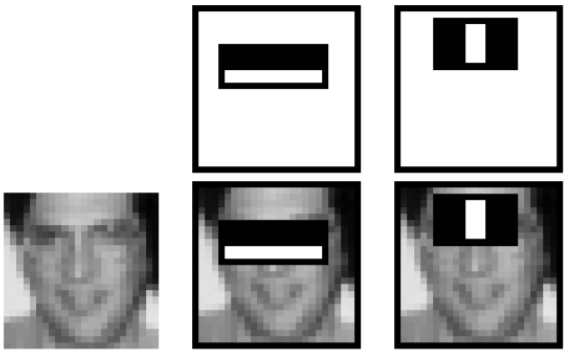
Aangezien deze aanpak zeer rekenintensief is, maken ze gebruik van een integraalbeeld. In deze afbeelding worden pixels (waardes) van de originele afbeelding vervangen door de som van de pixels links en boven de overeenkomstige pixel. Hierdoor staat de som van elke rechthoek startend in de linkerbovenhoek altijd in de rechteronderhoek van de rechthoek. Zo kan de som van de intensiteiten van de pixels I(p) behorende in het rechthoekige gebied D in Figuur 4 worden berekend door volgende vergelijking:

$$I(4) + I(1) - I(2) - I(3)$$ [3].



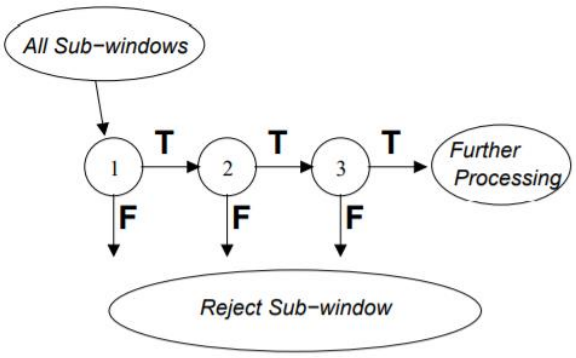
Figuur 4: Oppervlakte van rechthoek D berekenen [3, p. 3]

Hierdoor helpt dit integraalbeeld met het uitvoeren van oppervlakteberekeningen te, dit is nodig omdat de rechthoekige features op verschillende groottes over het scherm worden gegleden. In de volgende stap onderzoekt men welke features het beste werken om een gezicht van een achtergrond te onderscheiden. In Figuur 5 worden twee geschikte features hiervoor getoond met links de originele foto. In het midden de feature die steunt op het principe dat er een intensiteitsverschil is tussen de zone van de ogen en die van de kaken. Rechts staat de feature die steunt op het principe dat de ogen op een foto vaak donkerder zijn dan de neus [3].



Figuur 5: Sterke features voor het onderscheiden van gezichten met achtergronden [3, p. 4]

Om vervolgens gezichten te kunnen herkennen in diverse afbeeldingen zal er op verschillende features getest worden, hiervoor wordt er gebruik gemaakt van een cascade. Een cascade is een beslissingsboom die Figuur 6 weergeeft, hierin zal een positief resultaat van de eerste classificator (maakt gebruik van een of meerdere features) verder gaan naar de volgende classificator. De eerste classificator is een snelle classificator en gebruikt maar weinig features. Hoe verder in de cascade hoe sterker en zwaarder de classificator is [3].

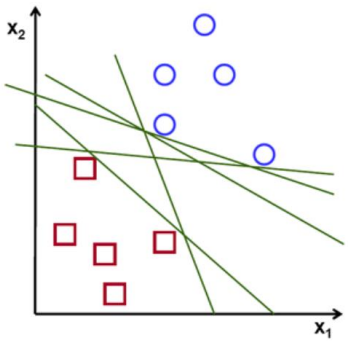


Figuur 6: Schematische voorstelling van een detectiecascade [3, p. 5]

Deze cascade wordt op een afbeelding uitgevoerd, dit betekent dat het verschillende *"sub-windows"* of deelvensters gebruikt worden om initieel te zoeken naar gewenste features. Deze deelvensters hebben verschillende groottes en bevinden zich op verschillende posities in de afbeelding. De cascade bezit het grote voordeel dat het voor het merendeel van de afbeelding weinig berekeningen moet doen. Dit omdat de eerste classificator snel is en op weinig features filtert. Hierdoor wordt alles wat bij de eerste classificator faalt verworpen en enkel de deelvensters die slagen worden meegenomen naar de volgende classificator [3]. Het is bijgevolg een snelle detector voor gezichten en wordt na twee decennia nog frequent gebruikt.

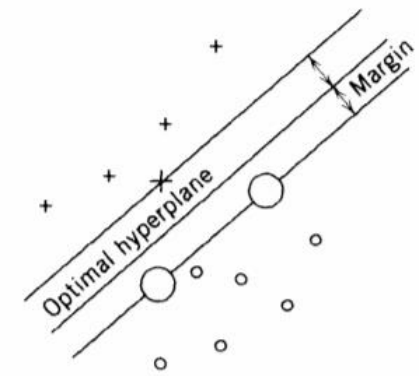
2.1.2 SVM

Het doel van een *support vector machine* (of *support vector classifiers*) is om het aantal fouten tijdens training te minimaliseren en de marge tussen twee of meerdere klassen te maximaliseren. Om dit visueel te duiden, wordt door middel van het voorbeeld in Figuur 7 toegelicht. Hier worden 2D-punten in twee klassen opgedeeld door een rechte lijn. In werkelijkheid zullen de punten vectoren zijn en de rechte een hypervlak. Een hypervlak is exact één dimensie minder dan de gebruikte vectorruimte, dus in dit geval zal het 1D zijn aangezien de vectorruimte 2D is. Voor dit hypervlak (rechte) bestaan uiteraard talloze oplossingen zoals in Figuur 7 weergegeven is [4].



Figuur 7: Classificatie voor twee klassen van 2D-punten [4, p. 1]

Om de classificatie zo succesvol mogelijk te maken wordt er gezocht naar het optimale hypervlak, hierbij zal de marge tussen de twee klassen het grootst zijn zoals in Figuur 8 [5].



Figuur 8: Voorstelling van het optimale hypervlak voor 2D-punten [5, p. 402]

Voor het opstellen van dit hypervlak maakt men gebruik van het *Rosenblatt's perceptron learning* algoritme. Hierbij wordt de keuzegrens/hypervlak voorgesteld als $|\beta_0 + \beta^T x| = 1$ met β_0 het gewicht, β de bias, en x de trainingsvoorbeelden het dichtste bij het hypevlak, deze worden de *support vectors* genoemd. De afstand van de support vectors tot het hypervlak wordt door formule (2.1) berekend.

$$afstand = M = \frac{|\beta_0 + \beta^T x|}{||\beta||} \qquad (2.1)$$

Door de op voorhand gekozen voorstelling van het hypervlak in te vullen wordt $M = \frac{1}{||\beta||}$ bekomen als afstand tot de dichtstbijzijnde punten voor het hypervlak met een marge gelijk aan tweemaal deze afstand. Om nu de maximale marge te berekenen, zal er gezocht worden naar een minimale waarde voor β voor alle punten van de trainingsvoorbeelden [4] [6].

2.1.3 Deep learning

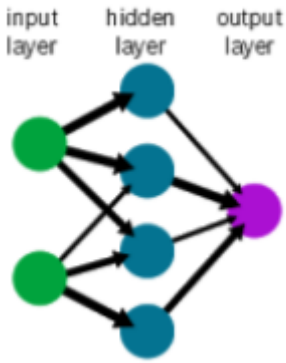
Het principe achter *deep learning* is dat "een machine", in dit geval een classificator, getraind wordt om te doen wat eigenlijk ook onze hersenen doen. Aan de hand van bepaalde karakteristieken (zoals de lange nek van een giraf of de grote grijze huid van een olifant) wordt het object bepaald en gelinkt met een bepaalde klasse. Net zoals bij onze hersenen kunnen classificatoren objecten die ze niet kennen ook niet classificeren. Hierdoor detecteren ze de objecten ook niet maar worden ze als omgeving/achtergrond beschouwd. Daarom is het belangrijk dat de classificatoren getraind worden, en hiervan komt de naam deep learning.

Het leerproces wordt uitgevoerd door het actief te testen op zeer grote datasets, hoe dit exact gebeurt wordt verder uitgelegd. Er bestaan twee verschillende soorten van deep learning, zijnde offline en online learning. Het grote verschil is dat de offline classificatoren op voorhand worden getraind. De online classificatoren worden tijdens het gebruik continu geüpdatet aan de hand van nieuwe data.

Om de karakteristieken uit beeldmateriaal te kunnen extraheren zal men gebruik maken van een neuraal netwerk. Dit en de meest bekende neurale netwerken worden in de volgende alinea besproken. Vervolgens zullen er ook enkele classificatoren aan bod komen.

2.1.1.2 Neuraal Netwerk

Een neuraal netwerk (NN) is een netwerk dat opgesteld is uit neuronen en gewichten. Een vereenvoudigde versie hiervan wordt in Figuur 9 getoond. In de afbeelding zijn neuronen voorgesteld als cirkels, waarbij de dikte van de pijlen tussen deze neuronen een maat vormen voor de gewichten. Zoals de afbeelding weergeeft zijn er verschillende niveaus in het netwerk, het aantal niveaus en de hoeveelheid neuronen per niveau is afhankelijk van de toepassing. Bij een NN zijn de opeenvolgende niveaus afhankelijk van de voorgaande waardoor de output van de ene laag wordt doorgegeven aan de volgende. Voor de toepassing van de paper ‚waaruit Figuur 9 is ontleed [7], zullen de input neuronen de pixels (grijswaarden of BRG-waardes) zijn en de output de classificatie van een bepaald gebied (pixels). Bijvoorbeeld bij het tonen van een hond aan het NN zullen er door de waardes uit de foto neuronen in de volgende laag worden geactiveerd enzovoort. Het activeren van een neuron wordt bepaald door de som van de gekoppelde neuronen uit de vorige laag, die elk vermenigvuldigd worden met hun eigen gewicht. Om een correcte classificatie bij een foto te bekomen, moet het NN getraind worden. Dit trainen wordt geschiedt door de gewichten zodanig aan te passen dat het neuraal netwerk de juiste classificatie geeft, hiervoor is een zeer grote gelabelde dataset nodig[7].

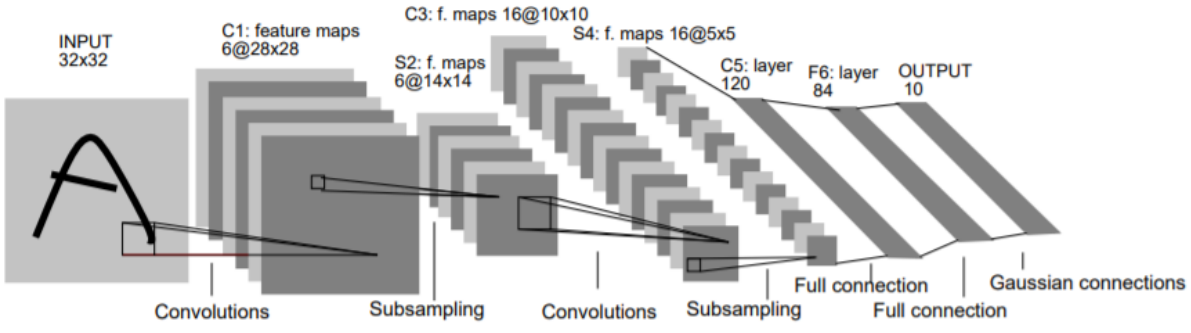


Figuur 9: Simpel neuraal netwerk [7]

2.1.1.3 Convolutioneel Neuraal Netwerk (CNN)

Een convolutioneel neuraal netwerk is een netwerk met convolutionele niveaus. Deze niveaus kunnen vergeleken worden met filters/maskeroperaties, bijvoorbeeld om randen of cirkels in een afbeelding te detecteren. Dit is uitgevoerd door een filter hierop toe te passen en zo verkrijgt men feature mappen, met voor elke filter één feature map. Dit zorgt ervoor dat er zeer veel data is en daarom nood is aan het verminderen van de hoeveelheid data. Er wordt dus aan *pooling* (sub-sampling) gedaan, door verlagen van de resolutie van de feature mappen. Hierbij verkleinen de feature mappen in data door bijvoorbeeld voor elke 2x2 pixels de maximale bij te houden. De hoeveelheid stappen en de volgorde hiervan kan van netwerk tot netwerk verschillen. Om de gepoolde feature mappen om te vormen naar volledige geconnecteerde laag zullen de 3D feature mappen naar een 1D feature vector omgevormd moeten worden. Vervolgens werkt dit als een NN en zal de combinatie van verschillende features d.m.v. training de CNN-classificatie mogelijk zijn. In Figuur 10 wordt het LeNet-5 voorgesteld, deze opstelling wordt gebruikt voor handschriftherkenning. Zoals eerder vermeld zijn er verschillende andere netwerken waarvan de werking analoog is aan deze van het LeNet-5 [8].

Vooraleer we de aanpak van een R-CNN-netwerk bespreken, overlopen we eerst de voorgaande techniek. Zoals eerder uitgelegd zal een CNN een groep pixels in een neuraal netwerk sturen om een classificatie hiervan te verkrijgen. Om deze groep van pixels te verkrijgen, wordt er over het scherm gegleden met verschillende groottes van vierkanten om deze vervolgens door te sturen naar het CNN. Dit is uiteraard een zeer intensief proces aangezien dit niet enkel voor elke pixel gedaan moet worden, maar ook voor verschillende groottes van vierkanten.

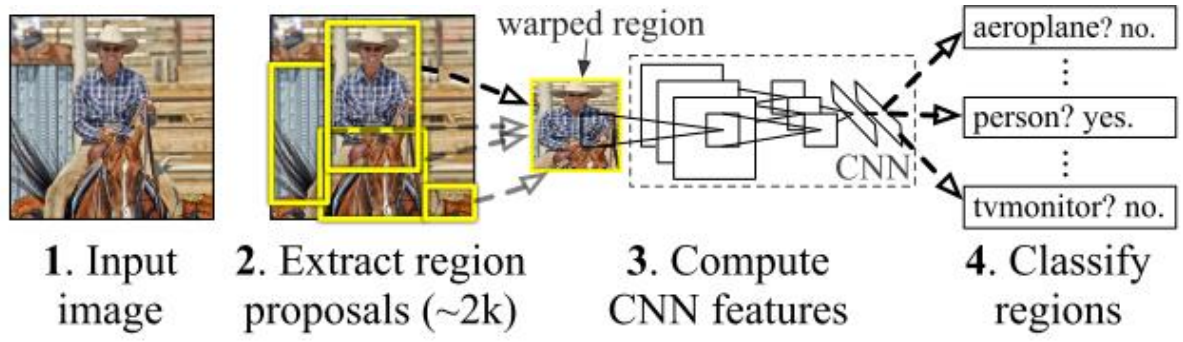


Figuur 10: Architecture of LeNet-5 [8, p. 7]

2.1.1.4 R-CNN (Regiogebaseerd Convolutioneel Neuraal Netwerk)

De werking van een R-CNN verschilt van deze van een CNN dat ze niet alle pixels afgaan, maar dat ze voorstellen doen op basis van een algoritme dat selectief zoekt. Deze stap wordt de *region proposal* stap genoemd en de voorstellen worden dan vervolgens in de CNN gevoed om zo classificatie te verkrijgen aan de hand van een SVM.

Op het moment van ontwikkeling in 2012 was dit een *"state of the art"* detectiesysteem vanwege zijn uitstekende performantie op de verscheidene testen. Maar het was een traag proces omdat elk voorstel doorheen de R-CNN moest passeren [9]. De werking van een R-CNN-detectiesysteem wordt in Figuur 11 weergegeven.

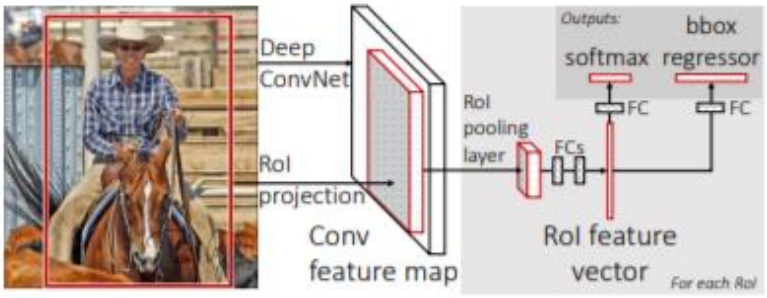


Figuur 11: Werking R-CNN [9, p. 2]

Omdat het CNN een vast aantal pixels/resolutie (inputs) gebruikt voor het verwerken van afbeeldingen, worden deze omgevormd naar één uniforme vorm. Dit proces noemt men *warpen* en is noodzakelijk omdat het inputbereik van het CNN niet aangepast kan worden.

2.1.1.5 Fast R-CNN

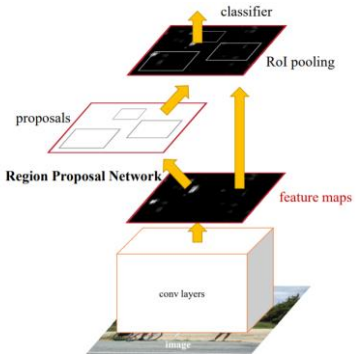
Dit algoritme is een verbetering op de R-CNN, het werkt voornamelijk met dezelfde componenten maar met een andere aanpak. Deze werking wordt in Figuur 12 afgebeeld. In plaats van iedere regio in de afbeelding afzonderlijk doorheen het CNN te sturen, zal men de volledige afbeelding met de voorstellen (door het selectief zoeken) doorheen het CNN sturen. Er is hier wel een groot verschil met het CNN. In dit netwerk worden enkel de feature mappen gemaakt en vervolgens de opsplitsing per voorstel uitgevoerd. Hierna komen de volgende stappen, namelijk warpen en pooling. Zo wordt de datahoeveelheid verlaagd en uiteindelijk de belangrijkste informatie behouden. Vervolgens worden de feature mappen omgevormd tot een volledig geconnecteerde laag [10].



Figuur 12: Werking Fast R-CNN [10, p. 2]

2.1.1.6 Faster R-CNN

Fast R-CNN komt al zeer dicht bij een oplossing voor real-time detectie, met als knelpunt de tijd die het maken van de voorstellen in beslag neemt. Bij faster R-CNN zal men in plaats van een algoritme voor selectief te zoeken om voorstellen te doen, een *Region Proposal Network* (RPN) integreren in het detectiesysteem. Het bijzondere aan dit is dat het RPN de convolutionele niveaus deelt met het detectiesysteem. Dus in plaats van op voorhand voorspellingen te doen, worden de voorspellingen gebaseerd op de feature mappen die uit het CNN komen. Figuur 13 geeft dit schematisch weer. In principe is het RPN een convolutioneel netwerk dat zijn niveaus deelt met de classificator. Hierdoor kan eveneens de lokalisatie getraind worden op dezelfde manier als de classificatie [11].



Figuur 13: Werking faster R-CNN [11, p. 3]

2.1.1.7 TensorFlow

TensorFlow is Google's tweedegeneratie systeem voor de implementatie en gebruik van grootschalige leermodellen. Deze soort leermodellen zijn aanwezig in verschillende diensten die Google aanbiedt, zoals Google Translate, Google Photos, Google Maps, YouTube, etc. TensorFlow werkt aan de hand van een data-flowmodel en maakt dit mogelijk voor een breed gamma aan hardware platforms, van een mobiele apparaatplatform tot grootschalige

trainingssystemen die draaien op honderden gespecialiseerde machines met duizenden GPU's [12].

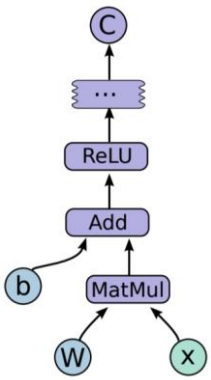
Een TensorFlow-programma is een programma dat werkt met Intelligente bouwblokken met specifieke eigenschappen en functies, die aan elkaar gekoppeld kunnen worden. Vergelijkbaar met een flowchart. Met als uitbreiding dat deze bouwblokken geüpdatet kunnen worden om de vertakking en looping van het programma te wijzigen. Figuur 14 toont voorbeeldcode met in Figuur 15 de bijhorende grafiek.

```
import tensorflow as tf

b = tf.Variable(tf.zeros([100]))                    # 100-d vector, init to zeroes
W = tf.Variable(tf.random_uniform([784,100],-1,1)) # 784x100 matrix w/rnd vals
x = tf.placeholder(name="x")                        # Placeholder for input
relu = tf.nn.relu(tf.matmul(W, x) + b)              # Relu(Wx+b)
C = [...]                                           # Cost computed as a function
                                                    # of Relu

s = tf.Session()
for step in xrange(0, 10):
  input = ...construct 100-D input array ...        # Create 100-d vector for input
  result = s.run(C, feed_dict={x: input})           # Fetch cost, feeding x=input
  print step, result
```

Figuur 14: Voorbeeldcode met TensorFlow [12, p. 3]



Figuur 15: Grafiekvoorbeeld [12, p. 3]

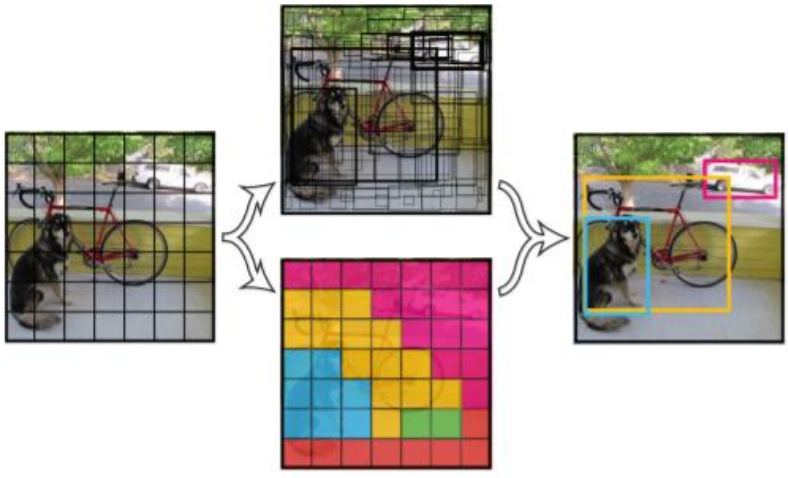
Een bouwblok kan dus geen of meerdere inputs en/of outputs hebben, en de waardes die van input naar output gaan worden tensors genoemd. Om met deze tensors en knooppunten te werken bevat TensorFlow verschillende operaties, deze gaan van optellen tot matrix-operaties. Figuur 16 toont enkele elementaire operaties uit de TensorFlow-bibliotheek [12]. Bij TensorFlow zijn de tensoren veranderlijk en daarom zijn ze zeer geliefd voor het gebruik in neurale netwerken.

Category	Examples
Element-wise mathematical operations	Add, Sub, Mul, Div, Exp, Log, Greater, Less, Equal, ...
Array operations	Concat, Slice, Split, Constant, Rank, Shape, Shuffle, ...
Matrix operations	MatMul, MatrixInverse, MatrixDeterminant, ...
Stateful operations	Variable, Assign, AssignAdd, ...
Neural-net building blocks	SoftMax, Sigmoid, ReLU, Convolution2D, MaxPool, ...
Checkpointing operations	Save, Restore
Queue and synchronization operations	Enqueue, Dequeue, MutexAcquire, MutexRelease, ...
Control flow operations	Merge, Switch, Enter, Leave, NextIteration

Figuur 16: Operaties uit de TensorFlow-bibliotheek [12, p. 3]

2.1.1.8 YOLO (Darknet)

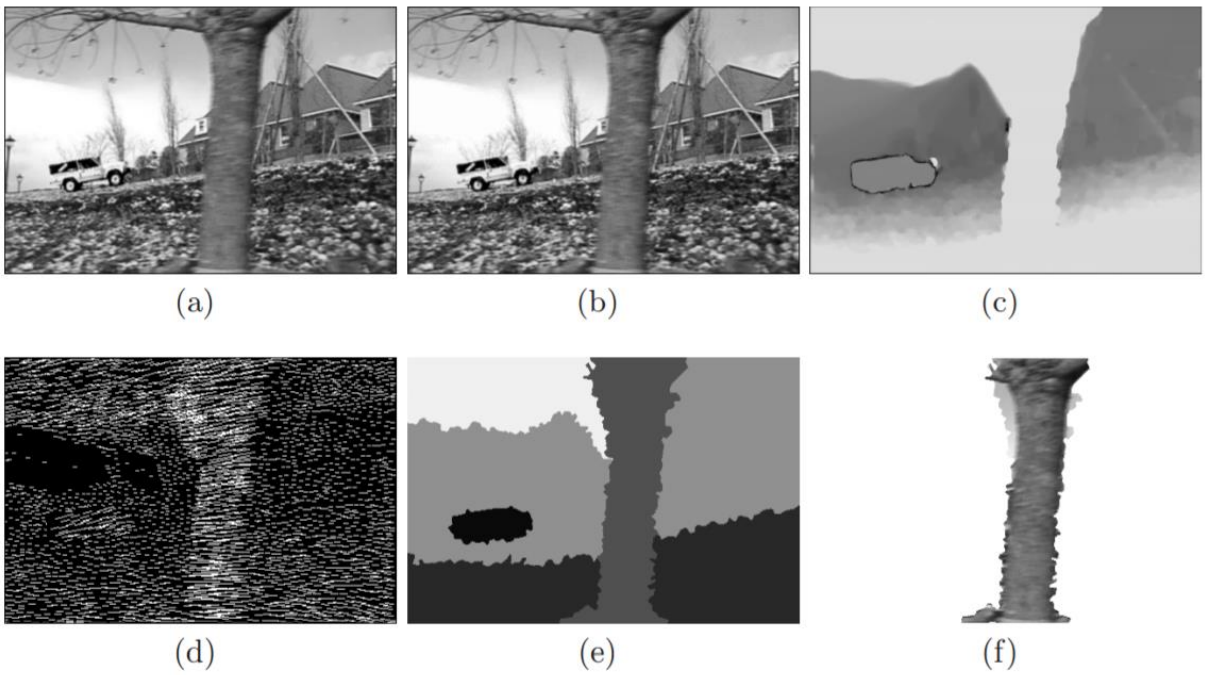
Bij YOLO is er een andere aanpak als het komt op voorspellen waar objecten zijn, het zal zoals de naam zegt (*You Onely Look Once*) alle bounding boxes in één keer voorspellen. Zo worden afbeeldingen opgedeeld in een raster (S x S), dat in Figuur 17 (links) wordt getoond. Vervolgens zal het model voor elke cel van dit raster B aantal bounding boxes voorspellen en aan elk een zekerheidsscore toekennen. Deze zekerheidsscore is een waarde voor hoe zeker het model is dat de box een object bevat en hoe accuraat die voorspelling is. Ook dit wordt weergegeven in Figuur 17 (boven), hier vertegenwoordigt de dikte van de bounding box de zekerheidsscore. Dus hoe dikker de box, hoe zekerder het model is dat er een object in aanwezig is. Tegelijkertijd zal het model voor elke bounding box voorspellen tot welke klasse het behoort door middel van een CNN. Figuur 17 (onder) toont de toekenning van elke rastercel aan een specifieke klasse. Dit is de klasse van de bounding box die het sterkst aanwezig is in deze cel. Deze twee kansen worden vermenigvuldigd om de klasse-specifieke zekerheidsscore voor elke box te bekomen, welke vervolgens met een drempelwaarde wordt vergeleken. Dit zorgt ervoor dat enkel de sterkste voorspellingen overblijven. Dit kan opgemerkt worden in Figuur 17 (rechts) [13].



Figuur 17: Het YOLO-model [13, p. 2]

2.1.4 Optical flow

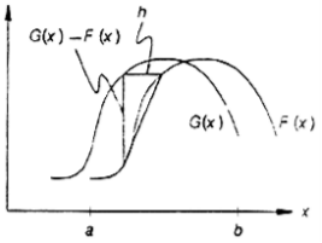
Optical flow kan gezien worden als de beweging van objecten, randen en vlakken tussen de observator en de geobserveerde omgeving (wat in beeld is). In Figuur 18 is een tuin met een auto in en een boom en de voorgrond waarbij de camera zal bewegen. Dit zorgt voor optische flow, omdat randen en vlakken tussen de twee opeenvolgende frames (a) en (b) in zullen bewegen. Deze bewegingen kunnen dan vervolgens als vectoren gerepresenteerd worden, zoals in (d) zichtbaar is. Met behulp van deze vectoren is het mogelijk om de volledige afbeelding te segmenteren gebaseerd op hun optische beweging, zoals in (e) zichtbaar is.



Figuur 18: Voorbeeld Optical flow met (a) en (b) de opeenvolgende frames, (e) de computerberekende optische flow ,(d) de regiogebaseerde vectoren, (e) de segmentatie gebaseerd op beweging, (f) het boomsegment [14, p. 616]

2.1.4.1 Lucas-Kanade methode

Om deze methode uit te leggen wordt er net zoals in [15] begonnen met de uitleg voor een ééndimensionaal voorbeeld en vervolgens een multidimensionaal. Het 1D-voorbeeld is afgebeeld in Figuur 19, met als dimensie enkel de horizontale beweging h tussen G(x) en F(x).



Figuur 19: 1D-voorbeeld van Lucas-Kanade methode [15, p. 2]

Om de afstand h te berekenen wordt veronderstelt dat h een zeer kleine waarde heeft bij een lineaire gedrag van de functie F(x).

$$F'(x) \approx \frac{F(x+h)-F(x)}{h} = \frac{G(x)-F(x)}{h} \qquad (2.2)$$

Met h:

$$h \approx \frac{G(x)-F(x)}{F'(x)} \qquad (2.3)$$

Uit (2.3) blijkt dat de waarde van h afhankelijk is van de waarde x, en hiervan neemt men het gemiddelde.

$$h \approx \sum_x \frac{G(x)-F(x)}{F'(x)} / \sum_x 1 \qquad (2.4)$$

Om deze benadering te verbeteren kan men naar de 2de afgeleide kijken en deze mee in rekening brengen door het als gewicht te gebruiken. Daarom zal de constante waarde van 1/h weggelaten worden en dit geeft het gewicht w(x).

$$F''(x) \approx \frac{G'(x)-F'(x)}{h} \qquad (2.5)$$

$$w(x) \approx \frac{1}{|G'(x)-F'(x)|} \qquad (2.6)$$

Na het integreren van (2.6) in (2.4) verkrijgt men:

$$h \approx \sum_x \frac{w(x)\,[G(x)-F(x)]}{F'(x)} \;/\; \sum_x w(x) \qquad (2.7)$$

Om de beste oplossing te bekomen wordt een Newton-Raphson iteratie toegepast.

$$h_0 = 0$$

$$h_{k+1} = h_k + \sum_x \frac{w(x)\,[G(x)-F(x+h_k)]}{F'(x+h_k)} \;/ \sum_x w(x) \qquad (2.8)$$

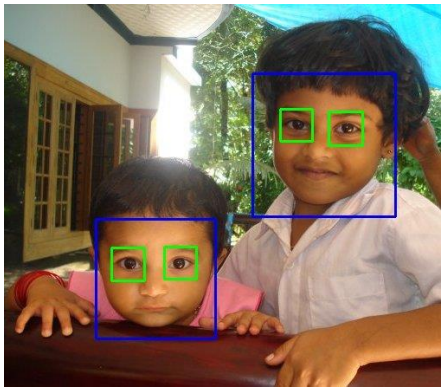
Aan de hand van de Newton-Raphson iteratie kan de Lucas-Kanade methode sneller features over frames heen matchen. Dit omdat er gebruik wordt gemaakt van de eerste en tweede afgeleide van de gradiënt. Hierdoor heeft het programma meer informatie over de te matchen features waardoor ze sneller een oplossing vindt [15].

2.1.5 OpenCV detectoren

In 1999 had het Intel-team als doel om een opensource computervisiebibliotheek te ontwikkelen, OpenCV genaamd. Deze software mag voor zowel academische als commerciële doeleinden gebruikt worden mits een BSD-vergunning. Deze bibliotheek bevat verschillende modules die elk dienen voor specifieke computervisieproblemen [16]. Deze modules bevatten algoritmes waarvan enkele verder besproken zullen worden.

2.1.3.1 Haarcascades

De haarcascades werden bij het Viola-Jones algoritme reeds aangehaald en bestaan uit een reeks simpele classificatoren. Dit type classificator behoort tot de offline learning classificatoren, omdat voor het gebruiken van deze detectie de classificator enkel gedownload moet worden. Deze classificatoren zijn op voorhand getraind door bedrijven zoals Intel om één specifiek ding te detecteren en de achtergrond te negeren en dit in real-time. De meest gebruikte en bekendste is de gezichtsdetectie (Viola-Jones) omdat deze talloze toepassingen heeft, bijvoorbeeld Snapchat, automatische focus bij camera's en nog veel meer. Het grote voordeel hiervan is dat ze zeer snel zijn en ook de mogelijkheid geven om er zelf één te bouwen die gespecialiseerd is in de detectie van voorwerpen naar keuze. Het proces om een eigen cascade te trainen is tijdrovend en vereist een grote hoeveelheid data en wordt bijgevolg best niet op een doorsnee computer uitgevoerd. Men kan hiervoor servers huren bij bedrijven voor een voordelige prijs. Zoals eerder vermeld heeft het trainen zeer veel afbeeldingen nodig van het te detecteren object en zijn locatie op de afbeelding. Er bestaan technieken om dit proces te vereenvoudigen in plaats van 10000 foto's te maken, maar hierop zal in deze masterproef verder niet worden ingegaan. In Figuur 20 kan men een voorbeeld terugvinden van de werking van een haarcascade voor de ogen (groen) en voor het gezicht (blauw) [17].



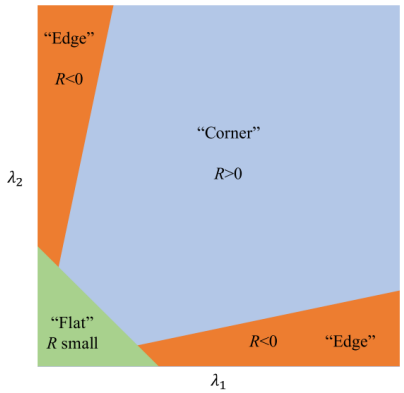
Figuur 20: Haarcascade voor ogen (groen) en voor het gezicht (blauw) [17]

2.1.3.2 Featuredetectie

Om te achterhalen wat het object is, is er nood aan informatie. Dit in de vorm van features. Bijvoorbeeld voor de mens zal het zien van een groene langwerpige lijn tezamen met andere karakteristieke kenmerken onmiddellijk een link leggen met gras. Voor een computer is dit niet zo evident aangezien zij die link tussen objecten en bepaalde features niet hebben zoals het menselijke brein. Daarom zal dit aan een computer moeten geleerd worden of met die reden geprogrammeerd worden. Beginnend door lijnen, hoekpunten en simpele maar opvallende features zoals kleur- of intensiteitsveranderingen van een afbeelding in kaart te brengen. Aan de hand van deze features kunnen objecten gedetecteerd en vervolgens getrackt worden.

Harris hoek Detection

Het basisidee achter de hoekdetectie is dat een hoek een sterke intensiteitsverandering heeft voor beide richtingen in zijn nabije omgeving. Dus in het venster (kleine zone rondom een specifieke pixel bv 5 op 5 pixels) rondom de feature zal het bewegen van het venster een grote intensiteitsverandering als gevolg hebben [18]. Dit in tegenstelling tot een plat vlak waarbij de intensiteitsverandering gering is of bij een lijn waarbij de verandering slechts in één richting een grootte waarde heeft. Om hoeken vervolgens te detecteren wordt elke pixel in het frame uitgedrukt met behulp van de eigenvectoren. Eigenvectoren geven de verandering van de omgeving aan. Dus als er een hoek aanwezig is, zullen de eigenvectoren van deze pixel groot zijn. In Figuur 21 is er een classificatie via de eigenvectoren (λ_1, λ_2) te zien [18].



Figuur 21: Classificatie a.d.h.v. de eigenvectoren [18, p. 313]

Een maat voor de kwaliteit van de hoekpunten is gegeven door formule (2.9). Voor de afleiding van deze formule wordt naar [18] gerefereerd. Met K typisch tussen 0,04 en 0,06 en λ_1, λ_2 de eigenvectoren van de featurepunten [18].

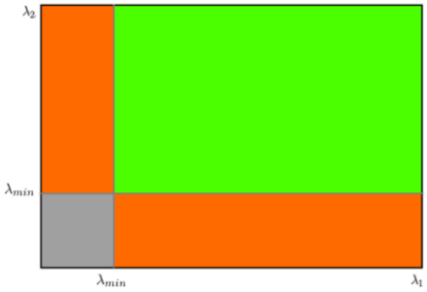
$$RH = \lambda_1\lambda_2 - K.(\lambda_1 + \lambda_2)^2 \qquad (2.9)$$

Shi-Tomasi Corner Detector & Good Features to Track

Shi en Tomasi hebben enkele kleine aanpassingen gemaakt aan de Harris Corner detector om vervolgens betere resultaten te bekomen. Zo wordt de kwaliteit van de features (hoekpunten) bekomen door middel van volgende formule:

$$R = min(\lambda_1 \lambda_2) \qquad (2.10)$$

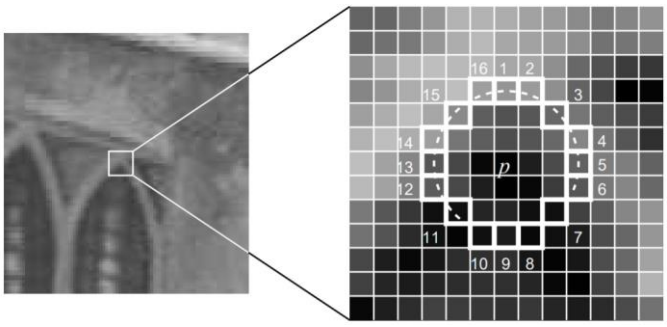
De waarde R wordt vervolgens vergeleken met een drempelwaarde λ_{min}. Indien R groter is dan deze drempelwaarde, geldt volgens formule (2.10) dat beide eigenwaardes groter zijn dan λ_{min}. Door dit in de $\lambda_1 \lambda_2$-ruimte af te beelden bekomt men Figuur 22, waarbij features die in het groene gebied vallen gezien worden als goede features om te tracken [19] [20].



Figuur 22: Classificatie van punten a.d.h.v. Good Features to Track methode [20, p. 1]

FAST Algoritme

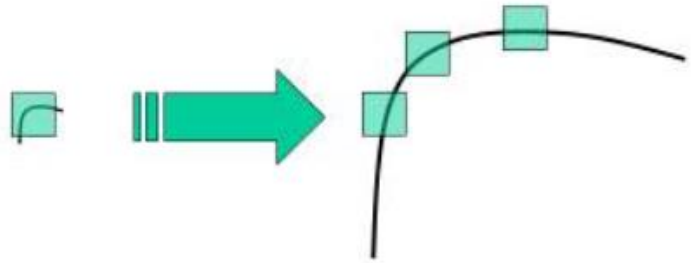
FAST *(Features from Accelerated Segment Test)* is een test die bepaalt of een punt in een afbeelding beschouwd kan worden als een hoekpunt (*corner*). Een pixel P in de afbeelding wordt gezien als een hoekpunt als N aaneensluitende pixels in een cirkel (16 pixels) lichter zijn dan $I_p + T$ of donkerder dan $I_p - T$. Met I_p de intensiteit van pixel P, T een ingestelde drempelwaarde en N gelijk aan twaalf. Om deze procedure te versnellen, wordt eerst naar enkel pixels 1, 5, 9, 13 gekeken (zie Figuur 23) om zo vervolgens veel hoekpunten uit te sluiten. In het geval dat de intensiteiten van drie van deze pixels niet lichter zijn dan $I_p + T$ of donkerder dan $I_p - T$, voldoet dit punt niet aan de voorwaardes van een hoekpunt. Bijgevolg kunnen door deze test veel pixels uitgesloten worden, zonder overbodige berekeningen op pixels die niet aan de test voldoen. Dit versnelt de hoekdetectie aanzienlijk [21].



Figuur 23: Hoekdetectie a.d.h.v. intensiteit van omliggende cirkel (16 pixels) [21, p. 4]

SIFT *(Scale-Invariant Feature Transform)*

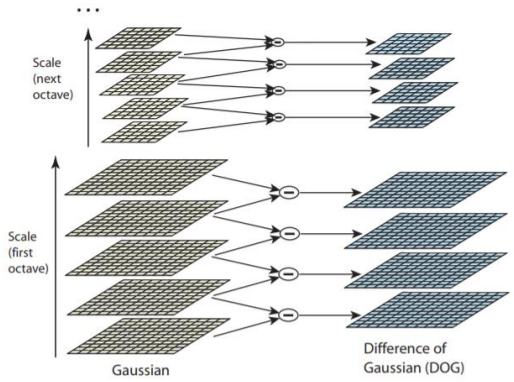
Als men de Harris Corner Detection bekijkt kan men concluderen dat deze features rotatie-invariant zijn. Dit betekent dat als de afbeelding/omgeving of camera draait dat dezelfde hoeken steeds gevonden worden, want een hoek blijft een hoek. Maar wanneer schaalverandering toegepast is, kunnen hoeken niet meer te detecteren zijn zoals in Figuur 24 te zien is [22].



Figuur 24: Detectie bij schaalverandering van een hoek [22, p. 1]

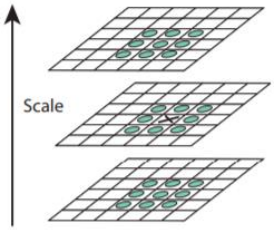
SIFT biedt een oplossing voor dit probleem aangezien het de data van de afbeelding omvormt naar schaal-invariante coördinaten. Deze techniek bestaat uit vier fasen stappen: *Scale-space extrema detection, Keypoint localization, Orientation assignment, Keypoint descriptor* [23].

Bij de *Scale-space extrema detection* wordt er gebruik gemaakt van verschillende vensters om zo hoeken van verschillende schalen te detecteren. Hiervoor wordt LOG (Laplace Of Gaussian) toegepast, omwille van het feit dat een Gaussische functie schaalonafhankelijk is en omdat met behulp van Laplace een extrema gevonden kan worden. Omdat dit Rekenintensief is, zal dit benaderd worden door het verschil in de Gaussische functie te berekenen. Dit wordt dan uitgevoerd voor verschillende octaven in de Gaussiaanse Pyramide zoals in Figuur 25 wordt getoond [23].



Figuur 25: Het verschil in de Gaussische functie voor opeenvolgende octaven[23, p. 6]

Uit deze DOG (Difference Of Gaussian) worden de lokale extrema bepaald door een punt met zijn acht buren te vergelijken en dit voor de vorige en volgende schaal te doen zoals in Figuur 26. Deze punten worden vervolgens als potentiele *keypoints* beschouwd [23].

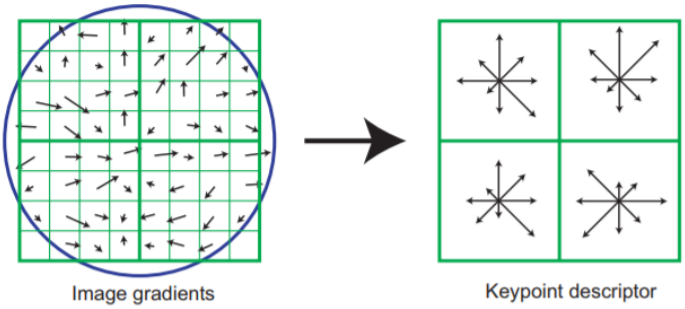


Figuur 26: DOG extremadetectie door pixels over verschillende schalen te vergelijken [23, p. 7]

In de tweede stap, *Keypoint localization*, worden de extrema uit de vorige stap nauwkeuriger gelokaliseerd. Deze stap zorgt ervoor dat keypoints met een laag contrast of keypoints die op een rand liggen; geëlimineerd worden en enkel sterke kandidaten resteren [23].

In de derde stap, *Orientation assignment*, zal een rotatie aan een punt worden gegeven om ze rotatie-invariant te maken. Dit wordt gedaan door de gradiënt van het extremum en zijn omliggende buren te berekenen. Met behulp van deze gradiëntoriëntaties wordt een oriëntatiehistogram opgesteld waaruit de piek en elke lokale piek boven de 80% genomen wordt om een keypoint met een oriëntatie te maken. Dit zorgt voor meerdere keypoints op dezelfde locatie met verschillende oriëntaties, deze hebben een sterk positief effect op de stabiliteit van het matchen [23].

In de laatste stap, *Keypoint descriptor*, worden de keypoints invariant gemaakt tegen variaties in belichting en verplaatsing. Dit wordt gedaan door in een 16 X 16 buurt rond het keypoint de gradiëntoriëntaties te berekenen van elke 4 X 4 blokken zoals in Figuur 27 (links) geïllustreerd wordt. Die worden dan naar een 2 X 2 descriptor omgevormd door de vectoren in dezelfde richting op te tellen. Het resultaat hiervan is in Figuur 27 (rechts) terug te vinden. Hierdoor kunnen keypoints over verschillende frames beter matchen daar hun omgeving ook in acht wordt genomen [23].

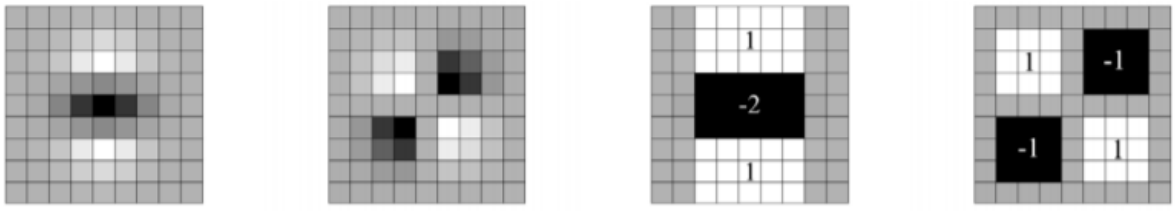


Figuur 27: Maken van de Keypoint descriptor [23, p. 15]

SURF *(Speeded-Up Robust Features)*

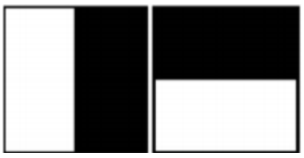
Aangezien de bovenvermelde techniek, de SIFT, relatief traag is was er nood aan een snellere versie, en daarom ontstond er SURF ("a speeded-up version of SIFT" [24]). Dit zonder de kwaliteit van de detector te verminderen. Bij SURF zal de aanpak voor het vinden van extrema

versimpeld worden door namelijk in plaats van de Gaussische filters (Figuur 28 - links), box filters (Figuur 28 - rechts) te gebruiken. Deze box filters worden voor verschillende schalen toegepast. Dit in combinatie met het integraalbeeld (zie 2.1.1.1) zorgt voor een snellere aanpak van de detector voor keypoints. Dit in combinatie met het gebruik van de determinant van de Hessiaan om zowel de schaal als locatie te vinden maakt het sneller dan de SIFT [25].



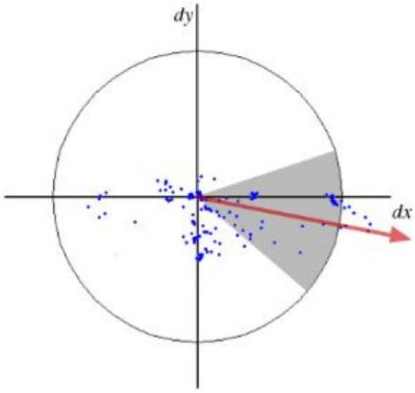
Figuur 28: (links) gaussiaanse 2de orde afgeleide voor y en xy, (rechts) aanpak met box filters [25, p. 5]

Voor de oriëntatie worden eerst de resultaten van de afbeeldingen na een Haar-wavelet filter voor de x- en y-richting (Figuur 29) berekend. Ook hier wordt het integraalbeeld aangewend om de berekeningen te vereenvoudigen en te versnellen. Daarbovenop zal door het toepassen van een Gaussische functie op deze resultaten, gecentreerd op het keypoint, de punten dichterbij het keypoint zwaarder doorwegen [25].



Figuur 29: Haar-wavlet gebruik in SURF [25, p. 6]

Door de Haar-wavelet filter ontstaan voor elk punt twee waardes, één voor de x- en één voor de y-richting. Vervolgens worden deze waardes geplot in een assenstelsel zoals in Figuur 30. Hieruit wordt de dominante vector berekend door een vector te maken van de som van de reeds bekomen x- en y-waardes die in een veld van 60° liggen (Figuur 30). De vector van de oriëntatie is de grootst bekomen vector die op voorgaande beschreven methode berekend wordt. Het is ook mogelijk om de oriëntatie bij SURF te verwaarlozen om zo de detectie sneller te laten verlopen, dit wordt U-SURF genoemd. U-SURF is handig voor toepassingen waarin niet geroteerd hoeft te worden en is door het verwaarlozen van de oriëntatie ook sneller dan SURF [24] [25].



Figuur 30: Punten in de vectorruimte na de Haar-wavelet filters [24, p. 1]

Voor de descriptor neemt de SURF analoog met SIFT een regio in de buurt van het keypoint, maar bij SURF zal deze regio georiënteerd worden naar de grootste oriëntatiewaarde van het keypoint. De regio wordt vervolgens in 4 X 4 deelregio's opgesplitst waarvan de resultaten van de horizontale en verticale Haar-wavelet (Figuur 29) genomen worden. Vervolgens wordt voor elke deelregio een vector volgens formule (2.11) berekend, dat bijgevolg neerkomt op 64 dimensies. Er bestaat echter ook een *extended* versie van SURF waarin er 128 dimensies zijn [24] [25].

$$v = (\sum dx, \sum dy, \sum |dx|, \sum |dy|) \qquad (2.11)$$

BRIEF *(Binary Robust Independent Elementary Features)*

BRIEF is een aanpassing op de voorgaande descriptoren van SIFT en SURF. Aangezien deze met 64/128-dimensionale vectoren werken, is er veel data, rekenkracht en bijgevolg tijd vereist om het uit te voeren. In BRIEF worden dus regio's effectief geclassificeerd aan de hand van een relatief klein aantal locatieparen. Deze paren geven 1 of 0 als waarde waardoor ze bitvectoren worden genoemd en bijgevolg weinig geheugen innemen. Op welke wijze deze paren worden geselecteerd, wordt in [26] beschreven. Aan de hand van deze paren gebeurt het matchen van afbeeldingen veel sneller [26]. Dit algoritme zoekt dus geen features of keypoints, maar is een snellere methode om regio's te beschrijven aan de hand van parameters (vectoren) [27].

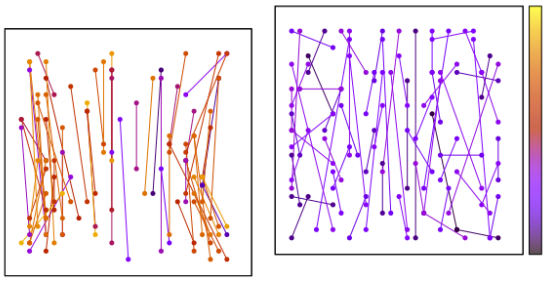
ORB *(Oriented FAST and Rotated BRIEF)*

ORB stelt een rekenkundig efficiëntere vervanger voor van SIFT met een evenwaardige kwaliteit en daarbovenop een betere bestendigheid tegen ruis en bruikbaar voor real-time applicaties. Dit door een FAST-detector te gebruiken en een BRIEF-descriptor, hiervan is ook de naam afkomstig. Voor detectie van keypoints wordt een FAST-detector gebruikt (zie 2.1.3.2 het FAST-algoritme), maar deze geeft geen waarde voor de kwaliteit van de keypoints. Daarom wordt er een Harris Corner Measure (analoog met Harris Corner Detection maar voor

enkel de keypoints) op toegepast om ze op deze manier te rangschikken. Uit deze rangschikking worden dan de beste N-aantal keypoints gekozen. Dit gebeurt op verschillende schaalniveaus om zo manier schaal-invariante keypoints te verkrijgen [28].

FAST is niet rotatie-invariant en daarom wordt er gebruik gemaakt van een intensiteit gewogen zwaartepunt C van zijn regio met het keypoint in het midden. De beweging van het keypoint tot dit zwaartepunt wordt vervolgens als vector voor de oriëntatie beschouwd. Om het nog meer rotatie-invariant te maken, wordt voor elke pixel in de regio (het is een cirkel dus binnen straal r) het moment berekend. Dus hoe dichter C bij het keypoint ligt, hoe onstabieler de meting is [28].

ORB maakt gebruik van een BRIEF-descriptor, maar deze is niet rotatie-invariant. Daarom zal de BRIEF gestuurd (steered BRIEF) worden in de richting van de oriëntatie, dat tot betere resultaten leidt. Maar uit de analyse van BRIEF en steered BRIEF blijkt dat er zeer veel correlatie is en dus ook weinig variatie bij die binaire testen. Nu wordt er gezocht naar 256 niet- gecorreleerde en gevarieerde bittesten om zo een beter descriptor te bekomen, deze wordt de rBRIEF genoemd. Het resultaat hiervan wordt afgebeeld in Figuur 31 met de kleur als waarde voor de correlatie (geel is veel, zwart weinig). Links in Figuur 31 is veel correlatie vast te stellen, maar door dit door een leeg algoritme te voeren bekomt men een vermindering van correlatie, dat in Figuur 31 (rechts) wordt getoond [28].



Figuur 31: Grafische weergave van de correlatie tussen verschillende binaire paren [28, p. 5]

2.1.3.3 ArUco-module

ArUco is een module in OpenCV dat gebruikt wordt voor markerdetectie. Dit betekent dat de te detecteren objecten gekend zijn door het programma, dat de detectie vereenvoudigd. De ArUco-module onderscheidt zich van andere markerdetectoren omdat het zich focust op poseschatting van voorwerpen in een 3D-wereldassenstelsel aan de hand van een 2D-beeldprojectie. Dit is mogelijk door gebruik te maken van markers waarvan de afmetingen (*ground truth*) gekend zijn. Deze markers zijn opgeslagen in bibliotheken die deel uitmaken van de ArUco-module. De ArUco-module bevat vier verschillende soorten van detectiemarkers: de ArUco-marker, het ArUco-paneel, het ChArUco-paneel en de diamantmarker. Een uitvoerige beschrijving van het volledige detectieproces van deze ArUco-

module kan in [29] teruggevonden worden. De grote voordelen van de ArUco-module is dat het zeer gebruiksvriendelijk is en alle programma's voor gebruik ter beschikking zijn gesteld in de module en ook terug te vinden op Github.

De eerste stap is de marker maken en uitprinten, vervolgens dient de camera gekalibreerd te worden (zie 2.1.3.3 camerakalibratie). Deze kalibratie geschiedt eveneens in OpenCV. De derde en laatste stap is de gekalibreerde camera gebruiken om het gekozen markertype te detecteren.

Camerakalibratie

Het uitvoeren van de camerakalibratie is noodzakelijk. Hiermee worden de inwendige parameters van de camera geschat, meer bepaald het beeldcentrum, de brandpuntsafstand en de parameters van de lensvervorming. Uiteraard zijn de afwijkingen bij zeer kwalitatieve camera's en lenzen gering tot zelfs nul, maar voor relatief goedkopere camera's en lenzen zijn deze afwijkingen wel significant. Deze afwijkingen maken echter wel een groot verschil bij het schatten van de locatie en stand van een voorwerp in het 3D-assenstelsel. Daarom moeten deze bepaald worden zodat ze gecompenseerd kunnen worden [30].

Zoals eerder vermeld bevat de ArUco-module een programma voor camerakalibratie. Dit programma maakt gebruik van het schaakbordpatroon (zie Figuur 32) om de parameters te schatten. De afmetingen van het schaakbordpatroon worden gemeten en in het programma ingevoerd zodat het deze informatie kan gebruiken. Vervolgens heeft het programma nood aan verschillende afbeeldingen van dit schaakbordpatroon in verschillende standen en posities (maar moet altijd volledig in beeld blijven) zoals in Figuur 32. Deze afbeeldingen worden dan verwerkt door de hoekpunten te zoeken (Figuur 33) en zo de onderlinge relatie tussen de hoekpunten te achterhalen. Uit de berekende relatie tussen de hoekpunten uit verschillende afbeeldingen en de afmetingen van het schaakbordpatroon kan het programma de parameters schatten.

Figuur 32: Camerakalibratie met behulp van het schaakbordpatroon [30, p. 18]

Figuur 33: Camerakalibratie met behulp van het schaakbordpatroon met hoekpuntdetectie [30, p. 19]

ArUco-markers

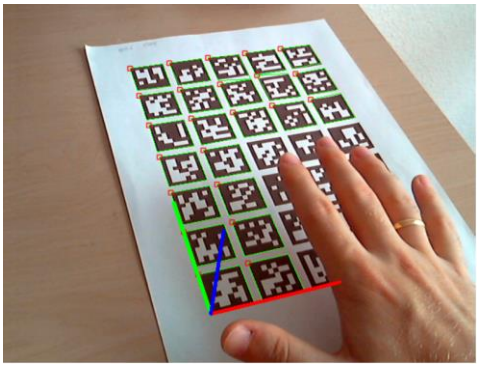
De ArUco-markers vormen de basis (Figuur 34), de andere drie methodes maken gebruik van deze markers maar elk op een andere manier. Zoals eerdergenoemd zijn deze markers ter beschikking in de bibliotheken behorende bij de ArUco-module. Deze bibliotheken verschillen onderling in twee aspecten: het aantal en de grootte (aantal pixels) van de markers die ze bevatten. Zo bestaat er de 4X4_50-, 4X4_100-, en de 5X5_50-bibliotheek, waarin 4X4/5X5 de grootte (4X4 pixels) en 50/100 het aantal markers aanduidt. Deze markers zijn uniek.

Figuur 34: 5X5 Aruco-markers [31, p. 3]

ArUco-paneel

Het principe van dit type paneel stamt af van de ArUco-markers, maar in plaats van één marker te gebruiken om een vlak te detecteren zullen er meerdere toegepast worden. Analoog aan de ArUco-markers zijn de markers en hun configuratie op het paneel gekend. Dit maakt het detectieproces robuuster en zorgt ervoor dat het paneel deels bedekt kan worden zonder dat dit de detectie verhindert (zie Figuur 36). Het grote verschil tussen dit paneel en een set van onafhankelijke markers is dat de relatieve posities van de markers op het paneel gekend zijn. Dit laat toe om de hoeken van alle markers te gebruiken om de pose van het paneel te schatten [32]. Voor het maken van het ArUco-paneel kan er net zoals bij de ArUco-markers gekozen worden uit verschillende bibliotheken met als voorbeeld het ArUco-paneel uit de 6X6-bibliotheek, getoond in Figuur 35.

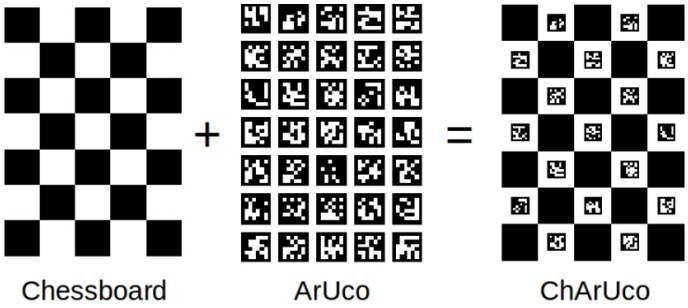
Figuur 35: ArUco-paneel uit de 6X6-bibliotheek [32]



Figuur 36: ArUco-paneel met occlusies [32]

CharUco-paneel

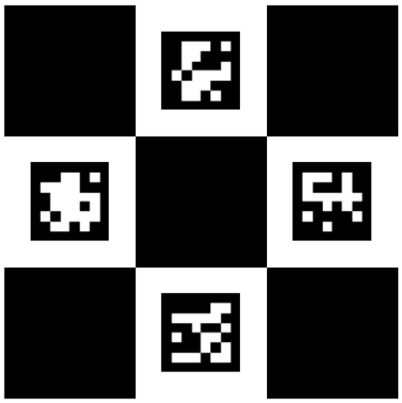
Het CharUco-paneel combineert de voordelen van het ArUco-paneel en het schaakbordpatroon om elkanders nadelen te compenseren. Daarom is het ChArUco-paneel de samenstelling van deze twee markertypes (zie Figuur 37). Het nadeel van het ArUco-paneel is dat het vinden van de hoekpunten (hun positie) geen hoge nauwkeurigheid heeft ter vergelijking met het schaakbordpatroon. Dit komt omdat de hoekpunten van een schaakbordpatroon omgeven zijn door twee zwarte vierkanten. Maar het nadeel van een schaakbordpatroon is dat het niet zo uniek is als het vinden van een ArUco-paneel en dat het geen occlusies toelaat [33].



Figuur 37: ChArUco-paneel [33]

Diamantmarkers

Het vierde en laatste type compenseert de nadelen verbonden aan het ArUco- en ChArUco-paneel, waarbij de meeste voordelen van het ChArUco-paneel behouden blijven. Het grote nadeel van de twee panelen is dat ze uniek zijn, dat betekent dat ze maar éénmaal gedetecteerd kunnen worden in een afbeelding. Het programma zal bij het vinden van een marker automatisch veronderstellen dat het deel uitmaakt van het paneel en bijgevolg bij het gebruik van twee dezelfde panelen niet weten welke het moet gebruiken. Dit probleem zal bij een diamantmarker niet voorkomen daar hun detectie afhankelijk is van de relatieve positie van de ArUco-markers. Deze zijn bijgevolg niet uniek. Door deze extra complexiteit zijn de diamantmarkers gelimiteerd tot een grootte van 3X3 vierkanten zoals Figuur 38 weergeeft. Hierdoor zal elke marker bestaan uit vier hoekpunten die overeenkomen met de hoekpunten van het schaakbordpatroon (de vier hoeken van het middelste vierkant) [34].



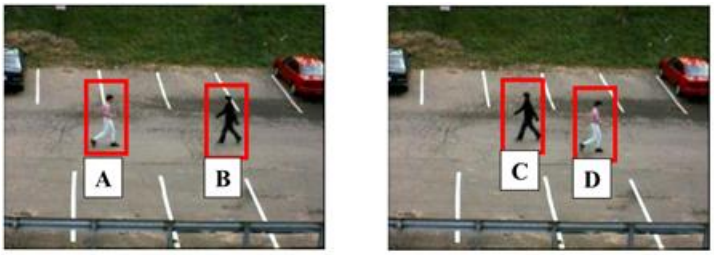
Figuur 38: Diamantmarker [34]

2.2 Tracking

Dit hoofdstuk behandelt verschillende bestaande algoritmes en filters voor het tracken van objecten of personen. Maar om deze algoritmes en filters goed te begrijpen, komt eerst een beknopte uitleg over het probabilistische model aan bod.

2.2.1 Het probabilistische model

Kansenberekening vormt een belangrijk onderdeel van het tracking- en data-associatieproces. Een kans geeft de mogelijkheid aan dat één object in twee opeenvolgende frames hetzelfde is. Dus als men twee foto's zeer snel (enkele milliseconde) achter elkaar maakt van een wandelende man zal aan de hand van een kans (getal tussen 0 en 1) uitgedrukt zijn in hoeverre de gefotografeerde mannen in de twee opeenvolgende foto's op elkaar lijken. Voor Figuur 39 betekent dit dat er zes kansen bestaan: dat persoon C persoon A is, dat persoon D persoon A is, dat persoon C persoon B is, dat persoon D persoon B is of zelfs dat D en/of C een nieuwe persoon zijn.



Figuur 39: (a) foto 1 met object A en B, (b) foto 2 met object C en D [2, p. 21]

Zoals eerder vermeld wordt de overeenkomstigheid of gelijkenis van objecten over twee frames beschreven met een kans. Wiskundige wordt dit beschreven als $Pr(x = x_i)$ of $P(x_i)$ met x de verzameling van alle mogelijkheden en x_i een element uit deze verzameling. $P(x_i)$ is bijgevolg de kans dat x de waarde x_i bezit, waarbij deze waarde zich tussen 0 en 1 bevindt omdat de som van alle kansen gelijk aan 1 dient te zijn.

$$\sum_x P(x) = 1 \qquad (2.12)$$

Aangezien $P(x_i)$ hetzelfde is als $Pr(x = x_i)$, kan $Pr(y = y_i)$ ook geschreven worden als $P(y_i)$. Als men nu de kans zoekt dat $x = x_i$ en $y = y_i$ spreekt men over de doorsnede tussen x_i en y_i of in het Engels de *joint probability* genaamd. Dit schrijft men wiskundig als $P(x_i \cup y_i) = P(x_i, y_i)$. Maar in het geval dat x_i of y_i gekend is, zal men gebruik maken van de voorwaardelijke kans. Deze kans beschrijft de kans van één gebeurtenis als de andere gebeurtenis gekend is. Dus $P(x = x_i \, als \, y = y_i)$ en wiskundig geschreven als $P(x_i|y_i)$. De voorwaardelijke kans wordt bepaald uit de joint probability en de kans op de gekende gebeurtenis.

$$P(x_i|y_i) = \frac{P(x_i,y_i)}{P(y_i)} \quad (2.13)$$

In het geval dat de x- en y-waardes onafhankelijk zijn van elkaar, zal $P(x_i|y_i) = P(x_i)$. De marginale kans is de kans op één gebeurtenis terwijl de informatie van de andere gebeurtenis volledig genegeerd wordt.

$$P(x) = \sum_y P(x,y) \quad (2.14)$$

Een belangrijke regel in kansrekening is de regel van Bayes. Deze verkrijgt men door de joint probabilities aan elkaar gelijk te stellen, maar dan uitgewerkt op twee manieren voorgesteld in formule (2.15) en formule (2.16).

$$P(x,y) = P(x|y)\,.P(y) \quad (2.15)$$

$$P(x,y) = P(y|x)\,.P(x) \quad (2.16)$$

Regel van Bayes:
$$P(x|y) = \frac{P(y|x)\,.\,P(x)}{P(y)} \quad (2.17)$$

Opmerking: in formule (2.17) kan $P(y)$ beschouwd worden als de normalisatiefactor. Dit kan bewezen worden door de marginale kans van y te nemen en hier de voorwaardelijke kans op het rechterlid toe te passen. Zo wordt de teller uit de regel van Bayes bekomen.

$$P(y) = \sum_x P(x,y) = \sum_x P(y|x)\;.P(x) \quad (2.18)$$

Wanneer een derde parameter mee in rekening genomen wordt, ziet de regel van Bayes als volgt uit:

$$P(x|y,z) = \frac{P(y|x,z)\,.\,P(x|z)}{P(y|z)} = \gamma\,.P(y|x,z)\,.\;P(x|z) \quad (2.19)$$

Met de kennis van het probabilistische model is het mogelijk om de kansdichtheidsfunctie op te stellen. Dit is een kansdichtheidsfunctie en berekent de kans voor alle mogelijke waardes van x. Met randvoorwaardes dat de kans altijd groter of gelijk aan nul is, en dat de som van de kansen voor alle willekeurige waardes voor x altijd gelijk is aan 1 voor reële getallen.

$$f(x) \geq 0 \quad voor\ elke\ x \quad (2.20)$$

$$\int_{-\infty}^{\infty} f(x)\,dx = 1 \quad (2.21)$$

Dit betekent dat als de grenzen verlegd worden, de kans op het vinden van gebeurtenis x tussen 0 en 1 zal liggen. Als we nu als interval $[x_a, x_b]$ nemen ziet het er als volgt uit:

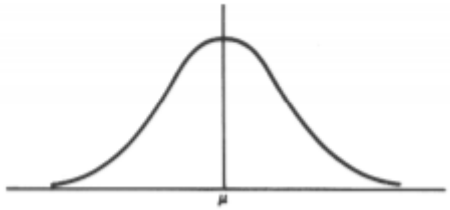
$$Pr(x \in [x_a, x_b]) = \int_{x_a}^{x_b} f(x)\,dx \quad (2.22)$$

Voor de kansdichtheidsfunctie f(x) wordt vaak de Gaussiaanse verdeling gekozen, omdat deze de volledige kansverdeling beschrijft met slechts twee parameters: het gemiddelde μ en de

variantie σ^2. De kans kan dan aan de hand van deze twee parameters met formule (2.23) berekend worden:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)} \text{ [35]} \qquad (2.23)$$

Figuur 40 stelt de vorm van een Gaussiaanse verdeling voor. Met een praktische eigenschap dat 68% van de mogelijke waarnemingen vallen binnen het gebied $\mu \pm \sigma$, 95% binnen de grenzen $\mu \pm 2\sigma$ en 99% binnen de grenzen $\mu \pm 3\sigma$ [36].



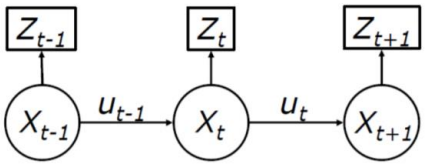
Figuur 40: Grafiek van de Gaussiaanse verdeling [35, p. 4]

2.2.2 Algoritmes en filters

Hieronder zijn verschillende algoritmes en filters uitgelegd die gebruik maken van de formules uit de bovenstaande kansberekening. Tot slot worden de trackers besproken die in OpenCV geïntegreerd zijn.

2.1.4.2 Bayes filter

Als eerste filter wordt er gestart met de Bayes filter, het schema hiervan wordt getoond in Figuur 41, met x_{t-1} , x_t en x_{t+1} de locatie van het object, z_{t-1} , z_t en z_{t+1} de waarde van de metingen, 't' staat voor het huidige tijdstip, 't-1' de vorige tijdstap en 't+1' de volgende tijdstap (voorspelling).



Figuur 41: Bayes filter [36, p. 63]

Bij het berekenen van een toekomstige locatie wordt de onzekerheid op de huidige locatie in rekening gebracht, welke op zijn beurt afhankelijk is van de onzekerheid op de vorige locatie enzovoort. Dit is uiteraard geen goede eigenschap voor een locatiebepaling omdat alle historische data hierdoor bijgehouden dient te worden, dat het een geheugenintensief systeem maakt. Dit nadeel verdwijnt door het toepassen van de Markov-assumptie: "given the present, the future does not depend on the past" [36]. Dit betekent dat om de toekomstige locatie te

bepalen en we de huidige locatie kennen, het verleden geen invloed meer heeft. Dit omwille van het feit dat de huidige locatie al afhankelijk is van het verleden. Aangezien we de positie op tijdstip k willen kennen, moeten we deze schatten. Zoals Figuur 41 aangeeft, is de positie afhankelijk van alle opeenvolgende bewegingen u en de metingen z. Deze schatting wordt het *belief* genoemd [36].

$$bel(x_k) = P(x_k|u_1 \dots u_k, z_1 \dots Z_k) \qquad (2.24)$$

Door hier de regel van Bayes (2.17) op toe te passen bekomt men:

$$Bel(x_k) = \gamma \, . P(z_k|u_1 \dots u_k, z_1 \dots Z_{k-1}, x_k) \, . \; P(x_k|u_1 \dots u_k, z_1 \dots Z_{k-1}) \qquad (2.25)$$

De Markov-assumptie toepassen geeft:

$$Bel(x_k) = \gamma \, . P(z_k|x_k) \, . \; P(x_k|u_1 \dots u_k, z_1 \dots Z_{k-1}) \qquad (2.26)$$

Nog steeds hangt het belief van metingen en historische data af, dit is te vermijden door de tweede term van de oplossing te herschrijven. Dit kan door hier formule (2.7) van de marginale kans op toe te passen en vervolgens de conditionele kans:

$$P(x_k|u_1 \dots u_k, z_1 \dots Z_{k-1}) = \sum_{x_{k-1}} P(x_k, x_{k-1}, u_1 \dots u_k, z_1 \dots Z_{k-1}) \qquad (2.27)$$

$$= \sum_{x_{k-1}} P(x_k|x_{k-1}, u_1 \dots u_k, z_1 \dots Z_{k-1}) \, . P(x_{k-1}|u_1 \dots u_k, z_1 \dots Z_{k-1}) \qquad (2.28)$$

De Markov assumptie toepassen geeft:

$$P(x_k|u_1 \dots u_k, z_1 \dots Z_{k-1}) = \sum_{x_{k-1}} P(x_k|x_{k-1}) \, . P(x_{k-1}|u_1 \dots u_k, z_1 \dots Z_{k-1}) \qquad (2.29)$$

Ook ditmaal is de tweede term van de oplossing afhankelijk van het verleden. Dit is logisch daar dit het belief van de huidige locatie is.

$$P(x_{k-1}|u_1 \dots u_k, z_1 \dots Z_{k-1}) = \; Bel(x_{k-1}) \qquad (2.30)$$

Bij het invullen van de tweede term verkrijgt men:

$$Bel(x_k) = \gamma \; . P(z_k|x_k) . \; \sum_{x_{k-1}} P(x_k|x_{k-1}) Bel(x_{k-1}) \qquad (2.31)$$

Met y de normalisatiefactor en $Bel(x_{k-1})$ het geloof van de vorige locatie. Formule (2.31) is op te splitsen in twee onderdelen, de voorspellingsstap $\sum_{x_{k-1}} P(x_k|x_{k-1}) Bel(x_{k-1})$ en de correctiestap $\gamma \, . P(z_k|x_k)$. Waarbij de voorspellingsstap de inputs van bijvoorbeeld encoders en de onzekerheid hiervan zal gebruiken om de locatie te voorspellen en de correctiestap de voorspelde locatie corrigeert met de waarde(s) van de meting en de onzekerheid op de meting.

2.1.4.3 De (Extended) Kalman-filter

De Kalman-filter heeft dezelfde aanpak als de Bayes filter. Ook hier is er gebruik gemaakt van de predictie- en correctiestap. Het grote verschil tussen de Kalman en de standaard Bayes filter is de *Kalman gain*. Deze factor bepaalt de juistheid van de meting en de voorspelling. Deze factor wordt bepaald door de verhouding van de onzekerheden van de voorspelling ten opzichte van de meting. De voorspelling wordt gedaan door een bewegingsmodel op te stellen voor het object, met dit bewegingsmodel kan vervolgens een geschatte waarde bekomen worden.

$$x_k^{pre} = f(x_{k-1}^{pre}) \qquad (2.32)$$

Met x_k^{pre} de voorspelde waarde en x_{k-1}^{pre} de vorige waarde en $f(x)$ de functie voor het bewegingsmodel. Dit bewegingsmodel kan, voor lineaire verbanden tussen de locatie, een transformatiematrix zijn. Voor het geval van tracking zal er een niet-lineair verband zijn. Daarom wordt er gebruik gemaakt van de Extended Kalman-filter. Want bij deze voorspelling heerst een extra onzekerheid C_k^{pre}, deze verdeling zal in eerste instantie niet gekend zijn en daarom bij het tracken van een nieuw object uit willekeurige waardes bestaan. Maar deze kansverdeling wordt bij elke stap aangepast met formule (2.33).

$$C_k^{pre} = F_k \, C_{k-1}^{co} \, {F_k}^T + Q_k \qquad (2.33)$$

Met F_k de jacobiaan, C_{k-1}^{co} de vorige onzekerheid (na de correctiestap) en Q_k stelt fouten in de voorspelling voor met een Gaussiaans verloop. Doordat de kansverdeling elke keer wordt aangepast, wordt deze steeds beter. Uiteraard zal de waarde voor de predictie in het begin zeer slecht zijn en daarom mogelijk de geschatte waarde ver van de werkelijke waarde plaatsen. Maar ook dit probleem wordt door de Kalman gain verholpen omdat deze afhankelijk is van de verhouding tussen de twee onzekerheden, dit wordt duidelijker in de correctiestap.

De tweede stap is de correctiestap. Hierbij zal de voorspelling gecorrigeerd worden op basis van de resultaten van de meting. Om dit uit te voeren wordt er gebruik gemaakt van de Kalman gain, zoals hierboven vermeld is dit de verhouding tussen de twee onzekerheden.

$$x_k^{bel} = x_k^{pre} + K_k \, y_k \qquad (2.34)$$

$$y_k = x_k^{me} - H_k x_k^{pre} H_k^T \qquad (2.35)$$

Met K_k de Kalman gain, y_k het verschil tussen de meting en de predictie, x_k^{me} de waarde van de meting en H_k de transformatiematrix. Deze transformatiematrix is nodig omdat de metingen en voorspelling een verschillend aantal parameters hebben. Hetzelfde probleem doet zich voor bij het optellen van de twee onzekerheden, daarom zal hier ook gebruik gemaakt worden van H_k.

$$S_k = H_k\, C_k^{pre} H_k^T + C_k^{me} \quad (2.36)$$

$$K_k = \frac{C_k^{pre}}{S_k} \quad (2.37)$$

$$C_k^{co} = (1 - K_k\, H_k) * C_k^{pre} \text{ [37]} \quad (2.38)$$

$$C_k^{co} = (I - K_k\, H_k) * C_k^{pre} \text{ [2]} \quad (2.39)$$

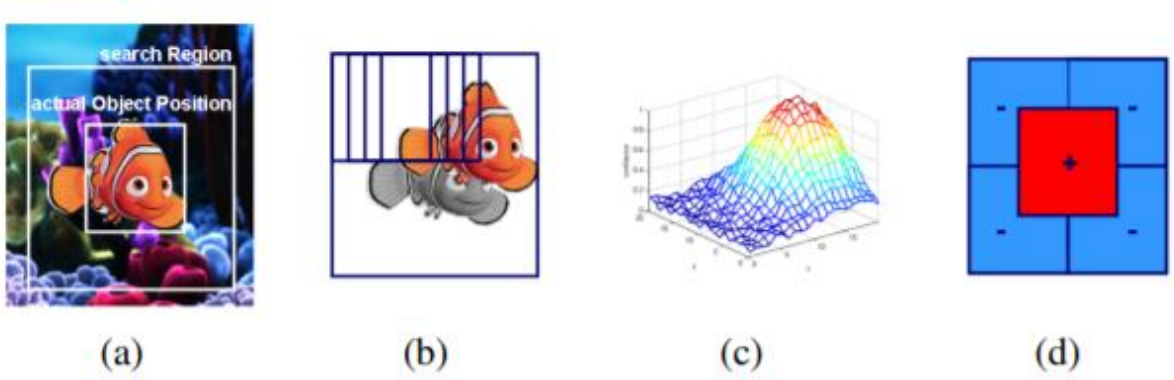
Met S_k de som van de twee onzekerheden en C_k^{co} de onzekerheid op x_k^{bel}. Voor C_k^{co} zijn er twee formules die mogelijk zijn: (2.38) en (2.39) [37] [2].

2.2.3 OpenCV-trackers

In OpenCV zijn verscheidene extra modules aanwezig, waaronder één module die een verzameling van trackers bevat. Deze bevat momenteel acht trackers: BOOSTING, MIL, KCF, TLD, MEDIANFLOW, GOTURN, MOSSE en CSRT.

2.2.3.1 BOOSTING

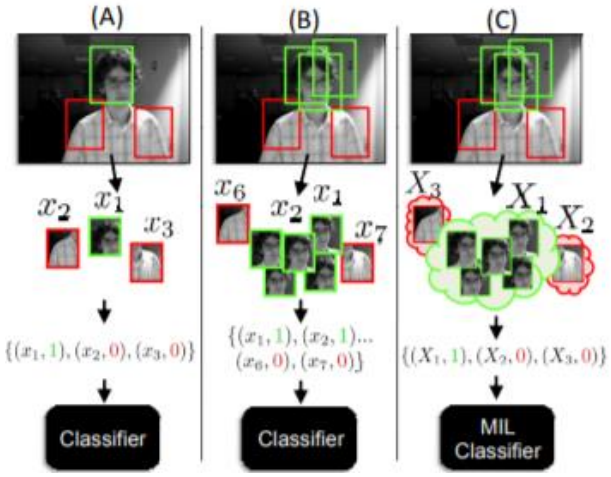
De BOOSTING-tracker is een real-time methode voor het volgen van objecten, gebaseerd op een nieuwe onlineversie van het AdaBoost-algoritme. Het tracken van een object kan gezien worden als een classificatie, deze classificatie moet het onderscheid kunnen maken tussen de achtergrond en het object zelf. Het object is dus al gedetecteerd door een detector, dit object/beeldgebied is dan een positief beeldvoorbeeld voor de tracker en tegelijkertijd neemt men beelden van de achtergrond rondom het object zodat deze kunnen dienen als negatieve voorbeelden. Deze voorbeelden dienen om de tracker te initialiseren en de classificator op te bouwen. Bij een volgende frame zal de tracker aan *template tracking* doen, hierbij zal de classificator alle interessante regio's evalueren. Dit geeft een betrouwbaarheidswaarde voor elke positie als resultaat en deze allemaal samen vormt de betrouwbaarheidsmap (Figuur 42c). Uit deze map wordt het maximum berekend om zo de meest waarschijnlijke locatie van het object te vinden. Ten slotte is de classificator geüpdatet om zich zo aan te passen aan veranderingen van de verschijning van het object. Dit zorgt ervoor dat het systeem veel robuuster is [38]. Deze vier stappen worden telkens herhaald en zijn in Figuur 42 geïllustreerd. Het gebruik van een bewegingsmodel zorgt voor een kleinere zoekregio van de tracker.



Figuur 42: De vier stappen van de BOOSTING-tracker. (a) initiële positie van het object, (b) zoeken in de zoekregio bij volgende frame, (c) de betrouwbaarheidsmap, (d) update van de classificator [38, p. 3]

2.2.3.2 MIL *(Multiple Instance Learning)*

Deze tracker focust zich op de aanpasbaarheid van de verschijning van het object. De voorgaande tracker heeft als nadeel dat ze niet heel robuust zijn doordat ze enkel het verschijningsmodel aanpassen met de huidige locatie. Ze zullen foutief blijven tracken in het geval dat de tracker een fout maakt en het verschijningsmodel hierdoor foutief updaten. Bij MIL zal dit niet gebeuren omdat er een groep positieve beeldvoorbeelden worden doorgegeven in plaats van één positief beeldvoorbeeld. Een dergelijke groep wordt een bag genoemd en bevat voor een positieve bag minstens één positief exemplaar, een negatieve bag bevat er geen. De positieve bag wordt gemaakt door voorbeelden dicht rond het center van het positieve beeldvoorbeeld te nemen, deze overlappen dus met het positief voorbeeld. Dit proces en het verschil met voorgaande trackers is zichtbaar in Figuur 43. Door deze aanpassing moet het algoritme in de leerstap achterhalen welk voorbeeld het meest correct is. Dit maakt het trackingsproces veel robuuster [39].



Figuur 43: Updaten van het verschijningsmodel. (a) gebruik van één positief voorbeeld, (b) gebruik van verschillende positieve voorbeelden, (c) gebruik van één positieve bag [39, p. 1]

2.2.3.3 KCF *(Kernelized Correlation Filters)*

Ook bij deze tracker is het doel om een classificator te maken en te updaten die het verschil tussen het object en de achtergrond kan onderscheiden. Dit door niet enkel positieve beeldvoorbeelden van het object te nemen bij elke frame maar ook negatieve beeldvoorbeelden van de omgeving. Het aantal negatieve beeldvoorbeelden is hierbij omgekeerd evenredig met de snelheid van de tracker. Dus bij dit type tracking wordt er niet gebruik gemaakt van template tracking zoals bij MIL en BOOSTING waar een verschijningsmodel wordt toegepast en geüpdatet. Maar deze tracker maakt gebruik van een correlatiefilter die wordt getraind aan de hand van de positieve en negatieve beeldvoorbeelden [40]. Voor meer uitleg omtrent de werking van een kernelgebaseerde tracker wordt naar [41] verwezen.

2.2.3.4 MOSSE *(Minimum Output Sum of Squared Error)*

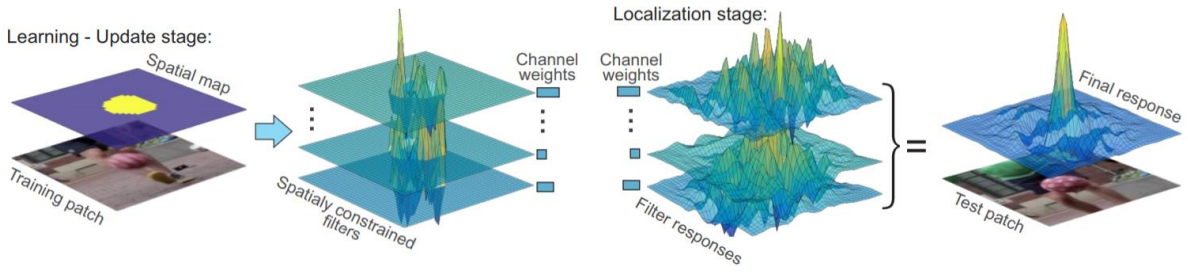
De MOSSE-tracker maakt gebruik van een aanpasbare correlatiefilter. Deze filter wordt initieel getraind op het object in het eerste frame (het object wordt door de gebruiker aangegeven). Vervolgens zal deze filter over het volgende frame geschoven worden om zo de correlatie map te maken. De locatie in deze map met de maximale waarde is de locatie van het object in het frame, met deze informatie kan de filter geüpdatet worden. De correlatiestap wordt in het Fourierdomein gedaan omdat hier een eenvoudige relatie tussen ingang en uitgang mogelijk is. Om een zo gewenst mogelijke correlatiemap te verkrijgen zal bij MOSSE de som van de kwadratische fout tussen de werkelijke output van de convolutie en de gewenste output van de convolutie geminimaliseerd worden. Dit betekent dat de tracker de afstand tussen een fout van een punt in de verkregen uitgang en het werkelijke punt van het object zal minimaliseren door de filter te updaten [42]. Deze tracker maakt nog extra gebruik van PSR (*The Peak-to-Sidelobe Ratio*), dit meet de sterkte van de correlatie piek. In het geval dat deze piek niet groot genoeg is, bijvoorbeeld bij occlusies of *tracking failure*, zal de tracker stoppen met online te updaten. Hierdoor gaat de informatie van het object niet verloren, en zal de tracker het object bij het terugkomen in beeld weer kunnen tracken [42].

2.2.3.5 CSRT *(Channel and Spatial Reliability tracker)*

Deze tracker heeft net zoals bij KCF en MOSSE een correlatiefilter, alleen wordt bij deze techniek de filter opgedeeld in verschillende kanalen. Elk van deze kanalen worden featurekanalen genoemd en een gewicht aan toegekend. Dit gewicht is de factor die bepaalt hoe belangrijk het featurekanaal is ten opzichte van de totale filter. De toevoeging van deze gewichten maakt het voor de leerstap/het updaten van de filter mogelijk om een betere fitting te krijgen. In een gewone correlatie filter zijn er meer lokale maxima aanwezig, dit bijvoorbeeld doordat het object features deelt met zijn omgeving. Door nu het gewicht van de featuremap die deze gedeelde features bevat te verkleinen zal de output correlatiemap duidelijker één piek tonen. Dit zorgt voor minder lokale maxima en dus minder verwarring voor de classificator.

Vervolgens zal de som van de kwadratische fout tussen de werkelijke output van de convolutie en de gewenste output van de convolutie beter geminimaliseerd worden [43].

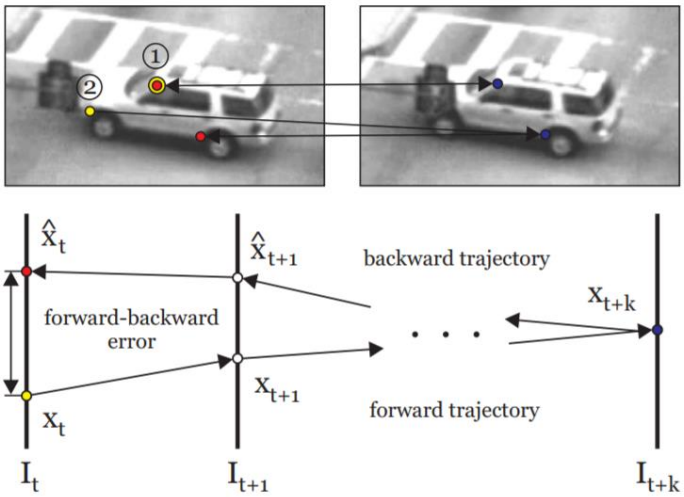
Zoals eerder vermeld worden de featurekanalen en hun gewichten in de leerstap geüpdatet, dit is geïllustreerd in Figuur 44 (links) en (rechts) zal de filter in zijn geheel (som van alle feature kanalen) gebruikt worden om het object in het volgende frame te lokaliseren [43].



Figuur 44: Leerstap/updatestap en lokalisatiestap van de CSRT-tracker [43, p. 2]

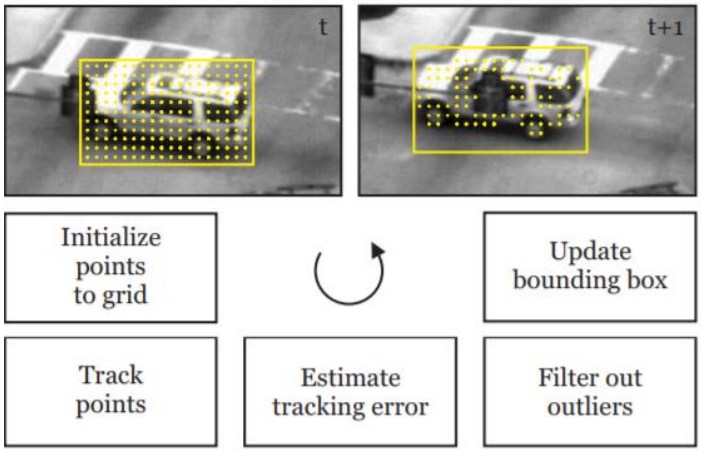
2.2.3.6 MEDIANFLOW

De MEDIANFLOW-tracker maakt gebruik van de *forward-backward error* om het falen van tracken te detecteren. Dit werkt als volgt, waarbij eerst de forwardstap plaatsvindt. Deze stap zal een featurepunt (maakt deel uit van het te tracken object) vooruit in de tijd tracken. Dan zal dit featurepunt in het laatste frame een valideringstraject creëren, op dit moment is het niet zeker of dit punt wel echt hetzelfde punt als in de vorige frame is (zie punt 2 in Figuur 45). Om het traject te valideren zal er aan backtracking worden gedaan, dit betekent dat ze terug in de tijd het punt uit het laatste object zullen tracken tot de eerste frame. In het geval dat dit punt wel correct is (zie punt 1 in Figuur 45) zal het forward- en backwardtraject hetzelfde zijn, en de fout tussen de twee gelijk aan nul. Als deze fout (de forward-backward error) te groot is, zal dit featurepunt in het laatste frame als incorrect beschouwd worden [44]. Om dit principe toe te passen op objecttracking in plaats van punttracking worden er verschillende punten in de bounding box genomen zoals geïllustreerd wordt in Figuur 46 (links). Deze punten worden dan aan de hand van Lucas-Kanade tracker gevolgd en de kwaliteit van dit wordt dan geschat en aan elk punt wordt een fout toegewezen. Van deze punten worden de 50% slechtste eruit gefilterd en de overblijvende punten worden gebruikt om de verplaatsing van de volledige bounding box te schatten. De overgebleven punten worden weergegeven in Figuur 46 (rechts). Deze verplaatsing van de bounding box wordt uitgevoerd door de mediaan van de overgebleven punten voor elke dimensie te bepalen [44]. De verschillende stappen van de Median Flow-tracker worden afgebeeld in Figuur 46.



Figuur 45: De Forward-Backward error[44, p. 1]

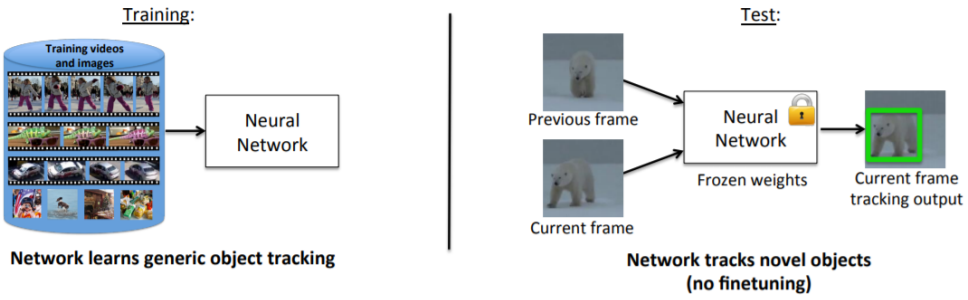
Schaalveranderingen worden berekend door voor elk paar van punten de verhouding tussen de afstand en de vorige afstand te berekenen. Ook voor de schaalverandering van de bounding box zal de mediaan van alle schaalverhoudingen (tussen elk paar punten) genomen worden [44].



Figuur 46: Stappen en verwerking van Median Flow [44, p. 3]

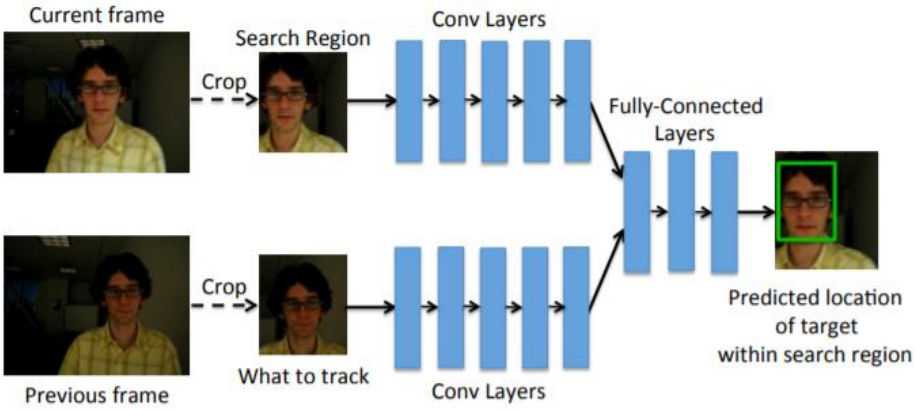
2.2.3.7 GOTURN

Deze tracker onderscheidt zich van de andere trackers aangezien het gebruik maakt van een neuraal netwerk. Dit neuraal netwerk wordt niet op dezelfde manier gebruikt als voor detectie waar ze op grote hoeveelheden offline data getraind worden. Bij de GOTURN *(Generic Object Tracking Using Regression Networks)* tracker is het mogelijk om een tracker te leren hoe een object beweegt door het te trainen op een grote hoeveelheid video's van bewegende voorwerpen. Dit wordt mogelijk gemaakt door een neuraal netwerk volledig offline te trainen en het neuraal netwerk vervolgens zo te gebruiken, dus zonder online updating zoals in Figuur 47 [45].



Figuur 47: Methode voor offline trainen (links) en werking zonder online updating (rechts) [45, p. 2]

Het offline trainen van deze tracker zorgt ervoor dat het programma veel sneller is dan gelijkaardige NN-trackers die online leren, aangezien dit een tijdrovend proces is. Een tweede verschil is dat dit neuraal netwerk geen gebruik maakt van een classificator om het te tracken object te herkennen, maar dat het gebruik maakt van een regressie-gebaseerde aanpak. Zo zal deze tracker enkel twee afbeeldingen doorheen het neuraal netwerk sturen, namelijk het te tracken object uit de vorige frame en een zoekregio uit het huidige frame. Dit en de structuur van het CNN wordt in Figuur 48 afgebeeld [45].

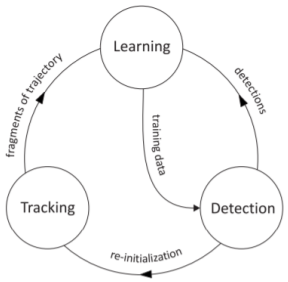


Figuur 48: Structuur van het CNN voor GOTURN [45, p. 4]

Door het convolutionele neuraal netwerk offline op video's van verschillende objecten te trainen, leert het CNN de generieke relatie tussen de beweging en verschijning van objecten. Deze relatie kan vervolgens gebruikt worden om nieuwe objecten te tracken zonder enige behoefte aan online training. Om het CNN duidelijk te maken welk object getrackt dient te worden, is het noodzakelijk dat één enkel object wordt doorgegeven aan het CNN (zie de Crop-versie van "Previous frame" in Figuur 48). Ook zal de omgeving van en rondom de voorgaande locatie dienen als voorstel voor de nieuwe locatie (zie de Crop-versie van "Current frame" in Figuur 48). Uiteraard zal voor snel bewegende objecten het voorstel groter moeten zijn en moet er mogelijk een bewegingsmodel geïnterpreteerd worden voor het voorstel. Bij occlusies zal de GOTURN-tracker in de problemen komen, maar dit kan opgelost worden door het te combineren met een online updatende detector [45].

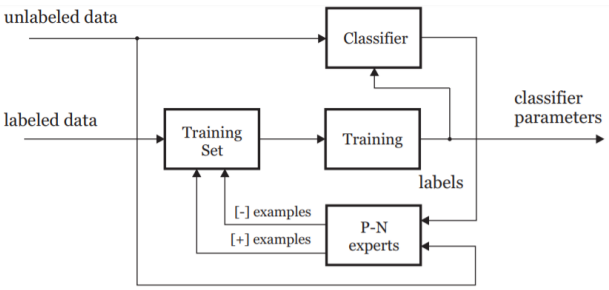
2.2.3.8 TLD

TLD *(Tracking-Learning-Detectie)* kan beschouwd worden als een kaderwerk voor een langetermijn tracker. Dit kaderwerk kan in drie delen worden opgedeeld: tracken, leren en detectie. Dit kaderwerk streeft naar het samenvoegen van de goede eigenschappen van de detector en de tracker. Het grote voordeel van een tracker is dat het sneller kan zijn dan een detector, maar er is sprake van afwijking van de voorspelling en de werkelijkheid. Ook faalt een tracker typisch wanneer het te volgen object uit het camerabeeld verdwijnt. Dit zijn net de sterktes van de detector, zij geven de locatie van het object voor elke frame terug zonder afwijking en zullen niet falen bij het verdwijnen van het object uit het camerabeeld. Maar hiervoor heeft de detector nood aan een offline trainingstap en kan dus geen ongekende objecten tracken. Daarom stelt het TLD-kaderwerk gelijktijdig lopende operaties op voor de drie onderdelen: tracking, learning en detectie. Met een tracker die het object van frame tot frame volgt, een detector die het object detecteert en de tracker opnieuw initialiseren zodat deze accurater is en ten slotte een leerstap die de fout van de detector voorspelt en updatet. Het blokschema van de drie stappen staat in Figuur 49 afgebeeld [46].



Figuur 49: Blokschema van het TLD-kaderwerk [46, p. 3]

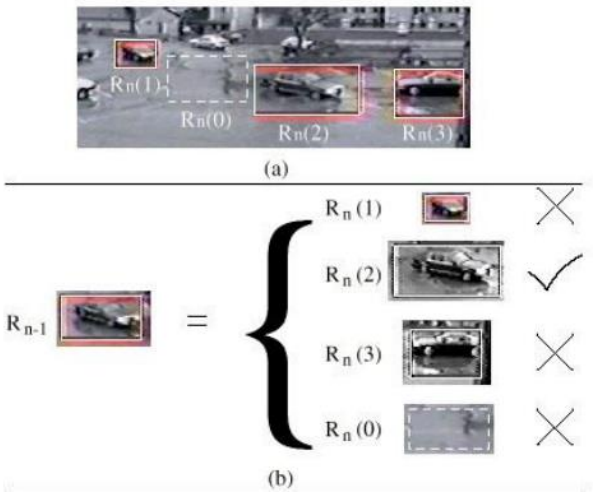
Voor de detector en tracker zijn er talloze mogelijkheden maar voor de leerstap wordt *P-N learning* voorgesteld. Deze evalueert de detector aan de hand van twee types van "experts", met de P-expert die gemiste detectie herkent en een N-expert die valse alarmen detecteert. P-N learning bestaat uit de volgende onderdelen: een classificator die geleerd moet worden, een training set met gelabelde data (door de tracker aangeleverd), een gesuperviseerde training die de classificator traint en een P-N-expert die positieve en negatieve trainingsvoorbeelden genereert. Deze onderdelen worden in het blokschema in Figuur 50 getoond [46].



Figuur 50: Blokschema van de P-N learning (learning-stap) [46, p. 4]

2.3 Data-associatie

Data-associatie speelt een grote rol voor de kwaliteit van het trackingsproces, dit omdat deze stap als doel heeft om de objecten doorheen verschillende frames te associëren op basis van één of meerdere criteria. In het voorbeeld weergegeven in Figuur 51 wordt een auto over twee beelden heen geassocieerd als zijnde dezelfde wagen op basis van zijn uiterlijk (*Appearance model*). Zoals te zien is, is het uiterlijk (wat de camera ziet) van de auto in R_{n-1}(het uiterlijk in de vorige frame) veranderd t.o.v. $R_n(2)$ (de auto in het huidige beeld). Maar omdat de auto in deze twee frames zeer weinig beweegt en daardoor het uiterlijk maar weinig verandert, zal deze auto geassocieerd kunnen worden over de twee frames heen.



Figuur 51: Data-associatie op basis van zijn uiterlijk [2]

In dit hoofdstuk zullen enkele algoritmes en associatiecriteria besproken worden. Er wordt wel een onderscheid gemaakt tussen algoritmes om MAP's (*multidimensional assignment problem*) op te lossen en andere algoritmes die de gegevens van de objecten en targets verwerken naar kansverdelingen.

2.3.1 Algoritmes

De volgende algoritmes lossen zelf niet het data-associatieprobleem op maar zorgen voor de verwerking van de data, nog specifieker ze verwerken de gekende trajecten en objecten om naar kansverdelingen. Zo zal *Nearest Neighbour algorithme (NN)* a.d.h.v. de locatie van de objecten en de vorige locatie van de targets kunnen bepalen wat de kans is voor elk object om geassocieerd te worden met een bepaald target.

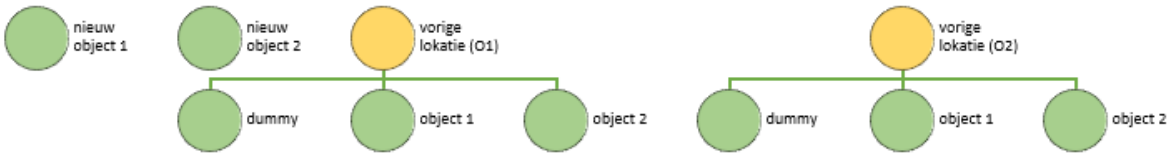
2.3.1.1 Multi Hypotheses Tracking (MHT)

De MHT associatietechniek is gebaseerd op het principe van een boomdiagram, zo zal dit data-associatiealgoritme aan de hand van een dergelijk diagram de link trachten te leggen tussen

elk object in het huidige frame en deze in het vorige frame. De sterkte van deze link wordt uitgedrukt in een reëel getal en is afhankelijk van de gebruikte associatiecriteria. Een boomstructuur is een manier om de verbanden tussen opeenvolgende frames te structuren, maar vervolgens moet de toewijzing van de objecten doorheen de frames gebeuren. Dit wordt uitgevoerd door een MAP-algoritme, het doel van dit algoritme is om de beste oplossing uit te rekenen. Het grote nadeel van dit algoritme is het feit dat het bijhouden van omvangrijke boomdiagrammen veel geheugen vereist [47]. Omdat bij elke stap het diagram exponentieel groeit, is het belangrijk om op tijd informatie te verwijderen. Maar eerst moet er gekeken worden welke stappen de MHT doorloopt om zo een boomdiagram op te bouwen.

Bij het maken en updaten van de boomdiagrammen worden er drie stappen ondernomen. Ten eerste zal voor elke observatie/object een nieuwe boom aangemaakt worden die de kans weergeeft dat dit object een nieuw object is. Vervolgens wordt elke bestaande boom geüpdatet met de observaties uit het nieuwe frame met de kans dat deze observaties dezelfde zijn. Deze stap kan gefilterd worden om zo het systeem minder geheugenintensief te maken. Ten slotte wordt er bij de update van de boom een dummy-tak toegevoegd, deze geeft de kans weer dat een object niet gedetecteerd werd [47].

Om dit te verduidelijken geeft Figuur 52 een voorbeeld weer voor het geval dat er in het vorige frame twee objecten waren en in het huidige frame ook twee. De objecten worden aangeduid met O1 en O2.



Figuur 52: Voorbeeld maken/update van een boomdiagram (voor 2 objecten)

De link die tussen een vorige en een huidige observatie wordt gemaakt, wordt voorgesteld in Figuur 52 met een tak, maar is feitelijk een getal. Het is de kans dat de huidige observatie hetzelfde object is als die van de vorige observatie. Deze kans bekomt men door de verschillende associatiecriteria. Uit deze informatie zal dan een MAP-algoritme worden opgelost om zo een globaal maximum te verkrijgen. Het globaal maximum slaat neer op welke combinatie van associaties van objecten aan trajecten of als nieuwe objecten de grootste totale som heeft. Zoals eerder vermeld zal dit algoritme zeer veel data aanmaken en bijhouden, daarom is het belangrijk dat de informatie/takken gesnoeid worden. Hiervoor ondergaat het algoritme twee stappen. Als eerste het verwijderen van bomen, als een track voor X aantal observaties als dummy wordt gelinkt, zal deze verwijderd worden en dus gezien worden als verdwenen uit het beeld. Vervolgens wordt er aan de bomen gesnoeid, er zal in elk van de boomdiagrammen N stappen teruggekeken worden en de takken die niet ter sprake zijn gekomen, zullen in het MAP-algoritme verwijderd worden. De grootte van N bepaalt de

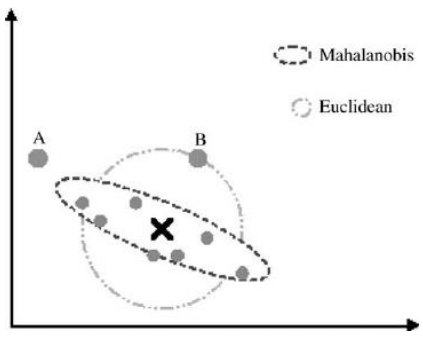
nauwkeurigheid maar ook de traagheid van het systeem. Een extra regel voor takken te verwijderen kan toegepast worden om het algoritme sneller te maken, dit door een maximum op het aantal takken per cyclus te implementeren. Hierbij worden x aantal takken met de hoogste score behouden en de rest verwijderd [47].

Om de score of de grootte van de kans te berekenen, maakt het MHT-algoritme gebruik van één of meerdere associatiecriteria, bijvoorbeeld de score kan afhankelijk zijn van de beweging en het uiterlijk van het object [47].

2.3.1.2 Nearest Neighbour algorithm (NN)

Kort uitgelegd zoekt het NN naar de dichtstbijzijnde observatie. Bij single tracking is het logisch dat dit dan ook het object is. Maar bij Multi-target tracking kunnen twee objecten elkaar zeer dicht benaderen zodat er verschillende punten zeer dichtbij liggen. Het is incorrect om te stellen dat de dichtstbijzijnde altijd het correcte object is. Daarom kan de Nearest Neighbour methode niet optimaal werken in het geval dat objecten elkaar overlappen of kruisen.

Voor de NN-techniek is het mogelijk om onderscheid te maken tussen twee types, nog specifieker onderscheid tussen het gebruik van twee types afstanden. Zo zijn er twee verschillende methodes voor de afstand te bepalen, of eigenlijk 2 soorten afstand. Mahalanobis afstand en de euclidische afstand, het verschil hiertussen is de correlatie tussen de vrijheidsgraden. De euclidische afstand is de afstand waar normaal mee gerekend wordt en dus zonder correlatie. De Mahalanobis afstand heeft een correlatie tussen de variabelen, dit kan ook gezien worden als een factor. Bijvoorbeeld als een fietser een snelheid in de x-richting heeft en geen snelheid in de y-richting, dan zal de afstand in de x-richting tussen twee frames veel groter zijn dan die van de y-richting. Als we dus de kans moeten schatten waar de fietser één frame later is, kan er mogelijk een grotere afwijking in de x-richting plaatsvinden dan in de y-richting. Dit wil zeggen dat de afwijking in de y-richting een veel zwaardere factor heeft omdat dit veel minder verwacht wordt. Dit geeft als resultaat dat de verwachte zone dat de fietser zich bevindt er eerder ovaal zal uit zien in tegenstelling tot euclidische afstand die een cirkel zal beschrijven [48]. Op Figuur 53 is het verschil duidelijk zichtbaar.



Figuur 53: Euclidean afstand en Mahalanobis afstand [49]

Dit algoritme is ook toepasbaar op Multi-object tracking. Hiervoor wordt het Global Nearest Neighbour (GNN) gebruikt. Deze zoekt naar een algemene oplossing aan de hand van een MAP-techniek (zie hieronder). Deze houdt dus met alle objecten rekening voor het toewijzen van een traject in plaats van voor elke waarneming apart [2].

2.3.2 Taakverdelingsprobleem-algoritmes

Taakverdelingsprobleem-algoritmes (*Assignment Problem*) werden origineel bedacht om taken in een bedrijf zo goed mogelijk te verdelen onder het personeel. In het geval van data-associatie betekent dit om de objecten doorheen frames zo goed mogelijk te matchen.

2.3.2.1 GRASP-methode

De GRASP-methode (*Greedy Randomized Adaptive Search Procedure*) is ontworpen voor Multi-target tracking. Het idee erachter is om willekeurig voor elke "taak" (*target*) te zoeken naar kandidaten, kandidaten zijn objecten die mogelijk aan het traject toegevoegd kunnen worden. Aan de hand van deze kandidaten kan deze methode een oplossing voorstellen. Door middel van een willekeurige zoektocht kan men de oplossing aanpassen, deze is daarom niet de beste oplossing. Dit wordt gedaan voor een bepaalde toegelaten tijd zodat het systeem real-time kan verlopen, of als de methode een bepaald aantal iteraties heeft doorlopen. Vervolgens geeft het algoritme de oplossing door, maar deze is daarom niet de optimale oplossing [50]. De controle op de snelheid is een praktische eigenschap die goed samengaat met het MHT-algoritme (zie hierboven), aangezien de boomdiagrammen van MHT veel mogelijkheden bieden en dus ook snel veel tijd in beslag neemt om op te lossen. De combinatie van de GRASP-methode op het MHT-algoritme staat in [51] uitgelegd.

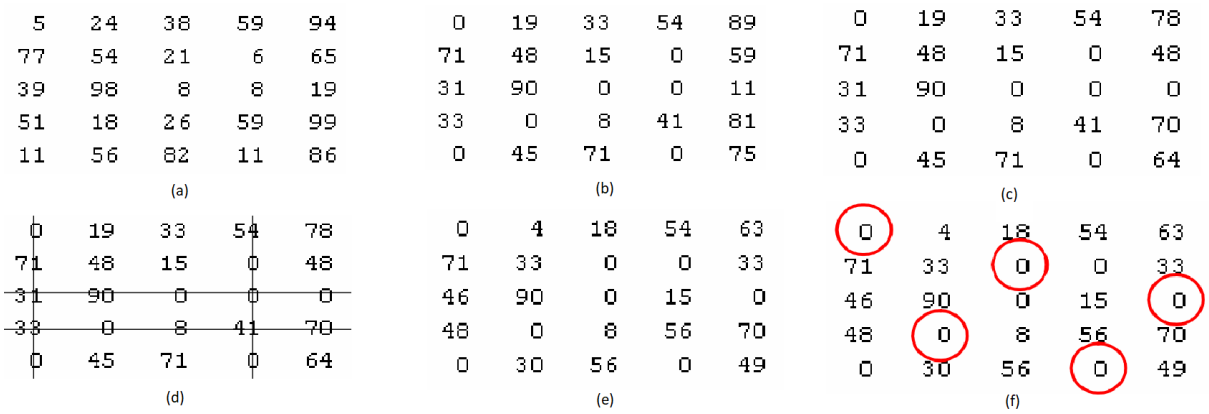
2.3.2.2 Viterbi algoritme (VA) / Viterbi Data-associatie (VDA)

Het Viterbi algoritme wordt gebuikt om het HMM (Hidden Markov model) uit te rekenen en zal zoeken naar het meest waarschijnlijke afgelegde pad. Aangezien dit met het Hidden Markov model werkt, moeten de kansen tussen de momentele toestand en die van de observatie (transitiematrix) gekend of berekenbaar zijn. Voor deze kansen zijn de waardes van de associatiecriteria bruikbaar, aangezien zij bij tracking de kans geven voor de verplaatsing van één toestand naar een andere. Nu zal het Viterbi algoritme alle kansen voor mogelijke paden berekenen, en zal de waarde met de grootste kans het meest waarschijnlijke pad zijn [52]. Dit algoritme zal dus zowel de data-associatie als ook de taakverdeling uitvoeren. Maar er moeten nog verschillende aanpassingen worden gedaan om dit algoritme van single object naar multi object tracking te brengen. Voor meer informatie over deze aanpassingen naar MTT (Multi target tracking) zie volgende paper [53], hierin is het Viterbi algoritme gebruikt om meerdere voetgangers te tracken.

2.3.2.3 Hongaars algoritme

Het Hongaars algoritme is origineel bedoeld voor het verdelen van personeelstaken (taken voor personeel in een bedrijf), exacter nog het zoeken van de ideale personeelsverdeling. Voor deze literatuurstudie is gekozen voor de versie die de minimale kost zoekt. De versie waarbij de maximale kast wordt berekend werkt analoog met die van de minimale. De eerste vereiste voor beide types is een kostmatrix, deze wordt opgesteld door de aantal taken als kolommen te zetten en de aantal personeelsleden als rijen te zetten. De kruising van één personeelslid met een taak (kruising rij met kolom) geeft weer voor hoe goed dit personeelslid is in die specifieke taak met behulp van een waarde. Deze waarde kan gaan van 0 tot 1 of van 0 tot 100 dit maakt niets uit voor het algoritme. Maar de redenering erachter wel, er kan op twee manieren geredeneerd worden ten eerste hoe hoger de waarde hoe beter het personeelslid, of omgekeerd. Afhankelijk van de redenering zal dan ook minimale of maximale kost gezocht worden. Voor deze uitleg is hoe lager de waarde het personeelslid beter in de taak en zal de minimale kost gezocht worden. Hiervoor zal de som van de waardes zo klein mogelijk zijn. Deze methode bestaat uit vijf wiskundig eenvoudige stappen voor een N x N matrix zoals in Figuur 54 [2] [54].

- Stap 1: trek van heel de rij de minimumwaarde van de rij af (Figuur 54 (b));
    - Check of een permutatiematrix van de nullen gemaakt kan worden;
- Stap 2: trek van heel de kolom de minimumwaarde van de kolom af (Figuur 54 (c));
    - Check of een permutatiematrix van de nullen gemaakt kan worden;
- Stap 3: doorstreep alle nullen in de matrix met zo min mogelijk lijnen (Figuur 54(d));
- Stap 4: zoek uit de overgebleven elementen het minimum en trek dit van alle niet doorstreepte af. (Figuur 54 (e));
- Stap 5: Check of een permutatiematrix van de nullen gemaakt kan worden, ga anders naar stap 3 (Figuur 54 (f)).



Figuur 54: Stappen van het Hongaars algoritme met (a) de startmatrix, (b) resultaat na stap 1, (c) resultaat na stap 2, (d) resultaat na stap 3, (e) resultaat na stap 4 en (f) resultaat na stap 5 [2]

Het algoritme geeft voor elke taak een personeelslid terug (in het geval dat er niet meer taken dan personeelsleden zijn). deze opstelling van personeelsleden voor elke taak is in totaal de beste opstelling en zou theoretisch ook het productiefste zijn voor een bedrijf. Dit betekend dat het mogelijk is dat niet de meest geschikte persoon voor een bepaalde taak die taak zal bemachtigen. In het geval voor tracking zal dit algoritme de beste verdeling voor de objecten aan de gekende trajecten geven. Voor meer uitleg over het Hongaars algoritme zie [54].

2.3.3 Associatiecriteria

Data-associatie betekent dat objecten (dit kunnen ook personen zijn) uit vorige frames gelinkt worden met die van uit het huidige frame. Deze link gebeurt op basis van één of meerdere associatiecriteria en bepaalt hierdoor de waarde van een link. Dit wordt ook de *weight* van de link genoemd. De weights (of het gewicht) van de link is de kans dat de objecten over twee opeenvolgende frames heen dezelfde is. De weights van een link kunnen afhankelijk zijn van meer dan één associatiecriteria en kunnen dan als volgt opgeteld worden.

$$S^i = w_{bew} S^i_{bew} + w_{uit} S^i_{uit} \qquad (2.40)$$

Met in dit voorbeeld S^i_{bew} de score a.d.h.v. het bewegingsmodel, S^i_{uit} a.d.h.v. hun uiterlijk (*appearance model*), w_{bew} *en* w_{uit} als gewichten voor deze scores. Met het instellen van deze gewichten kan de afhankelijkheid van totale score S^i tot S^i_{bew} en S^i_{uit} aangepast worden t.o.v. elkaar [47].

2.3.3.1 Dichtste buur (Nearest Neighbour)

Nearest Neighbour is reeds besproken als een volledig alleenstaand algoritme om aan data-associatie te doen, maar het kan ook gebruikt worden om de weights van een ander associatiealgoritme te bepalen. Dus hoe dichter de voorstellen liggen bij de positie of voorspelling van de gekende objecten hoe hoger de associatiescore is. Uiteraard zal dit geen vertrouwbare score zijn op zichzelf aangezien het mogelijk is dat objecten elkaar kruisen.

2.3.3.2 Verschijningsmodel (Appearance model)

Het Verschijningsmodel gebruikt het uiterlijk van objecten en personen als features voor data-associatie. Dit door de verschijning van deze objecten/personen te modelleren aan de hand van kleur- en/of intensiteitsdiagrammen. Het grote nadeel van dit type model is dat het afhankelijk is van veel parameters, zoals belichting van de omgeving/objecten, materiaaleigenschappen van het objectoppervlakte en cameraparameters [55]. Aan de hand van de diagrammen van het vorige frame en het huidige frame is het mogelijk om ze te vergelijken en zo kandidaten op te stellen aan de hand van een drempelwaarde. Zoals in Figuur 51 te zien is, zijn er vier mogelijke kandidaten aangezien hun diagram overeenkomt met die van de te tracken auto uit het vorige frame [2].

2.3.3.3 Bewegingsmodel (motion model)

Het bewegingsmodel registreert de snelheid en de positie van het object en aan de hand hiervan kan een voorspelling gemaakt worden voor de positie van de volgende locatie van het object. Er zijn grote gelijkenissen met NN want ook hier stelt de afstand tussen de voorspelling en de gedetecteerde objecten de waarde voor associatie voor. Hierbij kan er ook net zoals bij het NN-algoritme de Mahalanobis afstand gebruikt worden. Hierdoor is de score voor de associatie afhankelijk van de snelheid van de richting (bv: via x-as). Een mogelijk verschil dat het bewegingsmodel ook nog kan integreren, is de acceleratie. Door de integratie van de acceleratie zou de berekende positie nauwkeuriger zijn.

2.4 Extra

Om structuur te behouden in de drie eerste onderdelen van de literatuurstudie voor toekomstige geïnteresseerden, is er gekozen om specifiekere literatuur in het onderdeel "Extra" te plaatsen. Hierin wordt de functie `detectMarkers()` van de ArUco-module besproken.

2.4.1. ArUco-code

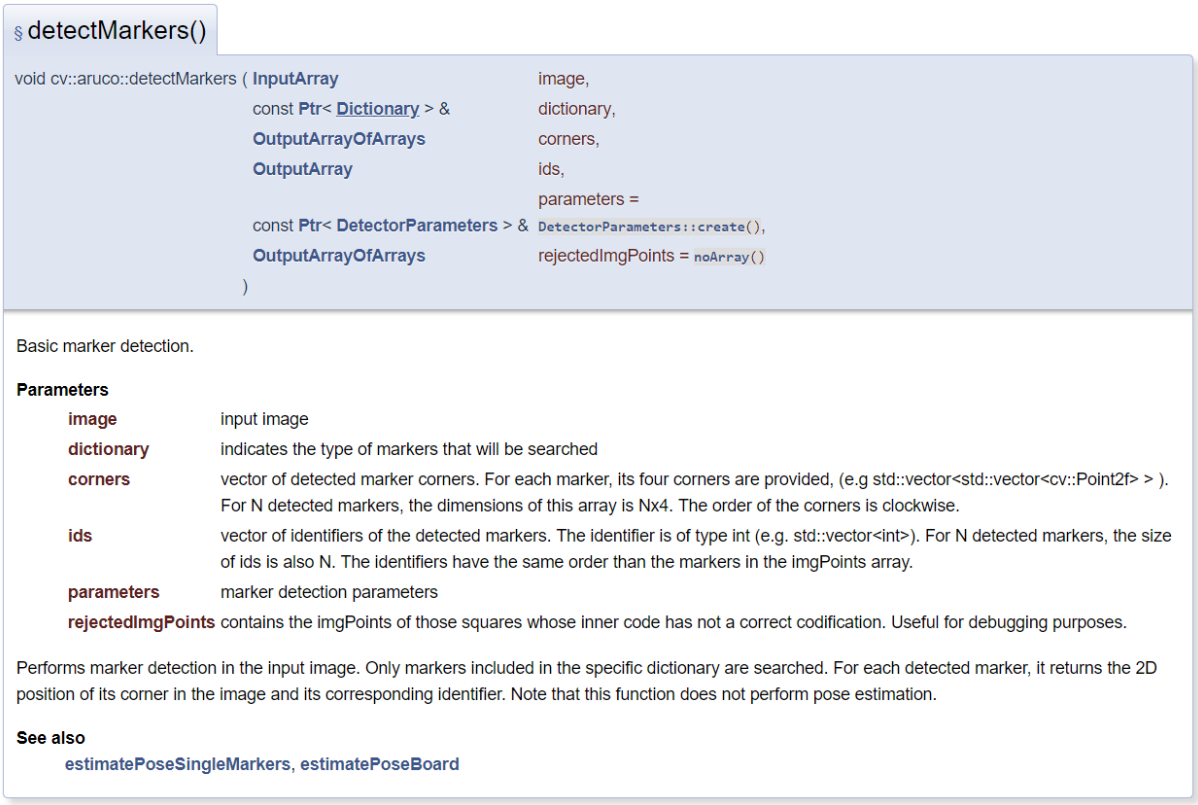
Deze masterproef focust zich op de detectie van de markers, met name de markers van de ArUco-module (OpenCV), daarom staat hieronder de uitleg van de functie om markers te detecteren.

Zoals eerder vermeld is OpenCV een open source bibliotheek met functies en klassen voor computervisie. Eén van deze modules is de ArUco-module. In 2.1.3.3 staat er meer algemene uitleg over deze module, hier is enkel de werking en zijn de stappen van de functie `detectMarkers()` toegelicht. Dit omdat deze functie een essentiële rol speelt in de volledige werking van de markerdetectie en omdat deze code noodzakelijk was om in casus twee de nodige aanpassingen te maken. De functie is in de source code van de bibliotheek terug te vinden is en op Github indien een back-up nodig is. Specifieker staat deze functie ingedeeld onder de ArUco-functies in het bestand ArUco.cpp, die te vinden valt op volgende locatie*: opencv_contrib\modules\ArUco\src\ArUco.cpp* [56].

In Figuur 55 staat de documentatie van de functie afgebeeld met de volledige uitleg over de werking en de data die de gebruiker moet doorgeven aan de functie. Kort uitgelegd zal deze functie de afbeelding en een bibliotheek als input nemen, in dit type bibliotheek staan markers in de vorm van bits opgeslagen voor al hun rotatiemogelijkheden. Op deze afbeelding wordt er gezocht naar de markers die deel uitmaken van de meegegeven bibliotheek (hieronder specifieker uitgelegd. Van de gevonden markers zijn de coördinaten van de vier hoeken in een lijst geplaatst (*"corners"*) en zijn de ID's hiervan in dezelfde volgorde in een lijst (*"ids"*) opgeslagen. Ook is het mogelijk om extra parameters mee te geven aan de functie genoemd de *"detectorParameters"*. In het geval dat dit niet gebeurt, maakt de functie zelf deze parameters automatisch aan. De parameters geven extra mogelijkheden om onder andere de positie te berekenen door bijvoorbeeld: enkel de hoeken, met behulp van de randen en nog veel andere opties die niet verder besproken worden. Het laatste wat deze functie teruggeeft zijn de hoekpunten van de afgewezen kandidaten in een lijst (*"rejecteIimgPoints"*). Deze kandidaten werden niet gezien als markers aangezien ze niet voldoen aan de specificaties die in de bibliotheken gedefinieerd staan [56].

Een nadelige eigenschap van deze functie/module is dat het voor elke ID maar één marker teruggeeft. Deze eigenschap is een vorm van data-associatie, aangezien elke marker hierdoor

uniek is en vervolgens niet dubbel kan voorkomen. Dit is met gevolg ook een probleem voor toepassingen waar identieke markers aanwezig zijn, dit is aangekaart in casus twee.



§ detectMarkers()

void cv::aruco::detectMarkers (InputArray image,
const Ptr< Dictionary > & dictionary,
OutputArrayOfArrays corners,
OutputArray ids,
parameters =
const Ptr< DetectorParameters > & DetectorParameters::create(),
OutputArrayOfArrays rejectedImgPoints = noArray()
)

Basic marker detection.

Parameters

image	input image
dictionary	indicates the type of markers that will be searched
corners	vector of detected marker corners. For each marker, its four corners are provided, (e.g std::vector<std::vector<cv::Point2f> >). For N detected markers, the dimensions of this array is Nx4. The order of the corners is clockwise.
ids	vector of identifiers of the detected markers. The identifier is of type int (e.g. std::vector<int>). For N detected markers, the size of ids is also N. The identifiers have the same order than the markers in the imgPoints array.
parameters	marker detection parameters
rejectedImgPoints	contains the imgPoints of those squares whose inner code has not a correct codification. Useful for debugging purposes.

Performs marker detection in the input image. Only markers included in the specific dictionary are searched. For each detected marker, it returns the 2D position of its corner in the image and its corresponding identifier. Note that this function does not perform pose estimation.

See also

estimatePoseSingleMarkers, estimatePoseBoard

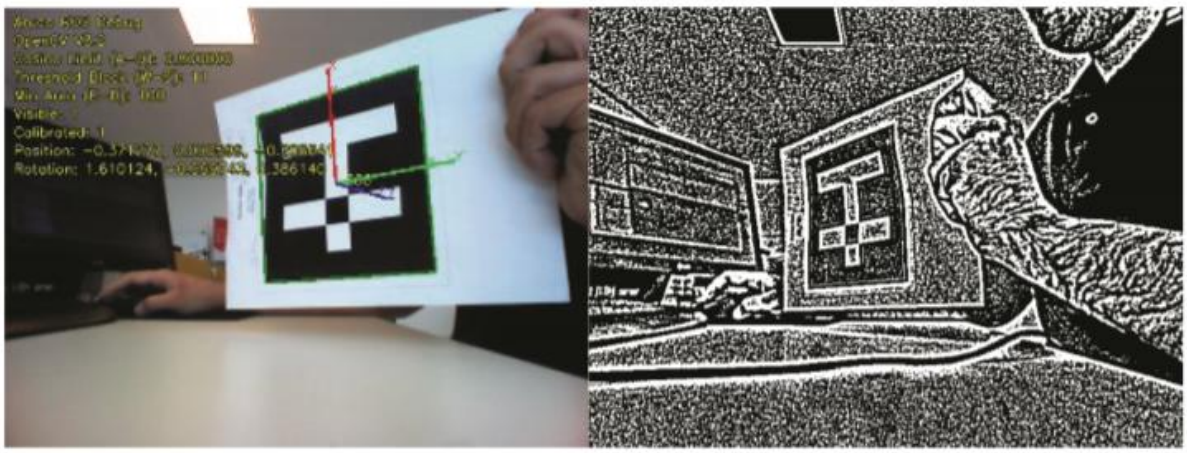
Figuur 55: Documentatie functie detectMarkers() [56]

Zoals in Figuur b. 1 uit de bijlage te zien is, zijn er vier stappen in de functie `detectMarkers()`. Ten eerste gaat men op zoek naar kandidaten, dit door de `detectCandidates()` functie. Om te beginnen vormt deze functie de afbeelding om naar grijswaardes, voor grijswaardes wordt elke pixel omgezet naar één waarde tussen 0 en 255 met 0 zwart en 255 wit. Dan zoekt men door middel van verschillende groottes van zoekvensters en de *adaptive thresholding* techniek (Figuur 56) in de afbeelding naar markerachtige vormen. Vervolgens berekent het algoritme de hoeken hiervan en worden de kandidaten die te dicht bijeen liggen gefilterd door de kleinste vorm van de twee kandidaten te elimineren. De overgebleven kandidaten zijn dan vervolgens degenen die verdergaan naar de tweede stap [56].

In de tweede stap test de functie `identifyCandidates()` aan de hand van de grijstinten van de afbeelding of de kandidaten een marker zijn uit de gekozen bibliotheek. Dit door de bits uit de afbeelding te extraheren en te vergelijken met die van de bibliotheek voor alle mogelijke rotaties. In het geval dat deze overeenkomt met die uit de bibliotheek, is deze kandidaat opgeslagen in de lijst *"validCandidates"* en zijn gevonden ID in de lijst *"idsTmp"*. In het andere geval is de kandidaat in de lijst *"rejected"* opgeslagen, wat betekent dat ze een afgewezen kandidaat zijn [56].

In stap drie filtert de `filterDetectedMarkers()` functie de markers die dubbel voorkomen. Dit betekent dat er één wegvalt als er twee dezelfde markers gebruikt worden, deze komt dus ook niet bij "rejected" in de lijst.

Stap vier dient om de vier buitenste hoeken te verfijnen, deze stap is optioneel met twee keuzes. De eerste is verfijnen aan de hand van subpixels en de tweede keuze is om het aan de hand van de contour te doen. Deze opties zijn in deze paper niet besproken maar zijn mogelijke verbeteringen op de kwaliteit van de detectie.



Figuur 56: Adaptive threshold toegepast op een afbeelding [57, p. 365]

3 Markerdetectie

Zoals in de literatuurstudie al is vermeld is het detecteren van objecten een computationeel intensieve opdracht. Daarom is er gekozen om deze detectie te benaderen aan de hand van markers. Deze zullen gedetecteerd worden in plaats van echte objecten. Eerst zal 3.1 toe lichten waarom er in deze masterproef gekozen is voor OpenCV en hierop aansluitend zal 3.2 een introductie geven voor de ArUco-module. Aangezien deze markerdetectiemethodes gaan gebruikt worden voor de casussen zal in 3.4 de nauwkeurigheid onderzocht worden. Maar om dit te kunnen doen moet de oorspronkelijke code enkele aanpassingen ondergaan. Deze aanpassingen worden in 3.3 besproken.

3.1 Introductie OpenCV

In deze masterproef is er gekozen voor de OpenCV bibliotheek, dit ten eerste omdat het open source is en daarom ter beschikking is voor iedereen. Ten tweede omdat het een zeer uitgebreide bibliotheek is die online goed gedocumenteerd is, hierdoor kan de gebruiker de werking en theorie achter elke functie achterhalen door middel van ze op te zoeken. Ten derde zijn er in deze bibliotheek al voorbeelden geïntegreerd die het voor de gebruiker zeer gemakkelijk maken en ervoor zorgen dat de gebruiker snel van start kan gaan met zijn project. Daarnaast heeft de website van OpenCV een eigen forum waarop zeer veel voorkomende problemen worden besproken en opgelost. Ook zijn er andere forums beschikbaar waar dat OpenCV uitvoerig aan bod komt. Dit omdat OpenCV een groot aantal gebruikers heeft die vaak gelijkaardige problemen hebben opgelost. Hierdoor is dit een zeer sterke troef voor iemand die pas begint met het gebruiken van deze bibliotheek. Ten slotte is OpenCV zowel compatibel voor Python als C++ en kan het op verschillende besturingssystemen worden geïnstalleerd.

3.2 Introductie ArUco-module

Deze masterproef focust zich op het ontwikkelen voor een oplossing voor de drie casussen die allemaal nood hebben aan detectie. Detectie is een zeer geheugen en processor intensieve handeling die met voorkeur uitgevoerd wordt met grafische kaarten. Omdat deze masterproef zich meer toespitst op het onderzoek dan op de werkelijke uitwerking van de casussen zijn de financiële middelen beperkt. Daarom is er gekozen voor een benadering van een oplossing, hierbij zal de detectie gedaan worden door middel van een snel en accuraat programma. Daarom is er gekozen voor de ArUco-module (deze maakt deel uit van de contrib, dit behoort niet tot het basispakket van OpenCV), bij deze module wordt detectie uitgevoerd aan de hand van markers in plaats van fysische features van de te detecteren objecten. Het grote voordeel aan deze markers en hun features is dat ze gekend zijn in het systeem en hierdoor snel gevonden kunnen worden. Zoals hierboven is vermeld ondersteund OpenCV zijn gebruikers door middel van voorbeeldcodes. Dit is ook het geval voor de ArUco-module, hiervoor zijn er

volledig functionerende programma's (ter beschikking op GitHub) die de markers detecteren en hun positie schatten.

3.3 Aanpassingen aan ArUco-module tracking

De programma's die door OpenCV ter beschikking zijn gesteld, zijn een zeer goede basis voor het detecteren van de markers. Een groot voordeel van deze programma's is dat ze zo zijn ontworpen om real-time te gebruiken. Maar om deze programma's te gebruiken voor een volledige tracker of om data van deze programma's te gebruiken zijn er aanpassingen nodig.

Als eerste moet alle data bijgehouden kunnen worden. Daarom wordt de code van de kalibratie zodanig aangepast dat zowel de gegevens uit de kalibratieafbeeldingen als de kalibratieafbeeldingen zelf opgeslagen worden. Dit wordt mogelijk gemaakt door het aantal van de opgeslagen figuren op te vragen en dit nummer in de naam te verwerken en vervolgens op te slaan met de functie `imwrite()`.

De tweede aanpassing is om de geschatte positie en pose uit te schrijven naar een CSV-bestand voor verdere verwerking. Hiervoor moet deze file aangemaakt worden en vervolgens worden alle gegevens hierin geschreven, dit alles wordt uitgevoerd met behulp van een extra bibliotheek fsteam van Visual Studio.

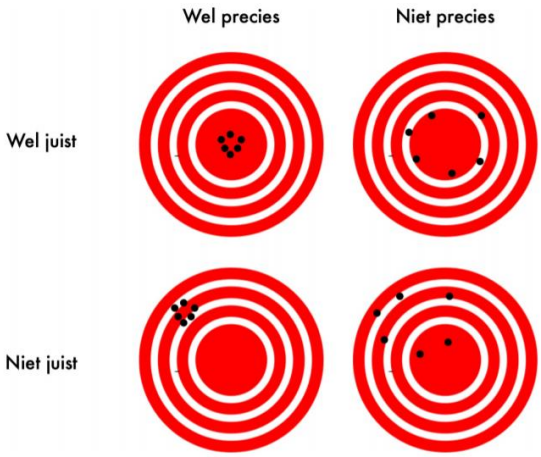
De derde aanpassing is het bekomen van de positie en rotatiegegevens. Voor de translatie parameters is dit geen probleem aangezien deze bij de functie `estimatePose()` van het origineel programma worden berekend. Bij deze functie worden de rotatievectoren ook berekend, maar deze zijn voor de verdere verwerking niet evident. Daarom worden ze omgezet in hoeken (°). Dit wordt in 2 stappen bereikt, eerst worden de rotatievectoren omgezet in een rotatiematrix met behulp van de functie `Rodrigues()` en vervolgens wordt de rotatiematrix omgevormd naar Euler hoeken met behulp van de functie `rotationMatrixToEulerAngles()`. Beide functies zijn functies van OpenCV en dus gemakkelijk te implementeren. In de laatste stap worden de hoeken in graden omgezet.

Als vierde aanpassing werd de tijd in het programma geïntegreerd zodat, met behulp van de tijd en de posities, de snelheid achterhaald kan worden. Dit wordt gedaan met behulp van ticks, dit zijn de aantal CPU ticks die sinds het opstarten van het systeem zijn gebeurd en worden opgeroepen door de functie `GetTickCount()`. Door deze waarde te delen door de frequentie van de CPU vindt men de tijd die gepasseerd is sinds het begin van het programma. De ticks (de tijd) worden telkens opgenomen wanneer een afbeelding wordt gemaakt (*retrieven* van een frame). Omdat de duur van het nemen van een frame kan variëren wordt het gemiddelde genomen van de ticks voor en na het nemen van een frame om accurater te zijn. Belangrijk om te weten is dat de ticks maar op 10 tot 16 milliseconde nauwkeurig is aangezien het elke 10-16 ms geüpdatet wordt [58].

Ten slotte werd het origineel programma omgevormd zodat het de mogelijkheid had om afbeeldingen te verwerken. Dit omdat de originele programma's enkel voor video en real-time (camera) beschikbaar zijn. Dit werd uitgevoerd door middel van de *while loop* te vervangen door een *for lus*.

3.4 Nauwkeurigheidsstudie ArUco-module

Om de nauwkeurigheid van de ArUco-module in beeld te brengen zullen de 4 types markers (ArUco-markers, ArUco-paneel, ChArUco-paneel en Diamantmarker) getest worden bij een constant beeld en een lineaire beweging. Voor de nauwkeurigheidsmetingen wordt er een onderscheid gemaakt tussen juistheid en precisie. Het onderscheid tussen de twee is in Figuur 57 afgebeeld. Voor de juistheid wordt word er gekeken in welke mate de gemiddelde overeenkomt met de werkelijke waarde [59], voor Figuur 57 is dit de afwijking van het gemiddelde van de zwarte stippen ten opzichte van het midden van de rode cirkels. De precisie is de waarde voor de mate van de spreiding van de resultaten [59], voor Figuur 57 is dit de grootte van de afwijking van de zwarte punten ten opzichte van hun gemiddelde.



Figuur 57: Verschil tussen juistheid en precisie [60, p. 7]

Vooraleer de metingen bij een constant beeld (hoofdstuk 3.4.3) en een lineaire beweging (hoofdstuk 3.4.4) gebeuren wordt de camera gekalibreerd (hoofdstuk 3.4.1) en getest op cameravervorming (hoofdstuk 3.4.2). Deze vervormingen (*camera distortion*) vinden zich voornamelijk plaats in de randen en hoeken van het beeld. Om deze te testen zal er een ArUco-paneel worden gebruikt die het volledige camerabeeld inneemt. Dan worden de afstanden tussen de markers onderling berekend. Als de afstanden in het midden ten opzichte van rand verschillen is er sprake van cameravervorming.

Voor de drie testen (cameravervormingen, constant beeld, lineaire beweging) en de camerakalibratie wordt er gebruik gemaakt van de opstelling in Figuur 58. Met bovenaan een uEye Ui 1225LE-C camera met een Fujinion 1:1.2/6mm lens op gemonteerd. De gegevens van zowel de uEye camera als de gegevens van de Fujinion lens staan respectievelijk gedocumenteerd in Figuur b. 3 en Figuur b. 2 in de bijlage afgebeeld.

Figuur 58: Opstelling voor het nauwkeurigheidsonderzoek

3.4.1 Camerakalibratie

Camerakalibratie is noodzakelijk om de interne camera parameters zoals de afstand van het brandpunt, grootte van de cellen en lens distorsies te bepalen. Door de mogelijke afwijkingen op deze parameters te berekenen kan de camera hierop gecorrigeerd worden. Deze camera kalibratie gegevens worden dan vervolgens meegegeven aan de programma's voor markerdetectie zodat er met deze afwijkingen rekening kan gehouden worden. Dit betekent dus dat de kwaliteit van de camera kalibratie een grote rol speelt in de nauwkeurigheid van de ArUco-module.

3.4.1.1 Methodiek

Vooraleer er metingen worden gemaakt zal de camera worden afgesteld en zal er een camera kalibratie van deze toestand worden genomen. Voor deze nauwkeurigheidsstudie is er gekozen om deze kalibratie te doen aan de hand van het ChArUco-paneel en niet het ArUco-paneel. Dit omdat de hoekpunten van een ChArUco-paneel betere features zijn dan die van een ArUco-paneel. Hetzelfde geld voor de kwaliteit van de camera. Een camera van hogere

kwaliteit zal een betere nauwkeurigheid hebben dan een standaard camera omdat de features beter zichtbaar zullen zijn vanwege de hogere resolutie.

Bij camerakalibratie zijn er enkele factoren waaraan gedacht moet worden, zo beschrijft de paper van HALCON dat er aan enkele regels voldaan moet worden om een zo accuraat mogelijk resultaat te bekomen [61].

- Gebruik een propere kalibratieplaat.
- Vul het volledige camerabeeld ("field of view") op met verschillende foto's, zorg dus dat het volledige camerabeeld op z'n minst één keer gebruikt is.
- Varieer de oriëntatie van de kalibratieplaat, zowel rotatie rond de X als Y-as.
- Neem minstens 10 tot 15 foto's.
- Zorg voor verlichting zodat de achtergrond donkerder is dan de kalibratieplaat.
- De lichtstukken van de kalibratieplaat moeten minstens een grijswaarde van 100 hebben.
- Zorg voor een belichting waarbij de kalibratieplaat homogeen is.
- De beelden mogen niet overbelicht zijn.
- De beelden moeten zo weinig mogelijk ruis bevatten.
- De kalibratieplaat moet volledig in beeld zijn.

De eerst negen regels gelden voor alle drie verschillende types kalibratie, maar enkel de tiende is noodzakelijk voor kalibratie aan de hand van het schaakbordpatroon [61]. Het ArUco en ChArUco-paneel mogen deels uit het beeld liggen, zo lang er maar meer dan de helft in het beeld blijft.

3.4.1.2 Resultaten en discussie

De bovenstaande stappen zijn uitgevoerd voor drie verschillende types van kalibratie: schaakbordpatroon kalibratie, ArUco-paneel kalibratie en ChArUco-paneel kalibratie. Elk van deze kalibratietechnieken gaf op dezelfde manier gegevens terug, alleen verschilden deze waardes onderling voor dezelfde camera. Om te beslissen welk type kalibratie het beste is, werd er naar de gemiddelde afkeuringsfout gekeken (*average reprojection error),* deze staan in Tabel 1 voor de drie methodes. Hoe kleiner de afkeuringsfout hoe beter de kalibratieparameters zijn.

Tabel 1: De gemiddelde afkeuringsfout voor de drie types kalibratie (in meter)

	Gem. afkeuring fout
Schaakbordpatroon	3.59e+02
ArUco-paneel	1.37e+00
ChArUco-paneel	1.33e+00

Uit Tabel 1 blijkt dat het schaakbordpatroon een zeer hoge waarde heeft in vergelijking met het ChArUco en ArUco-paneel, dit is hoogstwaarschijnlijk door een fout in de code. Aangezien de afkeuringsfout van het ChArUco-paneel zich dichter bij nul begeeft, zal er hiervoor gekozen worden. Om de kalibratie te optimaliseren werd er kunstmatig wit licht aan de omgeving toegevoegd. Ook werden er in plaats van 25 kalibratieafbeeldingen 32 kalibratieafbeeldingen genomen voor in het geval dat enkele afbeeldingen niet goed zijn. De gegevens van de kalibratie voor de metingen zijn te zien in Figuur 59. Met als eerste het moment van de kalibratie en vervolgens de grootte van het beeldscherm. Daarna volgt de camera-matrix. Hierin worden de afwijkingen van de interne parameters opgenomen en in een 3X3 matrix gegeven. De voorlaatste gegevens zijn de lens distorsies, deze worden in een 1X5 matrix gegeven. Als laatste de gemiddelde afkeuringsfout wat een maatstaf is voor de kwaliteit van de kalibratie. Zoals eerder vermeldt, hoe kleiner dit getal hoe beter de parameters zijn geschat.

```
%YAML:1.0
---
calibration_time: "Thu Mar 14 12:10:22 2019"
image_width: 640
image_height: 480
flags: 0
camera_matrix: !!opencv-matrix
   rows: 3
   cols: 3
   dt: d
   data: [ 1.0200722895843224e+03, 0., 3.6357042053326524e+02, 0.,
       1.0184873235821842e+03, 2.4780482830792738e+02, 0., 0., 1. ]
distortion_coefficients: !!opencv-matrix
   rows: 1
   cols: 5
   dt: d
   data: [ -2.3918813947154738e-01, 4.2093927123605109e-02,
       4.9945114376929826e-04, -2.8039062704152528e-03,
       8.2776100576640022e-01 ]
avg_reprojection_error: 1.1625183631800577e+00
```

Figuur 59: Kalibratiegegevens voor ChArUco-paneel van de metingen op 14-03-19

3.4.1.3 Conclusie

Camerakalibratie wordt typisch uitgevoerd aan de hand van een horizontaal vlak met uitgelijnde elementen van een vaste maat. Het is echter bijna onmogelijk om een vlak patroon te maken en deze vervolgens zonder vervorming voor de camera te positioneren[62]. Dit is ook het geval voor dit onderzoek, het ChArUco-paneel wordt geprint op een A4-blad en vervolgens op een harde, vlakke achtergrond bevestigd. Bij het printen zal in het eerste geval het papier lichtjes vervormen door de inkt en het is het haast onmogelijk om deze te bevestigen zonder imperfecties. Ook zal de nauwkeurigheid van de printer een rol spelen. De grootte van de markers in het ChArUco-paneel horen zo nauwkeurig mogelijk op maat te zijn om de kalibratie met hoge precisie uit te voeren. Dit is niet het geval bij de gebruikte printer maar sommige markers zijn halve millimeters te groot of te klein.

3.4.2 Metingen voor cameravervorming

Lens distorsies kunnen gezien worden als de afwijking van het camerabeeld ten opzichte van een rechthoekig patroon. In de voorgaande stap, de camera kalibratie, zijn deze reeds berekend. Hierdoor hebben de programma's de mogelijkheid om dit soort fouten te corrigeren. In deze stap wordt er gemeten in welke mate de kalibratie hieraan voldoet.

3.4.2.1 Methodiek

Om de cameravervormingen te meten wordt er een ArUco-paneel gebruikt die het volledige beeld van de camera in beslag neemt. Aan de hand van dit beeld wordt de onderlinge posities van ArUco-markers berekend en vergeleken. Zo kan een vergelijking worden opgesteld tussen de onderlinge afstanden van de markers in het midden van het beeld en aan de randen.

Voor deze vergelijking is er een serie van vijf foto's gemaakt van het ArUco-paneel dat het volledige beeld inneemt. Deze beelden worden doorheen het programma gevoerd, één keer met de kalibratie gegevens van de camerakalibratie en éénmaal zonder (Figuur 60). Als resultaat levert het programma een CSV-file met de posities en poses van de markers. De CSV-file kan vervolgens in Excel worden geïmporteerd en mits enkele aanpassingen zal hiervan de onderlinge afstand tussen de markers berekend kunnen worden met formule (3.1).

$$Afstand = \sqrt{(X_{M0} - X_{M1})^2 + (Y_{M0} - Y_{M1})^2 + (Z_{M0} - Z_{M1})^2} \qquad (3.1)$$

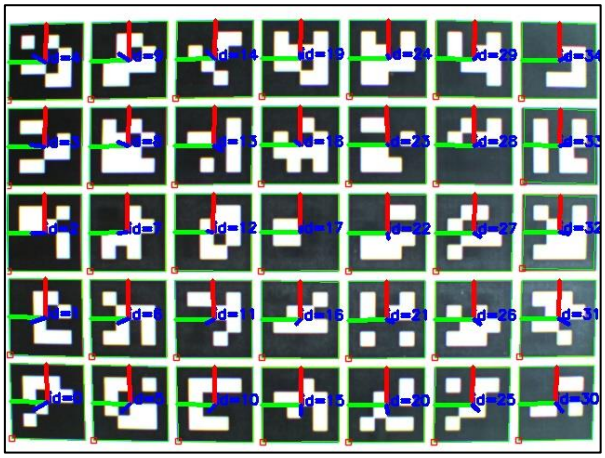
Met X_{M0} , Y_{M0} en Z_{M0} zijn respectievelijk de X, Y en Z coördinaten van marker 0 en X_{M1} , Y_{M1} en Z_{M1} zijn respectievelijk de X, Y en Z coördinaten van marker 1.

```
%YAML:1.0
---
calibration_time: "Thu Mar  7 09:08:10 2019"
image_width: 640
image_height: 480
flags: 0
camera_matrix: !!opencv-matrix
   rows: 3
   cols: 3
   dt: d
   data: [ 1.0200722895843224e+03, 0., 3.6357042053326524e+02, 0.,
       1.0184873235821842e+03, 2.4780482830792738e+02, 0., 0., 1. ]
distortion_coefficients: !!opencv-matrix
   rows: 1
   cols: 5
   dt: d
   data: [ 0,0,0,0,0]
```

Figuur 60: Camerakalibratie met lens distorsies naar nul gezet

3.4.2.2 Resultaten en discussie

Uit de afbeeldingen kan er met het blote oog gezien worden dat er cameravervorming aanwezig is, dit kan gezien worden in Figuur 61. Deze laat de verwerking van de data zien, voor elke marker worden er dus X, Y en Z coördinaten berekend.



Figuur 61: Verwerking van de metingen voor cameravervorming

Aan de hand van de genomen afbeelding van het ArUco-paneel zou men verwachten dat het programma de afstand tussen de markers aan de randen van het ArUco-paneel kleiner schat. Dit omdat het midden van de randen in Figuur 59 bolvormig te staan. Ook lijken de afstanden in het midden (rond marker 17) groter dan aan de randen.

De belangrijkste factor bij het verwerken van deze gegevens is de positie van de marker ten opzichte van het midden. Daarom worden de afstanden tussen de grenzende markers opgedeeld in vier zones, wat in Figuur 62 afgebeeld staat. In Tabel 2 en Tabel 3 staan de gemiddelde afstanden van de aangrenzende markers, respectievelijk zonder en met compensatie van lens distorsies. Ook wordt hier de som van elke rij en kolom uitgerekend om zo de eerste verwachtingen te controleren.

34	zone 4	**33**	zone 4	**32**	zone 4	**31**	zone 4	**30**
zone 4		zone 4		zone 3		zone 4		zone 4
29	zone 4	**28**	zone 3	**27**	zone 3	**26**	zone 4	**25**
zone 4		zone 3		zone 2		zone 3		zone 4
24	zone 3	**23**	zone 2	**22**	zone 2	**21**	zone 3	**20**
zone 4		zone 2		zone1		zone 2		zone 4
19	zone 3	**18**	zone1	**17**	zone 1	**16**	zone 3	**15**
zone 4		zone 2		zone 1		zone 2		zone 4
14	zone 3	**13**	zone 2	**12**	zone 2	**11**	zone 3	**10**
zone 4		zone 3		zone 2		zone 3		zone 4
9	zone 4	**8**	zone 3	**7**	zone 3	**6**	zone 4	**5**
zone 4		zone 4		zone 3		zone 4		zone 4
4	zone 4	**3**	zone 4	**2**	zone 4	**1**	zone 4	**0**

Figuur 62: Verdeling van de afstanden in vier zones voor een ArUco-paneel

Tabel 2: Gemiddeldes afstanden per kolom en rij zonder lens distorsies compensatie (in meter)

Gemiddelde onderlinge afstand									
34	3,47E-02	**33**	3,38E-02	**32**	3,39E-02	**31**	3,40E-02	**30**	**som=** 1,36E-01
3,45E-02		3,42E-02		3,36E-02		3,41E-02		3,37E-02	
29	3,51E-02	**28**	3,38E-02	**27**	3,39E-02	**26**	3,49E-02	**25**	**som=** 1,38E-01
3,41E-02		3,37E-02		3,42E-02		3,42E-02		3,38E-02	
24	3,36E-02	**23**	3,46E-02	**22**	3,37E-02	**21**	3,57E-02	**20**	**som=** 1,38E-01
3,36E-02		3,42E-02		3,38E-02		3,33E-02		3,42E-02	
19	3,41E-02	**18**	3,39E-02	**17**	3,36E-02	**16**	3,45E-02	**15**	**som=** 1,36E-01
3,36E-02		3,33E-02		3,36E-02		3,39E-02		3,36E-02	
14	3,40E-02	**13**	3,34E-02	**12**	3,38E-02	**11**	3,40E-02	**10**	**som=** 1,35E-01
3,29E-02		3,28E-02		3,35E-02		3,26E-02		3,35E-02	
9	3,37E-02	**8**	3,36E-02	**7**	3,36E-02	**6**	3,45E-02	**5**	**som=** 1,35E-01
3,47E-02		3,35E-02		3,37E-02		3,27E-02		3,50E-02	
4	3,45E-02	**3**	3,36E-02	**2**	3,37E-02	**1**	3,63E-02	**0**	**som=** 1,38E-01
som=		**som=**		**som=**		**som=**		**som=**	
2,03E-01		2,02E-01		2,03E-01		2,01E-01		2,04E-01	

Tabel 3: Gemiddeldes afstanden per kolom en rij met lens distorsies compensatie (in meter)

Gemiddelde onderlinge afstand									
34	3,44E-02	**33**	3,38E-02	**32**	3,40E-02	**31**	3,38E-02	**30**	**som=** 1,36E-01
3,43E-02		3,39E-02		3,33E-02		3,40E-02		3,36E-02	
29	3,48E-02	**28**	3,39E-02	**27**	3,39E-02	**26**	3,46E-02	**25**	**som=** 1,37E-01
3,41E-02		3,40E-02		3,41E-02		3,41E-02		3,38E-02	
24	3,38E-02	**23**	3,44E-02	**22**	3,37E-02	**21**	3,53E-02	**20**	**som=** 1,37E-01
3,36E-02		3,42E-02		3,38E-02		3,33E-02		3,41E-02	
19	3,40E-02	**18**	3,39E-02	**17**	3,36E-02	**16**	3,44E-02	**15**	**som=** 1,36E-01
3,37E-02		3,34E-02		3,36E-02		3,38E-02		3,36E-02	
14	3,39E-02	**13**	3,34E-02	**12**	3,37E-02	**11**	3,40E-02	**10**	**som=** 1,35E-01
3,30E-02		3,29E-02		3,34E-02		3,25E-02		3,34E-02	
9	3,37E-02	**8**	3,37E-02	**7**	3,35E-02	**6**	3,42E-02	**5**	**som=** 1,35E-01
3,40E-02		3,34E-02		3,36E-02		3,30E-02		3,45E-02	
4	3,40E-02	**3**	3,37E-02	**2**	3,37E-02	**1**	3,57E-02	**0**	**som=** 1,37E-01
som=		**som=**		**som=**		**som=**		**som=**	
2,03E-01		2,02E-01		2,02E-01		2,01E-01		2,03E-01	

Tabel 2 en Tabel 3 spreken de verwachtingen tegen, de som van de gemiddelde afstanden van de middelste rijen en kolommen zijn zelfs kleiner dan de andere rijen en kolommen. Dit is tegengesteld van wat de afbeelding van de verwerking (Figuur 61) laat blijken. Om de tweede verwachting te toetsen worden voor de vier zones het gemiddelde berekend. Tabel 4 weergeeft de metingen zowel met compensatie van de lensdistorsies als zonder. Ook hier spreken de metingen de verwachtingen tegen, zo zullen de afstanden naar het midden van het scherm toe kleiner zijn dan aan de randen van het beeld. Ook dit zorgt ervoor dat de afbeelding van de verwerking (Figuur 61) niet overeen komt met de metingen, want hierop zijn net door de distorsies de buitenste markers optisch kleiner.

Tabel 4: Vergelijking van met en zonder lens distorsies (in meter)

Lens distorsies	**Met**	**Zonder**
zone 1	0,033724	0,033723
zone 2	0,033744	0,03379
zone 3	0,033804	0,033837
zone 4	0,033938	0,034083

Om te achterhalen waarom de twee verwachtingen foutief waren is gekeken naar de twee versies van gegevens voor de metingen . Hierbij is een duidelijk verschil in de Z-coördinaat van de twee gegevens te zien, zo zullen de Z-coördinaten bij de dataset zonder compensatie van de lens distorsies groter zijn dan diegenen met. Dit betekent dat het programma de markers verder van zich af ziet liggen. Wanneer het verschil van gemiddeldes tussen de Z-coördinaten van met en zonder lens distorsie compensatie in Figuur 62 wordt gegoten bekomt men Tabel 5.

Tabel 5: Verschil in Z-coördinaat voor met en zonder lens distorsie compensatie (in meter)

8,71E-03	zone 4	6,19E-03	zone 4	7,07E-03	zone 4	5,72E-03	zone 4	8,42E-03
zone 4		zone 4		zone 3		zone 4		zone 4
5,50E-03	zone 4	3,15E-03	zone 3	2,25E-03	zone 3	2,64E-03	zone 4	4,88E-03
zone 4		zone 3		zone 2		zone 3		zone 4
4,32E-03	zone 3	1,82E-03	zone 2	3,09E-03	zone 2	3,70E-03	zone 3	3,09E-03
zone 4		zone 2		zone1		zone 2		zone 4
3,70E-03	zone 3	1,63E-03	zone1	4,98E-04	zone 1	1,12E-03	zone 3	2,74E-03
zone 4		zone 2		zone 1		zone 2		zone 4
5,08E-03	zone 3	3,02E-03	zone 2	1,52E-03	zone 2	2,80E-03	zone 3	4,04E-03
zone 4		zone 3		zone 2		zone 3		zone 4
7,29E-03	zone 4	5,30E-03	zone 3	4,31E-03	zone 3	4,49E-03	zone 4	7,01E-03
zone 4		zone 4		zone 3		zone 4		zone 4
1,15E-02	zone 4	8,66E-03	zone 4	8,24E-03	zone 4	8,32E-03	zone 4	1,03E-02

In Tabel 5 is het duidelijk dat het verschil in Z-coördinaat toeneemt naarmate men van het midden naar de rand beweegt. Dit betekent dat de lens distorsies ervoor zorgen dat de markers er optisch kleiner uitzien en dat het programma de markers kleiner ziet dan ze zijn en hierdoor de markers verder weg schat. Doordat het programma de markers verder weg schat, zal de afstand tussen de markers ook groter zijn dan ze werkelijk zijn. Ook in voorgaande berekeningen wordt dit bevestigd, zo zullen de afstanden in Tabel 4 zonder compensatie groter zijn dan met compensatie van lens distorsies. Dit kan ook gezien worden als Tabel 2 en Tabel 3 met elkaar vergeleken worden.

3.4.2.3 Conclusie

De camerakalibratie zorgt ervoor dat de lensdistorsies gecompenseerd kunnen worden. Uit metingen blijkt echter dat deze nog niet 100% gecompenseerd worden. Hierdoor zullen markers die zich dichter bij de randen van het scherm bevinden kleiner lijken dan ze werkelijk zijn. Daardoor worden ze dan verder weg geschat en zal er een afwijking ontstaan op de positie van de markers. De afwijking van de onderlinge afstand zal gemiddeld onder de 1 mm zijn met de hoogst gemeten waarde aan de rand van 1,1 mm. Deze afwijking wijst op een afwijking op de nauwkeurigheid van markerspositie en daarom is het dus essentieel om een zo goed mogelijke camerakalibratie uit te voeren. Zo kan dit type afwijkingen zo goed mogelijk geëlimineerd worden.

3.4.3 Metingen voor een constant beeld

In dit onderdeel zijn de vier types markers van de ArUco-module getest op een constante fout door middel van de markers te filmen terwijl ze niet gemanipuleerd worden. Hierdoor is het mogelijk om na te gaan hoe nauwkeurig een meting werkelijk is, zelfs als er geen beweging plaatsvindt.

3.4.3.1 Methodiek

Voor dit experiment zal de camera enkele seconden de verschillende markers filmen terwijl deze niet gemanipuleerd worden. Aan de hand van deze metingen kan er achterhaald worden of de programma's voor elk frame hetzelfde resultaat geeft en of de ground truth overeenkomt met wat uit de metingen voortvloeit.

Voor de ArUco-markers en het ArUco-paneel wordt dezelfde data genomen, dit omdat het ArUco-paneel bestaat uit verschillende ArUco-markers. De opstelling van de markers in het ArUco-paneel is afgebeeld in Figuur 63 met de markers gerepresenteerd met hun nummer en de ground truth van het ArUco-paneel wordt weergegeven in Tabel 6.

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24
25	26	27	28	29
30	31	32	33	34

Figuur 63: Opstelling markers in ArUco-paneel

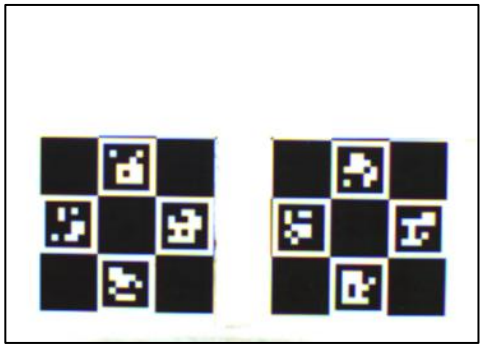
Tabel 6: Ground truth van het ArUco-paneel

Ground truth ArUco-paneel	
Type ArUco-markers	DICT_4X4_50
Grootte ArUco-paneel (X, Y)	5 op 7 (ArUco-markers)
Lengte (zijde) marker	0.0300 ± 0.001m
Lengte spatie tussen markers	0.0030 ± 0.001 m
Afstand van marker tot marker	0.0330 ± 0.001 m
Totaal lengte ArUco-paneel	0.2280 m
Totale breedte ArUco-paneel	0.1620 m

Voor de metingen van het ChArUco-paneel is er gebruik gemaakt van een paneel waarvan de gegevens in Tabel 7 staan genoteerd. Dit paneel wordt vervolgens ook gebruikt voor de metingen van de Diamantmarkers mits enkele aanpassingen, zo wordt het ChArUco-paneel deels afgedekt om zo twee Diamantmarkers te creëren die op een gekende afstand van elkaar af liggen. De gegevens van het afgedekte ChArUco-paneel staan in Tabel 6. De afstand tussen de 2 Diamantmarkers is hierdoor gelijk aan viermaal de zijde van een vierkant, dit is 0.14 meter. In Figuur 64 is te zien hoe het ChArUco-paneel eruitziet na het afplakken van overbodige markers.

Tabel 7: Ground truth ChArUco-paneel

Ground truth ChArUco-paneel	
type	DICT_5X5_50
Grootte ChArUco-paneel (X, Y)	5 op 7
Lengte vierkanten	0.0350 ± 0.001 m
Lengte (zijde) marker	0.0280 ± 0.001 m
Zijde Diamond	10,5 ± 0.1 cm
***Afstand tussen 2 Diamantmarkers**	14 ± 0.1 cm
Totaal lengte ChArUco-paneel	0.245 m
Totaal breedte ChArUco-paneel	0.175 m



Figuur 64: ChArUco-paneel deels afgedekt om twee Diamantmarkers te bekomen

Een belangrijke opmerking is dat de ground truth van het ChArUco-paneel en het ArUco-paneel met elkaar verschillen. Ten eerste gebruiken het ChArUco-paneel en het ArUco-paneel een andere bibliotheek, dit op zich heeft geen effect op de nauwkeurigheid want de nauwkeurigheid van het ChArUco-paneel is afhankelijk van de hoekpunten. Het grote verschil is dat de grootte van het ChArUco-paneel verschilt met die van het ArUco-paneel. Het resultaat hiervan is dat de verhouding van de grootte ten opzichte van de afstand tot de camera verschilt. Spijtig genoeg heeft dit een effect op de nauwkeurigheid en daarom moet er een factor toegevoegd worden om zo deze verhouding gelijk te maken. Deze factor werd spijtig genoeg niet in kaart gebracht in deze scriptie en ook niet in rekening gebracht met de gemaakte conclusies. Ook is te zien dat de markers een andere grootte hebben omdat het in het ChArUco-paneel noodzakelijk is om ruimte te laten tussen de vierkanten en de markers. Het effect van deze ruimte zal niet in dit onderzoek besproken worden.

3.4.3.2 ArUco-markers

Bij de ArUco-markers wordt er een video-opname gemaakt van het ArUco-paneel op een normale afstand, wat voor deze meting een afstand was tussen de 700 en 800 millimeter, en van tussen de 300 en 400 millimeter waarbij het ArUco-paneel het volledige scherm inneemt.

Resultaten en discussie

Theoretische gezien zouden voor elke frame de waardes van elke marker hetzelfde zijn. Dit is echter niet het geval, als er gekeken wordt naar de data is er wel sprake van constante metingen, maar er is ook sprake van uitschieters. Waardes worden als uitschieters beschouwd in het geval dat ze meer dan drie keer de standaarddeviatie afwijken van het gemiddelde. Deze uitschieters zijn duidelijk gemaakt voor de markers 34, 17 en 6 respectievelijk in Tabel b. 1, Tabel b. 2 en Tabel b. 3 in de bijlage. De reden van deze uitschieters is ongekend maar ze zijn bijna altijd aanwezig wat een negatief gevolg heeft op de nauwkeurigheid. De nauwkeurigheid van de markers 34, 17 en 6 met de uitschieters inbegrepen geeft als resultaat weergegeven in Tabel b. 4, Tabel b. 5 en Tabel b. 6 in bijlage.

Wanneer de resultaten van tussen de 300 en 400 millimeter vergeleken worden met de resultaten van op een normale afstand kan er gezien worden dat de standaarddeviatie groter is. Dit betekent dat voor de translatie coördinaten de Z-afstand, wat in dit geval de loodrechte afstand van de camera tot de marker is, een effect heeft op de nauwkeurigheid. Dus hoe verder de marker gelegen is van de camera, hoe onnauwkeuriger de positie en pose van de marker is. Hetzelfde bekomt men wanneer de resultaten van de vergelijking van ground truth met de resultaten van de metingen vergeleken worden. In Tabel 8 worden de resultaten van de berekeningen voor de afstand tussen 2 markers weergegeven voor zowel van dichtbij als van op een afstand.

Tabel 8: Resultaten van de berekeningen voor dichtbij en van op een afstand (in meters)

Horizontaal	Dichtbij	Op afstand	Verticaal	Dichtbij	Op afstand
Tussen 33 en 34	3,44E-02	3,43E-02	**Tussen 29 en 34**	3,41E-02	3,43E-02
Tussen 32 en 33	3,48E-02	3,35E-02	**Tussen 28 en 33**	3,56E-02	3,35E-02
Tussen 31 en 32	3,39E-02	3,36E-02	**Tussen 27 en 32**	3,39E-02	3,37E-02
Tussen 30 en 31	3,45E-02	3,60E-02	**Tussen 26 en 31**	3,40E-02	3,37E-02
Tussen 28 en 29	3,41E-02	3,36E-02	**Tussen 25 en 30**	3,39E-02	3,44E-02
Tussen 27 en 28	3,42E-02	3,36E-02	**Tussen 24 en 29**	3,40E-02	3,37E-02
Tussen 26 en 27	3,39E-02	3,37E-02	**Tussen 23 en 28**	3,41E-02	3,42E-02
Tussen 25 en 26	3,49E-02	3,96E-02	**Tussen 22 en 27**	3,40E-02	3,46E-02
Tussen 23 en 24	3,44E-02	3,39E-02	**Tussen 21 en 26**	3,37E-02	3,36E-02
Tussen 22 en 23	3,40E-02	3,33E-02	**Tussen 20 en 25**	3,43E-02	3,84E-02
Tussen 21 en 22	3,40E-02	3,37E-02	**Tussen 19 en 24**	3,40E-02	3,37E-02
Tussen 20 en 21	3,44E-02	3,38E-02	**Tussen 18 en 23**	3,40E-02	3,44E-02
Tussen 18 en 19	3,39E-02	3,36E-02	**Tussen 17 en 22**	3,43E-02	3,42E-02
Tussen 17 en 18	3,41E-02	3,37E-02	**Tussen 16 en 21**	3,40E-02	3,35E-02
Tussen 16 en 17	3,40E-02	3,36E-02	**Tussen 15 en 20**	3,38E-02	3,37E-02
Tussen 15 en 16	3,41E-02	3,37E-02	**Tussen 14 en 19**	3,38E-02	3,51E-02
Tussen 13 en 14	3,39E-02	3,56E-02	**Tussen 13 en 18**	3,36E-02	3,34E-02
Tussen 12 en 13	3,38E-02	3,34E-02	**Tussen 12 en 17**	3,35E-02	3,36E-02
Tussen 11 en 12	3,37E-02	3,36E-02	**Tussen 11 en 16**	3,33E-02	3,35E-02
Tussen 10 en 11	3,50E-02	3,51E-02	**Tussen 10 en 15**	3,41E-02	3,47E-02
Tussen 8 en 9	3,36E-02	3,34E-02	**Tussen 9 en 14**	3,31E-02	3,47E-02
Tussen 7 en 8	3,38E-02	3,33E-02	**Tussen 8 en 13**	3,35E-02	3,35E-02
Tussen 6 en 7	3,37E-02	3,35E-02	**Tussen 7 en 12**	3,37E-02	3,37E-02
Tussen 5 en 6	3,46E-02	3,91E-02	**Tussen 6 en 11**	3,37E-02	3,37E-02
Tussen 3 en 4	3,42E-02	3,47E-02	**Tussen 5 en 10**	3,35E-02	3,56E-02
Tussen 2 en 3	3,33E-02	3,39E-02	**Tussen 4 en 9**	3,35E-02	3,57E-02
Tussen 1 en 2	3,37E-02	3,39E-02	**Tussen 3 en 8**	3,26E-02	3,36E-02
Tussen 0 en 1	3,40E-02	3,85E-02	**Tussen 2 en 7**	3,31E-02	3,48E-02
			Tussen 1 en 6	3,33E-02	3,36E-02
			Tussen 0 en 5	3,33E-02	3,43E-02

Door de gegevens in Tabel 8 te verwerken zal men tot Tabel 9 bekomen en ook hier zal men tot hetzelfde besluit bekomen, namelijk dat de metingen van dichtbij de camera veel nauwkeuriger zijn dan die van op een grotere afstand. Volgens de ground truth is de onderlinge afstand tussen twee langs elkaar liggende markers 0.0330 m. Ook is de standaarddeviatie van dichtbij (8,33E-04) veel kleiner voor de onderlinge afstand tussen de markers dan die van op verdere afstand (1,04E-03), dit wil zeggen dat de loodrechte afstand tot de camera een rol speelt in de nauwkeurigheid van de markers.

Tabel 9: Verwerking van de onderlinge afstand tussen de markers (in meter)

	Dichtbij	**Op afstand**
Gemiddelde	3,40E-02	3,43E-02
Standaarddeviatie	8,33E-04	1,04E-03
Maximum	3,84E-02	3,84E-02
Minimum	3,26E-02	3,34E-02
Maximaal positieve afwijking	4,47E-03	4,18E-03
Maximaal negatieve afwijking	1,40E-03	8,80E-04

Conclusie

De gegevens van het onderzoek bij een constant beeld geven aan dat er sprake is van verschillende uitschieters. Er is geen voor de hand liggende, logische verklaring voor deze uitschieters, maar ze kunnen mogelijk een neveneffect zijn van extern licht in de omgeving. Dit heeft vervolgens een negatief effect op de nauwkeurigheid van de markers. Voor de nauwkeurigheid (tussen 2 markers) is er een gemiddelde afwijking van 1 millimeter (dichtbij) tot 1,3 millimeter (op afstand) waar te nemen, dit kan toegeschreven worden aan de lensdistorsies en is eerder aangehaald. Als de totale maximale afwijking voor de ArUco marker onderzocht wordt, wordt de som van de precisie en de nauwkeurigheid genomen. Voor de juistheid van de marker wordt geschat dat dit de helft is van de nauwkeurigheid tussen twee markers.

De nauwkeurigheid (afstand tussen 2 markers) heeft een afwijking van maximaal 5.4 mm. Dit betekent dat een marker tot 3 mm kan afwijken. Deze afwijking zal aan de rand van het scherm plaatsvinden, dit vanwege de lensdistorsies die in het vorige onderzoek zijn uitgelegd.

De precisie en nauwkeurigheid van de metingen voor van tussen de 300 en 400 millimeter zijn beter dan die van op een afstand, wat betekent dat de afstand van de camera tot de markers een effect heeft op de nauwkeurigheid. Dit is ook zichtbaar bij de gemiddelde afstand tussen de markers, hierbij benaderen de metingen van tussen de 300 en 400 millimeter de ground truth beter dan die van de metingen op een afstand.

3.4.3.3 ArUco-paneel

De berekeningen worden herhaald voor het ArUco-paneel aangezien er gebruik gemaakt wordt van dezelfde videofragment. Het verschil met de ArUco-markers is dat het ArUcopaneel maar één dataset geeft per frame, hierdoor zal de verwerking sneller verlopen.

Resultaten en discussie

Aangezien de metingen van het ArUco bord en de ArUco-markers hetzelfde zijn, moeten de gegevens met elkaar overeenkomen. Dit is ongeveer het geval als men kijkt naar marker 30, aangezien het assenstelsel van de marker midden in deze marker geplaatst wordt en de as van het ArUco-paneel in de hoek linksonder. Uiteraard zullen deze gegevens niet exact overeenkomen, want het ArUco-paneel wordt berekend aan de hand van alle gedetecteerde markers. Met als gevolg dat er geen sprake meer is van veel identieke metingen en verschillende uitschieters maar telkens een gemiddelde van de gedetecteerde markers. Dit gemiddelde wordt berekend door een zo goed mogelijk vlak te fitten doorheen alle punten van de gedetecteerde markers en aan de hand van de positie en meegegeven gegevens van de ground truth wordt het assenstelsel van het ArUco-paneel berekend.

Vermoedelijk zullen de verschillende uitschieters een effect hebben op de nauwkeurigheid van het ArUco-paneel maar ze worden er in dit geval deels uitgemiddeld door de goede metingen van de andere markers. In Tabel 10 staan de resultaten voor de meting van een constant beeld op een afstand tussen 700 en 800 millimeter en in

Tabel 11 voor tussen de 300 en 400 millimeter. Bij het vergelijken van deze 2 tabellen is het duidelijk zichtbaar dat ook hier de precisie van tussen de 300 en 400 millimeter beter is dan die vanop afstand.

Tabel 10: Nauwkeurigheid ArUco-paneel van een constant beeld (700-800mm)

ArUco-paneel						
	Rotatie (rad)			**Translatie (m)**		
	X	**Y**	**Z**	**X**	**Y**	**Z**
Gemiddelde	178,9338	-1,19434	180,0528	1,10E-01	-1,24E-01	7,30E-01
Standaarddeviatie	6,01E-02	7,47E-02	3,08E-03	1,23E-05	1,09E-05	1,49E-04
Maximum	179,152	-1,08034	180,062	1,10E-01	-1,24E-01	7,30E-01
Minimum	178,822	-1,4753	180,045	1,10E-01	-1,24E-01	7,29E-01
Maximaal positieve afwijking	2,18E-01	1,14E-01	9,20E-03	4,77E-05	4,27E-05	6,87E-04
Maximaal negatieve afwijking	1,12E-01	2,81E-01	7,80E-03	2,93E-05	2,13E-05	3,30E-04

Tabel 11: Nauwkeurigheid ArUco-paneel van een constant beeld (300-400mm)

ArUco-paneel						
	Rotatie (rad)			Translatie (m)		
	X	Y	Z	X	Y	Z
Gemiddelde	179,4417	1,728115	-90,2553	9,82E-02	8,03E-02	3,60E-01
Standaarddeviatie	3,23E-03	2,50E-03	3,40E-04	1,56E-06	1,04E-06	7,79E-06
Maximum	179,447	1,7331	-90,2545	9,83E-02	8,03E-02	3,60E-01
Minimum	179,413	1,72076	-90,2561	9,82E-02	8,03E-02	3,60E-01
Maximaal positieve afwijking	5,33E-03	4,99E-03	7,98E-04	2,71E-06	2,64E-06	3,20E-05
Maximaal negatieve afwijking	2,87E-02	7,36E-03	8,02E-04	1,23E-05	3,06E-06	4,70E-05

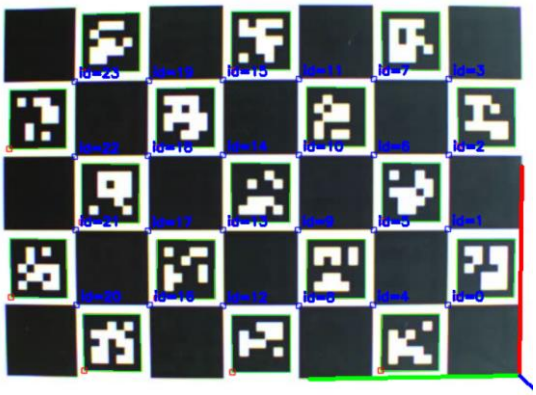
Conclusie

Analoog met de ArUco-markers deelt het ArUco-paneel dezelfde eigenschap, namelijk dat de nauwkeurigheid afhankelijk is van de loodrechte afstand tot de camera. Over de nauwkeurigheid van de meeting kan er helaas niets geconcludeerd worden. Dit omdat de werkelijke positie van het ArUco-paneel ten opzichte van de camera niet gekend is en het niet mogelijk is om de onderlinge afstand te berekenen. Bij het vergelijken van de precisie voor de positie en de pose van het ArUco-paneel (Tabel 10 en 11 met Tabel b. 5 en b. 8) met die van de markers kan er duidelijk geconcludeerd worden dat het ArUco-paneel accurater is.

3.4.3.4 ChArUco-paneel

Aangezien het ChArUco-paneel analoog met het ArUco-paneel werkt is er hier ook sprake van uitschieters bij de markers wat effect heeft op de nauwkeurigheid van het ArUco-paneel. Aangezien het ChArUco-paneel de positie bepaalt aan de hand van hoekpunten van het geïntegreerd schaakbordpatroon, wordt er verwacht dat het nauwkeuriger is. Dit is terug te vinden in Figuur 65 waarop de hoekpunten een ID worden gegeven in plaats van aan de markers.

Aangezien deze hoekpunten betere features zijn om te detecteren wordt er verwacht dat ze nauwkeuriger zijn. Voor het ChArUco-paneel zijn er twee metingen gedaan, één op afstand (700 - 800 mm) en één van dichterbij (300 - 400 mm).



Figuur 65: Verwerking van het ChArUco-paneel

Resultaten en discussie

In Tabel 12 en Tabel 13 staan de resultaten van de nauwkeurigheidsmeting voor het ChArUco-paneel op de twee afstanden. Ook bij het ChArUco-paneel speelt de loodrechte afstand van de camera tot het paneel een rol in de nauwkeurigheid.

Tabel 12: Nauwkeurigheid ChArUco-paneel van een constant beeld (700-800mm)

ChArUco-paneel						
	Rotatie (rad)			Translatie (m)		
	X	Y	Z	X	Y	Z
Gemiddelde	1,79E+02	-2,51E+00	1,80E+02	1,22E-01	-1,01E-01	7,24E-01
Standaarddeviatie	4,08E-02	5,23E-02	2,66E-03	1,32E-05	7,22E-06	1,68E-04
Maximum	1,79E+02	-2,19E+00	1,80E+02	1,22E-01	-1,01E-01	7,24E-01
Minimum	1,79E+02	-2,66E+00	1,80E+02	1,22E-01	-1,01E-01	7,23E-01
Maximaal positieve afwijking	1,05E-01	3,13E-01	6,25E-03	4,83E-05	1,57E-05	7,25E-04
Maximaal negatieve afwijking	1,01E-01	1,54E-01	8,75E-03	4,27E-05	1,93E-05	4,12E-04

Tabel 13: Nauwkeurigheid ChArUco-paneel van een constant beeld (350-450mm)

ChArUco-paneel						
	Rotatie (rad)			Translatie (m)		
	X	Y	Z	X	Y	Z
Gemiddelde	1,80E+02	1,08E+00	-9,01E+01	1,06E-01	8,49E-02	4,08E-01
Standaarddeviatie	5,58E-03	7,80E-03	7,93E-04	2,08E-06	2,21E-06	2,21E-05
Maximum	1,80E+02	1,10E+00	-9,01E+01	1,06E-01	8,49E-02	4,08E-01
Minimum	1,80E+02	1,06E+00	-9,01E+01	1,06E-01	8,48E-02	4,07E-01
Maximaal positieve afwijking	1,11E-02	1,92E-02	1,90E-03	4,90E-06	5,66E-06	5,61E-05
Maximaal negatieve afwijking	1,79E-02	2,16E-02	1,70E-03	5,10E-06	5,34E-06	5,59E-05

Bij het vergelijken van het ChArUco-paneel (Tabel 12) met het ArUco-paneel (Tabel 10) is het kunnen we stellen dat het ChArUco-paneel nauwkeuriger is dan het ArUco-paneel. Zoals eerder vermeld is een mogelijke reden dat de hoekpunten van het ChArUco-paneel betere features zijn om te detecteren.

Conclusie

Het ChArUco-paneel is nauwkeuriger dan het ArUco-paneel. Ook geldt voor het ChArUco-paneel dat de nauwkeurigheid afhankelijk is van de loodrechte afstand van de camera tot het paneel.

3.4.3.5 Diamantmarker

In de literatuurstudie beschrijft men de Diamantmarker als een combinatie van het ChArUco-paneel en de ArUco-markers. Hierdoor wordt verwacht dat de Diamantmarkers nauwkeuriger zijn dan de ArUco-markers.

Resultaten en discussie

In de CSV-file zouden er maar twee types Diamantmarkers mogen zijn, zijnde [1, 3, 4, 6] en [11, 13, 14, 16]. Dit is echter niet het geval, uit 110 frames werd dertien keer de marker [6, 11, 4, 13] gedetecteerd. De reden hiervoor is onbekend maar kan mogelijk toegeschreven worden aan het feit dat de ArUco-markers in één vlak liggen en het programma één Diamantmarker zoekt in het beeld. Dit heeft als gevolg dat de twee gewenste markers een detectieverlies van 11.8% voor deze meting hebben. De resultaten van de Diamantmarkers staan in Tabel b. 10 in de bijlage. Deze foutieve detecties zijn mogelijk te verwijten aan meerdere Diamantmarkers die dicht bijeen in één vlak liggen.

In Tabel 14 en Tabel 15 staat de verwerking van de Diamantmarkers [1, 3, 4, 6] en [11, 13, 14, 16]. Aan de hand van deze gemiddelde waarden wordt de gemiddelde aftand tussen de twee markers berekend. Dit is 0.139596 meter, wat zeer accuraat is.

Tabel 14: Nauwkeurigheid Diamantmarker [1, 3, 4, 6] van een constant beeld (700-800mm)

Marker [1, 3, 4, 6]						
	Rotatie (rad)			**Translatie (m)**		
	X	**Y**	**Z**	**X**	**Y**	**Z**
Gemiddelde	1,91E+02	9,62E+00	1,81E+02	-4,60E-02	6,21E-02	7,33E-01
Standaarddeviatie	2,29E-01	2,76E-01	3,05E-02	2,08E-05	2,14E-05	2,26E-04
Maximum	1,91E+02	1,01E+01	1,81E+02	-4,59E-02	6,21E-02	7,33E-01
Minimum	1,90E+02	8,95E+00	1,81E+02	-4,61E-02	6,20E-02	7,32E-01
Maximaal positieve afwijking	6,63E-01	4,39E-01	9,16E-02	1,03E-04	5,40E-05	4,92E-04
Maximaal negatieve afwijking	4,70E-01	6,66E-01	6,14E-02	3,52E-05	7,55E-05	1,11E-03

Tabel 15: Nauwkeurigheid Diamantmarker [11, 13, 14, 16] van een constant beeld (700-800mm)

Marker [11, 13, 14, 16]						
	Rotatie (rad)			Translatie (m)		
	X	Y	Z	X	Y	Z
Gemiddelde	1,74E+02	5,64E+00	1,79E+02	-4,77E-02	-7,75E-02	7,30E-01
Standaarddeviatie	7,62E-01	3,26E+00	3,43E-01	5,95E-05	5,93E-05	4,94E-04
Maximum	1,77E+02	9,82E+00	1,80E+02	-4,76E-02	-7,73E-02	7,31E-01
Minimum	1,72E+02	-1,18E+00	1,78E+02	-4,78E-02	-7,76E-02	7,29E-01
Maximaal positieve afwijking	3,55E+00	4,19E+00	7,50E-01	1,20E-04	1,53E-04	8,80E-04
Maximaal negatieve afwijking	1,90E+00	6,82E+00	5,37E-01	1,12E-04	8,74E-05	1,19E-03

Conclusie

Uit dit onderzoek wordt besloten dat de Diamantmarker nauwkeuriger is dan de ArUco marker. Dit is te wijten aan de goede eigenschappen overgenomen van het ChArUco-paneel, namelijk dat het gedetecteerd wordt aan de hand van schaakbordpatroon hoekpunten. Maar bij het gebruik van meerdere Diamantmarkers die dicht bijeen in één vlak liggen ontstaan er detectiefouten.

3.4.3.6 Algemene conclusie

Om de vier types met elkaar te vergelijken en een waarde voor nauwkeurigheid uit te drukken, worden er radardiagrammen gemaakt voor elk type marker van de metingen op afstand. Met de zes assen staande voor de zes gemeten vrijheidsgraden van de markers: de rotatie rond de X, Y en Z-as en de translatie volgens de X, Y en Z-as. Van een radardiagram is het mogelijk om de oppervlakte te berekenen (verderop wordt uitgelegd op welke manier), deze oppervlakte zal gebruikt worden als evaluatiecriteria voor de markers. Zo kan deze waarde gebruikt worden om de 4 types markerdetectie onderling te vergelijken.

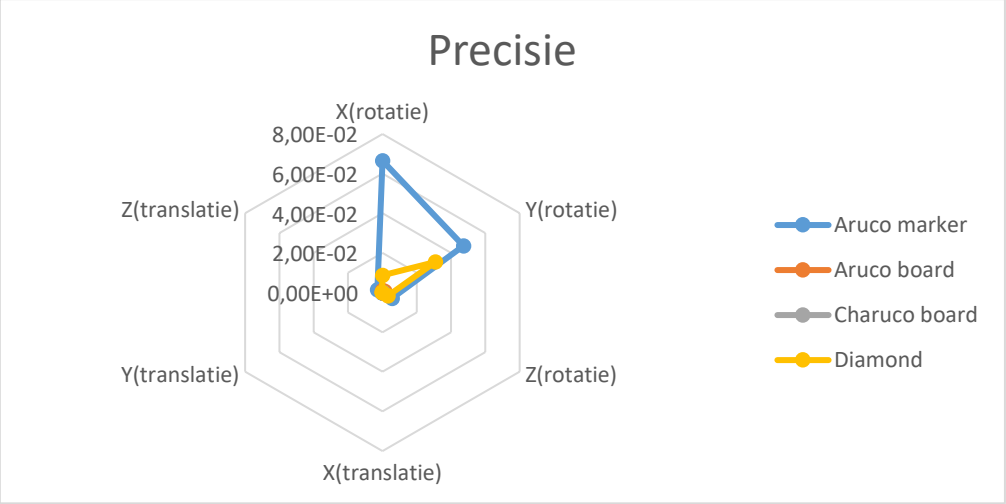
Figuur 66 weergeeft het radardiagram voor de precisie van de verschillende markers. De gegevens hiervoor komen uit Tabel 16. In zowel Tabel 16 als Figuur 66 is te zien dat de precisie van het ArUco-paneel en ChArUco-paneel significant beter is, waardoor ze nauwelijks tot zelfs niet waarneembaar zijn. Daarom zal Figuur 67 enkel de precisie van het ArUco en ChArUco-paneel tonen.

Uit de radardiagrammen is het duidelijk dat het ChArUco-paneel de hoogste precisie behaald, gevolgd door het ArUco-paneel, vervolgens de Diamantmarker en ten laatste de ArUco-markers. Dit is zoals verwacht en beschreven in de literatuurstudie, de panelen zijn accurater dan de markers omdat een bord meer informatie heeft om een schatting op te stellen. Ook hebben het ChArUco-paneel en de Diamantmarkers een betere nauwkeurigheid omdat ze

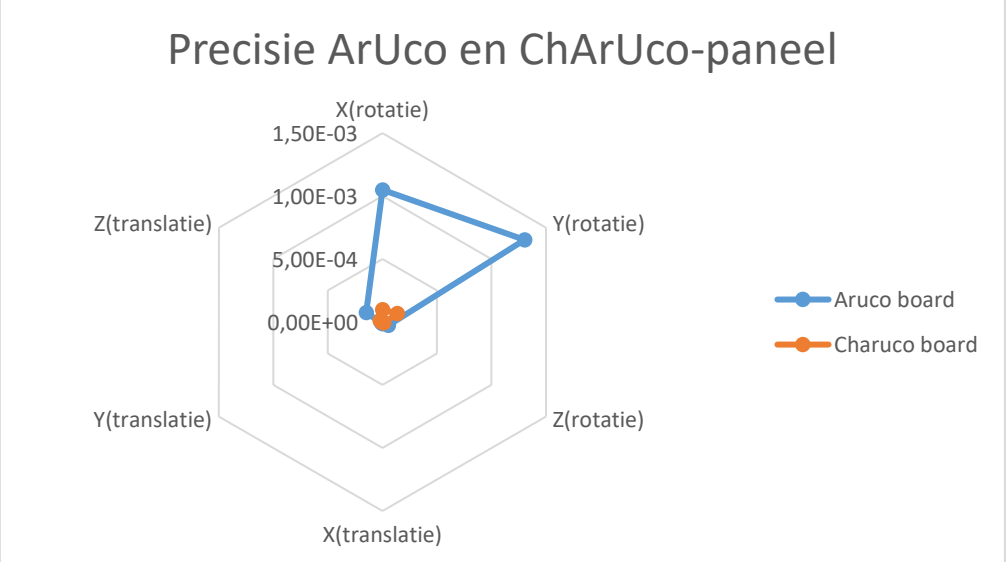
gebruik maken van betere features (de hoekpunten) voor de detectie en schatting van de positie en locatie.

Tabel 16: Gegevens precisie van de vier markertypes van de ArUco-module (in radialen/meter)

	ArUco marker	ArUco-paneel	ChArUco-paneel	Diamantmarker
X (rotatie)	6,65E-02	1,05E-03	9,73E-05	8,65E-03
Y (rotatie)	4,72E-02	1,30E-03	1,37E-04	3,09E-02
Z (rotatie)	5,73E-03	5,37E-05	1,38E-05	3,26E-03
X (translatie)	2,04E-04	1,23E-05	2,08E-06	4,01E-05
Y (translatie)	2,26E-04	1,09E-05	2,21E-06	4,04E-05
Z (translatie)	2,94E-03	1,49E-04	2,21E-05	3,60E-04



Figuur 66: Radardiagram voor de vier markertypes van de ArUco-module



Figuur 67: Radardiagram voor het ArUco en ChArUco-paneel

Om een waarde op de precisie te zetten wordt de oppervlakte van de radardiagrammen berekend voor elk markertype. Dit wordt gedaan door de oppervlakte van zes driehoeken op te tellen. Vervolgens wordt de oppervlakte van elke driehoek met formule (3.2) berekend. Met a en b de lengte van de zijdes van de driehoek (waardes van de precisie) en de tussenliggende hoek C (van 60°). De resultaten staan in Tabel 17.

$$oppervlakte = \frac{a\,.\,b\,.\,cos(C)}{2} \qquad (3\,.\,2\,)$$

Voor de nauwkeurigheid van de markertypes wordt de gemiddelde afwijking tussen twee markers met de werkelijkheid vergeleken, maar aangezien het ArUco en ChArUco-paneel onderling niet vergeleken kan worden, is het niet mogelijk om een correcte nauwkeurigheid uit te drukken. Daarom zal er enkel een nauwkeurigheid worden uitgedrukt voor de ArUco en Diamantmarkers, ook deze resultaten staan in Tabel 17. Hiervoor werd gekeken naar de gemiddelde afwijking voor de onderlinge afstand en het verschil in hun pose. Vervolgens werd de oppervlakte onder de grafiek voor deze twee punten berekend om ze onderling te vergelijken. Uit deze vergelijking is blijkt de Diamantmarker het nauwkeurigste te zijn.

Tabel 17: Resultaten nauwkeurigheid constante meeting (in meter)

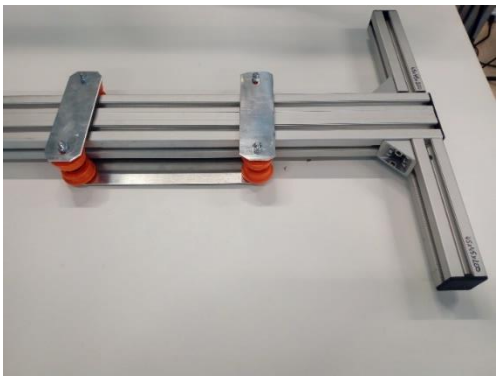
	ArUco marker	ArUco-paneel	ChArUco-paneel	Diamantmarker
precisie	1,56E-03	6,91E-07	7,56E-09	1,61E-04
juistheid	2,12E-04			1,26E-04

3.4.4 Metingen voor een lineaire beweging

In dit onderdeel van het nauwkeurigheidsstudie wordt er gekeken of de nauwkeurigheid van de markers verandert door beweging van de markers. In het geval dat er een verschil is tussen markers in beweging en stationaire markers wordt er onderzocht hoe groot dit verschil is.

4.2.4.1 Methodiek

De aanpak bij dit onderdeel zal grotendeels hetzelfde verlopen als bij het meten van een constant beeld, alleen zullen ditmaal alle types markers gemanipuleerd worden. De manipulatie in dit onderzoek zal aan de hand van een lineaire geleiding (Figuur 68) gebeuren. Dit wordt ‚net als het vorig onderdeel, gefilmd en vervolgens gebruikt om de verschillende detectieprogramma's uit te voeren. Het is hier belangrijk om op te merken dat de lineaire Geleiding ook onderhevig is aan speling rond zijn z-as.



Figuur 68: Lineaire geleiding voor Nauwkeurigheidsonderzoek

Er kan niet gegarandeerd worden dat de camera exact parallel bevestigd is ten opzichte van het werkvlak. Daardoor is er sprake van een lineaire beweging in de 3D (XYZ). Het voordeel van een lineaire beweging (een rechte) in 3D is dat elke 2D projectie hiervan ook een lineaire beweging (een rechte) beschrijft. Dit echter met uitzondering van twee punten; het punt waarvan de beweging weggaat en het punt waar het naar toe gaat. Dit zorgt ervoor dat de 3D beweging kan bekeken worden in drie 2D-projecties zijnde: XY, XZ en YZ-dimensies. Om uit deze 2D-projecties een nauwkeurigheid te bekomen, zal er een rechte gemaakt worden vanuit het beginpunt tot het eindpunt. Voor deze twee punten wordt het gemiddelde van de punten genomen wanneer de marker voor en na de beweging stilstaat. De rechte stelt dan de 2D-projectie voor van de 3D lineaire beweging. Op basis van deze rechte zal elk punt tijdens de beweging worden vergeleken aan de hand van volgende formule (3.3) voor de loodrechte afstand van een punt tot een rechte. Met a, b en c uit de vergelijking $aX + bY + c = 0$ en (X_i, Y_i) de coördinaten van een gemeten punt. Hierbij zal a de richtingscoëfficiënt van de rechte zijn, b zal gelijk aan min één zijn en c gelijk aan het snijpunt van de rechte met de Y-as. Van deze afwijkingen zal één algemeen gemiddelde worden genomen als waarde voor de nauwkeurigheid.

$$\text{Loodrechte afstand punt tot rechte} = \frac{|aX_i + bY_i + C|}{\sqrt{a^2 + b^2}} \qquad (3.3)$$

4.2.4.2 ArUco-markers

Er wordt gestart met de ArUco-markers aangezien zij de basis zijn van alle andere types markers. Analoog hiervoor zal hetzelfde filmpje gebruikt worden voor de ArUco-markers en het ArUco-paneel. In dit onderzoek zal, net zoals voor het onderzoek zonder manipulatie, (constant beeld) de ground truth vergeleken worden met de resultaten van de metingen. Ook zullen de resultaten vergeleken worden met de metingen zonder manipulatie om te achterhalen welk het effect van een beweging heeft op de nauwkeurigheid. Aangezien er bewezen is dat de distorsies een effect hebben op de nauwkeurigheid, worden de berekeningen éénmaal op markers aan de rand van het scherm uitgevoerd en éénmaal voor markers in het midden van het scherm.

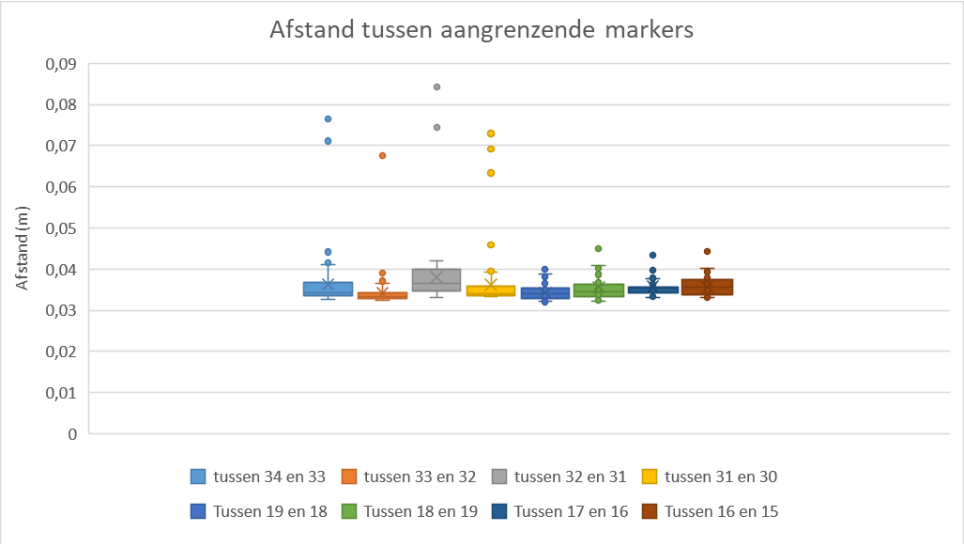
Resultaten en discussie

Ten eerste worden de onderlinge afstanden vergeleken met de ground truth, hiervan staan de resultaten van twee rijen markers in Tabel 18. De eerste rij markers liggen aan de rand van het scherm en zijn de markers 34,33,32,31 en 30. De tweede rij markers zijn gelegen in het midden van het scherm en zijn de markers 19,18,17,16 en 15.

Tabel 18: Resultaten onderlinge afstanden tussen markers voor lineaire beweging (in meter)

	Rand	Midden
Gemiddelde	3,65E-02	3,51E-02
Standaarddeviatie	7,44E-03	2,06E-03
Maximum	8,42E-02	4,43E-02
Minimum	3,25E-02	3,21E-02

Als Tabel 18 nu vergeleken wordt met de ground truth, waarbij een onderlinge afstand van 0,033 m is, is het zeer duidelijk dat er hier een afwijking is. Dit was ook te verwachten aangezien deze systematische fout ook aanwezig is bij een stilstaand beeld. Bij het vergelijken van de resultaten met de resultaten van een constant beeld kan de conclusie getrokken worden dat die systematische fout groter is bij manipulatie. Ook de precisie is gestegen bij manipulatie, dit is hierbij ook te danken aan uitschieters. Als men kijkt naar het maximum kan er duidelijk vastgesteld worden dat dit een uitschieter is. Deze uitschieters zullen het gemiddelde en de standaarddeviatie doen stijgen. Net zoals bij de metingen voor een constant beeld zijn er geen duidelijke oorzaken die deze uitschieters als gevolg kunnen hebben. In Figuur 69 zijn er boxplotten gemaakt voor elke onderlinge afstand apart en ook hierbij kan gezien worden dat er sprake is van uitschieters. Ook kan er vastgesteld worden dat deze uitschieters veel frequenter voorkomen aan de rand van het scherm dan in het midden.



Figuur 69: Boxplot van de afstanden tussen de aangrenzende markers

Zoals in de methodiek vermeld staat, kan de lineaire beweging in alle drie de 2D-dimensies worden gecontroleerd op nauwkeurigheid. Dit zorgt voor een individuele nauwkeurigheid voor elke marker in het ArUco-paneel (Tabel 19). Maar om één algemene waarde voor de nauwkeurigheid bij de ArUco-markers te bekomen, zal er het gemiddelde van de gemiddelde afwijkingen van de 34 markers genomen worden, deze waarde is $3{,}96E^{-3}$m. Voor de precisie uit te drukken zal de standaarddeviatie van de afwijkingen genomen worden, die voor de markers $7{,}44E^{-3}$m is.

Tabel 19: Tabel met waardes voor nauwkeurigheid voor elke marker (in meter)

34	**33**	**32**	**31**	**30**
3,41E-03	4,21E-03	3,12E-03	5,63E-03	3,47E-03
29	**28**	**27**	**26**	**25**
3,87E-03	3,05E-03	4,07E-03	3,29E-03	3,76E-03
24	**23**	**22**	**21**	**20**
4,20E-03	4,06E-03	3,28E-03	3,47E-03	4,78E-03
19	**18**	**17**	**16**	**15**
2,75E-03	2,02E-03	3,08E-03	3,70E-03	3,70E-03
14	**13**	**12**	**11**	**10**
3,76E-03	4,55E-03	3,68E-03	7,08E-03	3,16E-03
9	**8**	**7**	**6**	**5**
4,48E-03	5,29E-03	3,19E-03	3,16E-03	3,78E-03
4	**3**	**2**	**1**	**0**
3,12E-03	4,52E-03	4,46E-03	8,29E-03	3,05E-03

Conclusie

Uit de resultaten kan er geconcludeerd worden dat het fenomeen van uitschieters ook aanwezig is bij markers in beweging. Deze uitschieters zijn echter nog sterker aanwezig dan bij de constante meting. Wat als resultaat geeft dat de nauwkeurigheid minder goed is voor markers in beweging.

4.2.4.3 ArUco en ChArUco-paneel

Bij het ArUco en ChArUco-paneel worden enkel de berekeningen voor de nauwkeurigheid in de 2D-dimensies uitgevoerd, aangezien het ArUco-paneel niet vergeleken kan worden met een ander paneel.

Resultaten en discussie

Uit deze berekeningen bekomt men een waarde van $6{,}79E^{-4}$m voor het ArUco-paneel en $3{,}58E^{-4}$ m voor het ChArUco-paneel. Als deze vergeleken wordt met de nauwkeurigheid van de ArUco-markers (is $3{,}96E^{-3}$ m) kan er geconcludeerd worden dat het ArUco-paneel beter is dan de markers en dat het ChArUco-paneel zelfs nog nauwkeuriger is. De precisie van de

beweging is voor het ArUco-paneel $5{,}45E^{-4}$ m en voor het ChArUco-paneel $2,77E^{-4}$ m, dus hier kan hetzelfde als voor juistheid geconcludeerd worden aangezien de precisie voor ArUco-markers $7{,}44E^{-3}$ m is.

Conclusie

Voor de nauwkeurigheid bij beweging van het ArUco en ChArUco-paneel kan er verondersteld worden dat de ArUco-module zeer accuraat is. Dit betekent dat de panelen betrouwbaar zijn voor mobiele toepassingen.

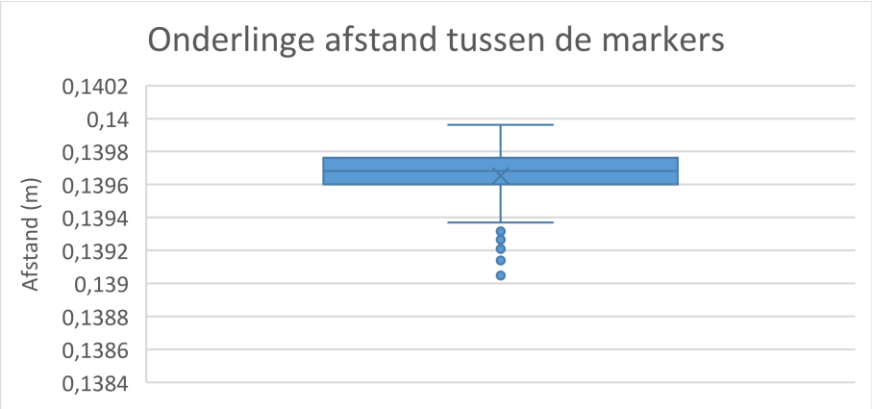
4.2.4.4 Diamantmarker

Voor de Diamantmarker kan hetzelfde uitgevoerd worden als voor de ArUco-markers, namelijk hun onderlinge afstand vergelijken met de ground truth. Ze worden ook vergeleken aan de hand van de 2D projectie methode.

Resultaten en discussie

Aangezien er voor de lineaire meeting dezelfde opstelling is gebruikt als die van bij de metingen van een constant beeld, werden er foutieve markers gedetecteerd. Waarvan veertien keer de markers [3, 6, 4, 11] en vijf keer de markers [6, 11, 4, 13] uit de 117 frames. Dit laat blijken dat er meer foutieve metingen en meer soorten foutieve markers zijn bij de metingen van de bewegende markers. Ook ditmaal is er geen eenduidige verklaring waarom dit gebeurt, maar een mogelijke oorzaak is dat alle markers dezelfde positie hebben en dat het programma daardoor ze probeert als één marker voor te stellen. In dit geval zelfs foutieve Diamantmarkers uit aanwezige ArUco-markers.

In Figuur 70 is de boxplot van de onderlinge berekende afstand tussen de twee Diamantmarkers te zien. In Tabel 7 kan vastgesteld worden dat de werkelijke afstand tussen de twee markers 14 cm bedraagt, wanneer dit vergeleken wordt met het gemiddelde van 13,965 cm is het verschil minder dan 0.5 millimeter.



Figuur 70: Boxplot onderlinge afstand tussen twee Diamantmarkers

De resultaten voor de nauwkeurigheid en precisie van de beweging bij de Diamantmarkers is respectievelijk $5{,}45E^{-4}$ m en $4{,}37E^{-3}$m.

Conclusie

Wanneer de nauwkeurigheid of correctheid en precisie van beweging van de Diamond vergeleken wordt met die van het ArUco-paneel en ChArUco-paneel kan men stellen dat deze minder nauwkeurig is. Het is wel nog steeds nauwkeuriger dan de ArUco-markers maar bij deze metingen is er sprake van foutieve markers. Dit is dus een groot minpunt. Deze foutieve markers kunnen in de toekomst er wel gemakkelijk uitgefilterd worden aangezien de gebruikte markers gekend zijn.

4.2.4.5 Conclusie

In Tabel 20 worden de precisie en correctheid voor de 4 markertypes van de ArUco-module voor een lineaire beweging vergeleken. Met de correctheid het gemiddelde van de afwijking van de "perfecte" beweging en de precisie de standaarddeviatie van deze afwijkingen. Het is belangrijk om deze waardes niet te vergelijken met die van de constante meting aangezien er hierbij enkel de precisie en juistheid berekend wordt aan de hand van de positie.

Tabel 20: Resultaten nauwkeurigheid voor lineaire beweging (in meter)

	ArUco marker	ArUco-paneel	ChArUco-paneel	Diamantmarker
Precisie beweging	7,44E-03	5,45E-04	2,77E-04	4,37E-04
Juistheid beweging	3,96E-03	6,79E-04	3,58E-04	5,45E-04

3.5 Conclusie

Uit de resultaten van het nauwkeurigheidsstudie in Tabel 17 en Tabel 20 blijkt dat het ChArUco-paneel het nauwkeurigst is. Maar zoals in het begin reeds is vermeld, is er een verhoudingsverschil tussen het ArUco en ChArUco-paneel. Het ChArUco-paneel is groter dan het ArUco-paneel en zal daarom nauwkeuriger zijn. Dit is bewezen in de metingen voor cameravervormingen en bij de constante metingen voor dichtbij en van op een afstand. Ook zijn de features van het schaakbordpatroon (geïntegreerd in het ChArUco-paneel) inderdaad nauwkeuriger voor detectie van markers. Deze features zijn daarboven ook beter bij detectie van bewegende markers, dit wordt aangetoond in Tabel 20. Hier zijn de Diamantmarkers zelfs nauwkeuriger dan het ArUco-paneel ondanks dat ze veel kleiner zijn.

Ondanks het verhoudingsverschil tussen het ChArUco en ArUco-paneel is het ChArUco-paneel nauwkeuriger en heeft daarom ook de voorkeur bij detectie waarbij de nauwkeurigheid essentieel is. Maar in het geval dat er meer dan één object tegelijk

gedetecteerd moet worden, zal geen van beide panelen aan de eisen voldoen. Daarom zal er hiervoor een keuze gemaakt moeten worden tussen de Diamantmarkers en de ArUco-markers. Hierbij heeft de ArUco marker het grote voordeel dat ze in proportie veel groter kunnen zijn dan de Diamantmarker aangezien deze vier ArUco-markers bevat. Daarom moet er dus afgewogen worden hoe belangrijk de afstand tot de camera is aangezien deze in verhouding staat met de nauwkeurigheid van de meting en de nodige grootte van de markers.

4 Casussen

4.1 Kalibratie bewegingsmodel gidsrobot

Bij de Kalibratie van het bewegingsmodel voor de gidsrobot zal er een zo goed mogelijke oplossing gevonden moeten worden voor het kalibreren van het bewegingsmodel van een gidsrobot. Dit onderzoek werd gevoerd voor de masterproef 'Ontwerpen, kalibreren en aansturen van een mobiele autonome robot' van Tijs Cardeynaels en Simone Del Gallo voor het onderzoekscentrum ACRO.

4.1.1 Situering

Het doel van de gidsrobot is om mensen te ontvangen en ze op ACRO rond te leiden. Dit betekent dus dat de gidsrobot zowel aan (*padplanning)* routebepaling als obstakelvermijding moet doen, maar vooraleer dit kan gebeuren moet de robot omgebouwd en gekalibreerd worden om zich te kunnen navigeren op ACRO. Dit onderzoek omvat de masterproef van Simone en Tijs, meer specifieker zal de gidsrobot worden afgesteld aan de hand van de intrinsieke parameters van de robot. Deze nodige parameters beschrijven de inwendige afmetingen en eigenschappen van de mechanische opstelling van de robot. Deze afmetingen zijn de straal van de wielen en de afstand tussen de wielen in.

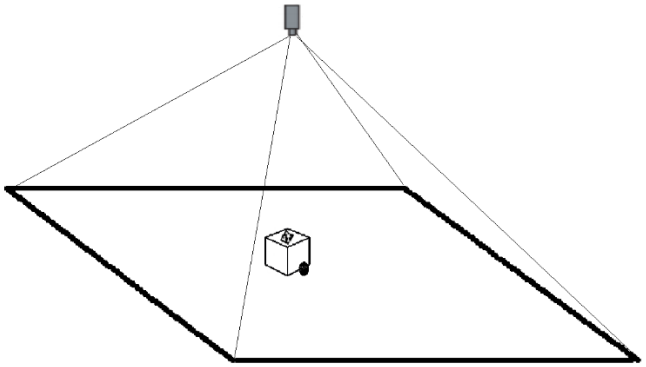
4.1.2 Probleemstelling

Door een probleem in de aansturing van de motoren, draaide één van de twee motoren sneller, met als gevolg dat de aangedreven wielen van de gidsrobot niet even snel draaien. Hierdoor zal de gidsrobot een bocht maken op het moment dat deze aangestuurd wordt om vooruit te rijden, ondanks dat de motoren dezelfde spanning en sturing krijgen vanuit de (micro)controller. Deze afwijking valt softwarematig op te lossen door de sturing van de motoren en de werkelijke beweging hiervan op elkaar af te stellen. Ook zijn deze afwijkingen te interpreteren als een intrinsieke parameter die deel uitmaken van het bewegingsmodel van de robot. Het is dus geweten dat er een afwijking aanwezig is, maar de maat van de afwijking is niet gekend. Op het moment van de start van deze casus is er nog geen oplossing voor het vinden van de exacte afwijking die de wielen ten opzichte van elkaar hebben.

4.1.3 Methodiek

Voor het kalibreren van het bewegingsmodel van de gidsrobot zal de detectie van de ArUco-markers op twee verschillende methodes toegepast worden. Voor de eerste opstelling zal een camera van bovenaanzicht de robotgids (ArUco marker) detecteren, om vervolgens het traject van de gidsrobot te bekomen. Dit traject zal dan vergeleken worden met de sturingen voor de

gidsrobot met als resultaat de relatieve afwijking van de gidsrobot. In Figuur 71 is opstelling 1 afgebeeld.



Figuur 71: Opstelling 1: bovenaanzicht

4.1.4 Resultaten en discussie

In Hoofdstuk 3.4 staan de aanpassingen die reeds gemaakt zijn aan de originele OpenCV-code om de detectie van de markers in 3D t.o.v. de camera uit te schrijven. Ondanks dit programma in C++ geschreven is zal door de toevoeging van deze aanpassingen het programma niet real time kunnen runnen. Daarom wordt er gekozen om het programma als verwerkingsstap te gebruiken. Dit betekent dat de bewegingen van de gidsrobot (marker) gefilmd worden en vervolgens door dit programma gevoerd worden. Hierdoor heeft het programma een grotere tijdspanne voor het verwerken van elke frame dan dat het rechtstreeks met de camera zou gebeuren. Dit geeft echter wel een probleem met zich mee, dit dat de tijd die nu door middel van de functie `GetTickCount()` wordt verkregen, niet de werkelijke tijd is maar die van de verwerking. Daarom moet men bij het filmen van de gidsrobot (marker) de tijd (UTC-tijd) per frame bijhouden, dit kan ook door middel van een extern bestand. Het verkregen bestand samen met het resultaat (CSV-bestand) van de verwerking van de frames geven vervolgens hetzelfde resultaat als de verwerking van de beelden sneller dan de framerate zou zijn. In Tabel b. 11 (bijlage) zijn de resultaten voor de verwerking van een ChArUco-paneel met de tijd voorgesteld. Mits enkele verwerkingsstappen in Excel worden de waardes omgezet in graden (°), dit is in Tabel 21 te zien. Belangrijk is om hierbij op te merken dat de resultaten in Tabel 21 voor rechtstreekse verwerking (dus niet real time snelheid) zijn, maar dit verschilt niet met de andere aanpak in resultaten.

Tabel 21: Verwerkte resultaten van het CSV-bestand

Rotatie (°)			Translatie (m)			Time (s)
X	Y	Z	X	Y	Z	
1,80E+02	7,77E-01	2,49E+00	-3,37E-02	1,24E-01	4,62E-01	2,46E-04
1,80E+02	7,88E-01	2,40E+00	-3,36E-02	1,24E-01	4,63E-01	2,11E-01
1,80E+02	8,06E-01	2,49E+00	-3,36E-02	1,24E-01	4,63E-01	4,17E-01
1,80E+02	7,73E-01	2,15E+00	-3,37E-02	1,24E-01	4,62E-01	6,01E-01
1,81E+02	7,77E-01	1,64E+00	-3,37E-02	1,24E-01	4,63E-01	7,96E-01
1,80E+02	7,91E-01	2,98E+00	-3,36E-02	1,24E-01	4,63E-01	9,75E-01
1,80E+02	7,77E-01	2,70E+00	-3,37E-02	1,24E-01	4,62E-01	1,18E+00
1,80E+02	8,03E-01	2,65E+00	-3,36E-02	1,24E-01	4,63E-01	1,39E+00
1,80E+02	7,95E-01	2,60E+00	-3,36E-02	1,24E-01	4,63E-01	1,59E+00
1,80E+02	7,49E-01	2,80E+00	-3,37E-02	1,24E-01	4,62E-01	1,79E+00
1,80E+02	7,64E-01	2,79E+00	-3,37E-02	1,24E-01	4,62E-01	1,99E+00
1,80E+02	7,75E-01	2,47E+00	-3,37E-02	1,24E-01	4,62E-01	2,19E+00
1,80E+02	7,65E-01	2,63E+00	-3,37E-02	1,24E-01	4,62E-01	2,39E+00
1,80E+02	7,62E-01	2,58E+00	-3,37E-02	1,24E-01	4,62E-01	2,57E+00
1,80E+02	7,76E-01	2,17E+00	-3,37E-02	1,24E-01	4,62E-01	2,75E+00
1,80E+02	7,95E-01	2,22E+00	-3,36E-02	1,24E-01	4,63E-01	2,97E+00
1,80E+02	8,19E-01	2,36E+00	-3,36E-02	1,24E-01	4,63E-01	3,18E+00
1,80E+02	7,88E-01	2,90E+00	-3,37E-02	1,24E-01	4,63E-01	3,41E+00
1,81E+02	7,99E-01	1,82E+00	-3,36E-02	1,24E-01	4,63E-01	3,62E+00
1,80E+02	7,63E-01	2,66E+00	-3,37E-02	1,24E-01	4,62E-01	3,81E+00
1,80E+02	7,73E-01	2,55E+00	-3,37E-02	1,24E-01	4,62E-01	4,03E+00
1,80E+02	7,54E-01	2,83E+00	-3,37E-02	1,24E-01	4,62E-01	4,23E+00
1,80E+02	8,10E-01	2,48E+00	-3,36E-02	1,24E-01	4,63E-01	4,47E+00
1,80E+02	7,88E-01	1,87E+00	-3,37E-02	1,24E-01	4,63E-01	4,68E+00
1,80E+02	7,80E-01	2,12E+00	-3,37E-02	1,24E-01	4,62E-01	4,86E+00
1,80E+02	-8,21E-01	-2,36E+00	-3,36E-02	1,24E-01	4,63E-01	5,07E+00
1,80E+02	7,61E-01	2,81E+00	-3,37E-02	1,24E-01	4,62E-01	5,25E+00
1,80E+02	7,65E-01	2,73E+00	-3,37E-02	1,24E-01	4,62E-01	5,47E+00
1,80E+02	7,51E-01	2,80E+00	-3,37E-02	1,24E-01	4,62E-01	5,69E+00
1,80E+02	7,67E-01	2,63E+00	-3,37E-02	1,24E-01	4,62E-01	6,09E+00
1,80E+02	7,67E-01	2,85E+00	-3,37E-02	1,24E-01	4,63E-01	6,31E+00
1,80E+02	7,84E-01	3,08E+00	-3,37E-02	1,24E-01	4,63E-01	6,49E+00
1,80E+02	7,76E-01	2,38E+00	-3,37E-02	1,24E-01	4,62E-01	6,69E+00
1,80E+02	8,06E-01	2,71E+00	-3,36E-02	1,24E-01	4,63E-01	6,90E+00
1,80E+02	7,62E-01	2,80E+00	-3,37E-02	1,24E-01	4,62E-01	7,09E+00
1,80E+02	7,53E-01	2,86E+00	-3,37E-02	1,24E-01	4,62E-01	7,28E+00
1,80E+02	7,59E-01	2,81E+00	-3,37E-02	1,24E-01	4,62E-01	7,47E+00
1,80E+02	8,10E-01	2,52E+00	-3,36E-02	1,24E-01	4,63E-01	7,67E+00
1,80E+02	7,67E-01	2,67E+00	-3,36E-02	1,24E-01	4,63E-01	7,85E+00
1,80E+02	7,97E-01	2,56E+00	-3,36E-02	1,24E-01	4,63E-01	8,05E+00
1,81E+02	8,49E-01	2,45E+00	-3,36E-02	1,24E-01	4,63E-01	8,24E+00
1,81E+02	8,43E-01	2,92E+00	-3,36E-02	1,24E-01	4,63E-01	8,42E+00

4.1.5 Conclusie

De kwaliteit van de nauwkeurigheid voor deze metingen is afhankelijk van verschillende parameters zoals: de camera, de lens, de gebruikte marker (zie 3.4), de ruimte, lichtinval, type lichtinval en de grootste de verhouding van de gebruikte marker met het beeld. Voor de metingen van Simone en Tijs is er de keuze gemaakt om de Diamantmarker te gebruiken i.p.v. het aangeraden ChArUco-paneel vanwege de verhouding tussen de grootte van de marker en de afstand tot de camera. Dit heeft uiteraard ook als nadeel dat de nauwkeurigheid wat minder is en daarom is er ook verbetering mogelijk. Ook dit geldt voor de camera, want voor dit type metingen waarbij een groot werkvlak gewenst is, is de kwaliteit van de camera essentieel. Daarom is een betere camera nodig in combinatie met een volledig afgesloten ruimte waarin enkel monochroom wit licht aanwezig is.

4.2 Tracking bij mens-robotsamenwerking

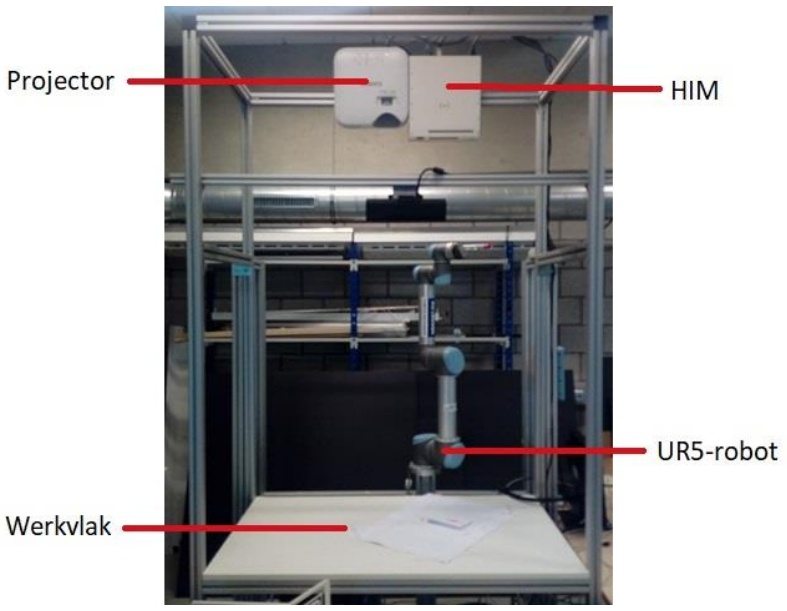
Deze casus dient volledig als ondersteunend onderzoek voor een doctoraatonderzoek dat lopende is op het onderzoekscentrum ACRO. De casus in kwestie is die voor de mens-robotsamenwerking door de doctoraatstudent Martijn Creamer.

4.2.1 Situering

Op de dag van vandaag is er een verschuiving waarneembaar in de richting van grote hoeveelheden aan kleine productieaantallen. Hierdoor is er meer nood aan zowel flexibele als geautomatiseerde systemen, daarom is de oplossing vindbaar bij de mens-robotsamenwerking waarbij de mens een vaste taakvolgorde krijgen toegewezen en afwijken hiervan is vaak uitgesloten. Daarom zal de mens beschouwd worden als een beheersbare entiteit. Tijdens de samenwerking tussen mens en robot zal de robot voortdurend proberen de activiteiten en intenties van de mens te ontdekken. Met de bedoeling dat hij op een gepaste manier anticipeert door deze informatie op te nemen in de taakplanning van de mens.

De samenwerking wordt gerealiseerd met een HIM-interface, de 'Human Interface Mate' zal aan de hand van een 3D-sensor handelingen van een operator monitoren en indien nodig corrigeren. Het corrigeren kan gebeuren door middel van een projector zo aan te sturen dat de mens hierop kan anticiperen of zelfs kan veranderen van aanpak.

Voor deze casus zal er specifiek naar de toepassing voor een assemblagetaak van een mens gekeken worden, waarbij de mens verschillende onderdelen samenvoegt tot één geheel onder toezicht van de robot. In Figuur 72 is te zien hoe de opstelling van de mens-robotsamenwerking er ongeveer zal uit zien.



Figuur 72: Opstelling mens-robotsamenwerking

4.2.2 Probleemstelling

Om het mogelijk te maken dat de HIM de handelingen van de mens superviseert en kan bevestigen, moet het zeker zijn dat de handeling correct zijn uitgevoerd. Dit door de voorwerpen die geassembleerd worden te detecteren en tracken. Ook is er nood aan data-associatie aangezien een assemblage typisch uit meer dan één onderdeel bestaat met de mogelijkheid dat één enkel onderdeel meer dan eens gebruikt wordt. Dit zorgt dus voor drie uitdagingen: Hoe het detecteren van de onderdelen? Hoe tracken (volgen) van elk onderdeel? Hoe de identiteit voor elk onderdeel behouden (data-associatie)? De derde uitdaging legt bij elke detectie de link tussen de gedetecteerde onderdelen en de kennis van de reeds gedetecteerde onderdelen die getrackt worden. Om de volledige tracker stapsgewijs op te bouwen mag er voor deze casus vanuit gegaan worden dat de te tracken objecten gekend zijn, en dat er geen nieuwe objecten in beeld kunnen komen.

4.2.3 Methodiek

Aangezien de uitdagingen die in de probleemstelling staan uitgelegd elkaar opvolgen, is de beschreven volgorde ook de logische manier van aanpak. Dus startend bij uitdaging 1: Hoe het detecteren van objecten? Dan uitdaging 2: Hoe tracken (volgen) van elk onderdeel? En ten slotte uitdaging 3: Hoe de identiteit voor elk onderdeel behouden (data-associatie)?

Voor elk van deze uitdagingen is de logische eerste stap uiteraard onderzoek in vorm van literatuurstudie. Hiervoor zal in Hoofdstuk 2 een uitgebreide literatuurstudie aanwezig zijn. Vervolgens is de 2de stap het implementeren van de nodige software. De derde en laatste stap betreft het uittesten van de software en de evaluatie hiervan.

4.2.3.1 Uitdaging 1: Hoe het detecteren van objecten?

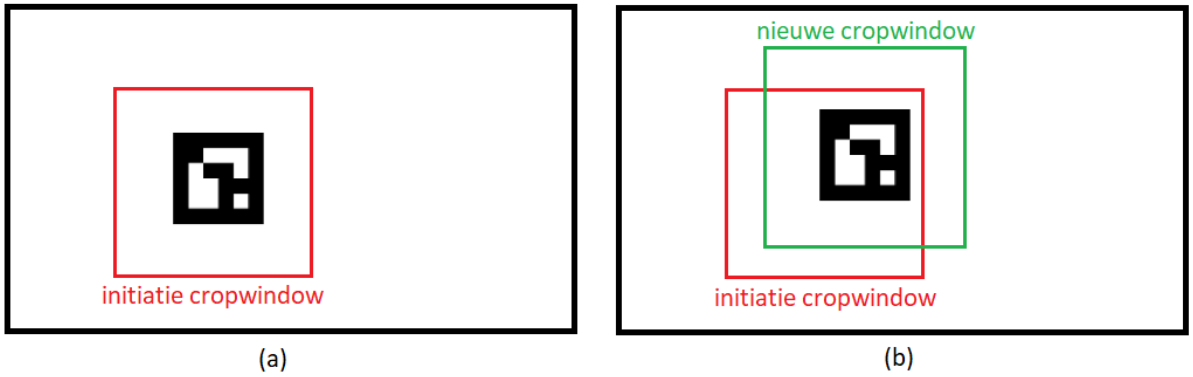
Zoals in Hoofdstuk 3 vermeld is, is de detectie van voorwerpen computationeel intensief. Daarom is er ook voor deze casus gekozen om gebruik te maken van markerdetectie/tracking, namelijk die van de ArUco-module. Dit omdat voor deze masterproef er niet de financiële middelen zijn om computervermogen intensieve algoritmes/programma's te runnen. Daarom zal het enkel een benadering zijn in plaats volledige van een oplossing voor de casus. Maar aangezien de literatuurstudie zeer uitgebreid is, is er de mogelijkheid om de markertracking achteraf, wanneer de middelen ervoor ter beschikking zijn, te vervangen door een algoritme uit Hoofdstuk 2. De kwaliteit en resultaten voor deze detectiemethode staan in detail in Hoofdstuk 3 uitgeschreven.

Als benadering is er ook voor deze casus gekozen om de ArUco-module te gebruiken, maar deze keer is er voor de ArUco-markers gekozen. Dit omdat de nauwkeurigheid van de 3D positie en pose voor deze casus geen prioriteit is maar eerder een extra. Want het tracken van

de objecten gebeurt enkel in het 2D camerabeeld zodat het eenvoudiger is en het volledige systeem sneller kan verlopen.

Voor de code van de detector is het ditmaal niet meer zo eenvoudig als bij de uitvoering van de eerste casus, waar het mogelijk was om aanpassingen aan de originele OpenCV-code te maken. Daarom is er voor de code voor deze casus volledig vanaf nul begonnen, maar ook ditmaal is er gebruik gemaakt van de OpenCV-bibliotheek. De detector is in twee delen opgedeeld, de initialisatiefase en de loopfase. Aangezien er in deze casus de te tracken objecten gekend zijn, kan de initiatie voor elk object apart een traject starten en ook een specifieke bibliotheek aanmaken. In de tweede fase, de loopfase, gaat het programma telkens opnieuw proberen om de objecten te detecteren.

Een groot verschil tussen gevonden codes op het internet en deze codes is dat voor elke loop de detector een "cropwindow" aanmaakt, dit venster is een rechthoek waarin men de volgende detectie verwacht in terug te vinden. Hiervoor is de functie `CalculateCropwindow()` geschreven om ervoor te zorgen dat de cropwindow niet buiten de afbeelding valt. De breedte van dit venster is een vaste waarde die de gebruiker kan instellen, maar is vermenigvuldigd met het aantal keer dat de marker niet gedetecteerd is. Deze cropwindow in combinatie met het maken van een bibliotheek voor elke marker apart, zorgen ervoor dat het programma telkens maar naar één marker zoekt in een klein venster. Hierdoor zal de Detectie sneller verlopen in het geval dat het aantal gezochte objecten minder is dan het aantal markers in de bibliotheek. Ook is de cropwindow voor elk object (marker) opnieuw berekend in elke loop. De correctie van de cropwindow per loop is in Figuur 73 geïllustreerd.



Figuur 73: Corrigeren van de cropwindow in elke loop met (a) de initiatiestap en (b) na één stap.

4.2.3.2 Uitdaging 2: Hoe tracken (volgen) van elk onderdeel?

Voor het volgen/tracken van de objecten (markers) is er gebruik gemaakt van de Kalman-filter. Deze filter is een klasse van OpenCV, en daarom ook gemakkelijk om in de code te integreren. De ontworpen tracker voor deze casus heeft drie functies, de initialisatie, de tracker en de voorspeller. Met de eerste functie, `initTrack()`, zal de tracker initialiseren, dit door voor elk

object een tracker en alle nodige matrices aan te maken. Met de tweede functie, `track()`, zal een object gevolgd worden door de Kalman-filter telkens te laten updaten en de gevonden positie te laten corrigeren. Deze functie zal iedere keer bij het detecteren van een marker runnen. In het geval dat er geen marker is gedetecteerd in de gegeven cropwindow, zal functie drie in plaats van functie twee runnen. De derde en laatste functie, `predictKF()`, zal enkel de volgende locatie voorspellen door de vorige voorspelling als vorige positie te gebruiken. Hierdoor zal vervolgens ook de cropwindow opschuiven in die richting en omdat er geen detectie is, groeit deze recht evenredig groeien met het aantal gefaalde detecties.

4.2.3.3 Uitdaging 3: Hoe de identiteit voor elk onderdeel behouden (data-associatie)?

Aangezien elk object automatisch een eigen identiteit krijgt door de ArUco-markers, is data-associatie niet mogelijk. Dit komt doordat men de dubbel gedetecteerde markers eruit filtert, de functie `detectMarkers()` staat in de literatuurstudie verder uitgelegd bij het onderdeel ArUco-code. het is echter zeer eenvoudig om dit probleem op te lossen, namelijk door bij de desbetreffende functie voor het detecteren van de markers in stap drie commentaar te zetten. In Figuur 74 is te zien hoe dit gedaan wordt, maar het heeft ook als gevolg dat deze functie op andere plaatsen, waar de functie `detectMarkers()` in gebruik is, anders gaat werken.

```
/// STEP 1.a Detect marker candidates :: using AprilTag
if(_params->cornerRefinementMethod == CORNER_REFINE_APRILTAG)
    _apriltag(grey, _params, candidates, contours);

/// STEP 1.b Detect marker candidates :: traditional way
else
    _detectCandidates(grey, candidates, contours, _params);

/// STEP 2: Check candidate codification (identify markers)
_identifyCandidates(grey, candidates, contours, _dictionary, candidates, ids, _params,
                    _rejectedImgPoints);

/// STEP 3: Filter detected markers;
//_filterDetectedMarkers(candidates, ids, contours);

// copy to output arrays
_copyVector2Output(candidates, _corners);
Mat(ids).copyTo(_ids);

/// STEP 4: Corner refinement :: use corner subpix
```

Figuur 74: Aanpassing source code van OpenCV

Voor de data-associatie voor de verschillende objecten te associëren aan de verschillende trajecten is het Hongaars algoritme gekozen. De waardes voor associatie zijn door middel van dichtste buur (NN) berekend. Helaas werkt deze aanpak de aanpak van de cropwindows tegen en geeft dit het volgende probleem.

Probleem: bij de aanpak van de cropwindows zijn de objecten enkel gekend in elk venster apart, maar voor het Hongaars algoritme is het nodig om alle objecten te kennen voor het volledig venster.

Oplossing: een detector maken waarin alle objecten gedetecteerd worden voor het volledig frame. Dit is gedaan in `detector2.cpp`, en maakt dus gebruik van één algemene bibliotheek en zal geen cropwindows maken.

Zoals eerder vermeld zal het maken van de kostmatrix voor het Hongaars algoritme gebeuren door NN, maar dit is niet honderd procent correct. Aangezien er met verschillende markers gewerkt wordt, kan hier ook het appearance model worden toegepast. Dit verschijningsmodel is niet op basis van de verschijning van het object maar de ID van de gevonden marker. Dus wanneer de ID van een gevonden object niet overeenkomt met die van een traject zal in de kostmatrix een hoog getal (1000 op dit moment, is manueel ingesteld) worden ingegeven. Dit weerspiegelt een zeer kleine kans dat de gevonden marker bij het traject hoort dat door een ander ID getypeerd is. In het geval dat de ID's wel overeenkomen wordt NN toegepast, met dezelfde redenering als hierboven, namelijk hoe hoger het getal hoe lager de kans dat dit object deel uitmaakt van het traject.

4.2.3.4 Extra: uitschrijven van de pose en positie van de markers voor elke detectie.

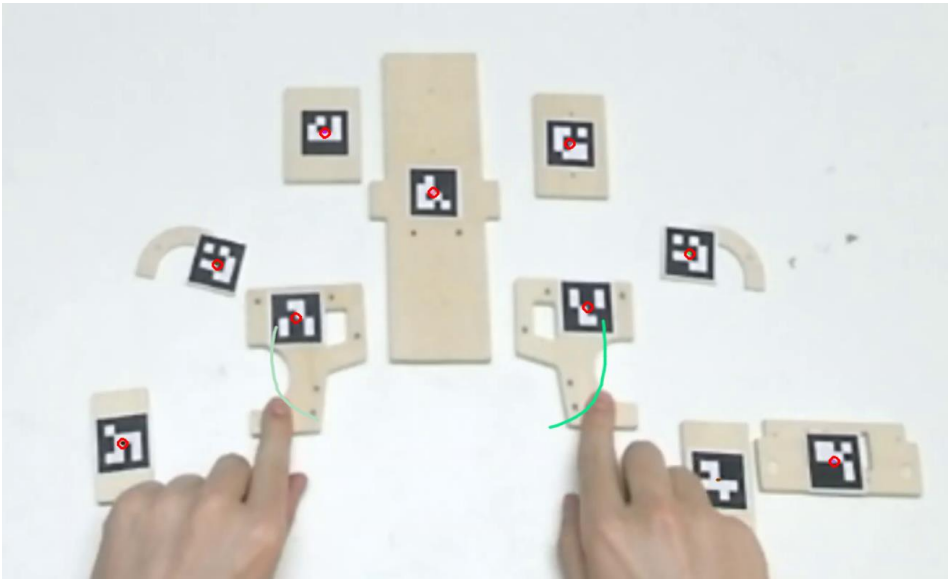
Ook hier zal het programma net als bij casus één de code van de ArUco-module gebruiken, nog specifieker de functie `estimatePoseSingleMarkers()`. Ook zullen analoog met 3.3 aanpassingen gemaakt worden om het uit te schrijven naar een CSV-file en de hoeken om te zetten naar Euler. Dit wordt enkel gedaan wanneer de `boolean estimatePose = True` staat. Dit geeft uiteraard bij het rechtstreeks lezen van de beelden via de camera (Kinect) een beduidende vertraging aan het volledige systeem. Daarom is ook hier aangeraden om de videobeelden op te nemen en vervolgens pas te verwerken.

4.2.4 Resultaten en discussie

Om het volledige programma uit te testen, is er een videofragment (zie Figuur 75) gemaakt van een persoon die een speelgoedwagentje monteert. Op elk onderdeel van het speelgoedwagentje zijn er markers geplaatst en op identieke onderdelen dezelfde markers. Dit zodat de data-associatie niet enkel zich kon focussen op de verschijning (ID marker) van de objecten, maar dat ook het bewegingsmodel een rol speelde voor de associatie. Ter verduidelijking van Figuur 75, is er op de afbeelding een rode cirkel getekend wanneer de marker gedetecteerd is.

Voor dit specifiek filmpje verwisselde het trackingsprogramma maar één keer de markers van trajecten. Dit kwam omdat de data-associatie enkel is afgesteld met behulp van de NN-techniek, hierdoor is enkel de afstand tussen twee frames van belang voor de kostmatrix. Dit is niet heel nauwkeurig want de snelheid van de objecten waren niet in rekening gebracht, noch de reeds afgelegde trajecten. Dit valt aan te passen door de snelheid en richting van de objecten een rol te laten meespelen op de afstand, dit door middel van de Mahalanobis afstand

te gebruiken. Ook door het traject mee als associatiecriteria te gebruiken is het programma accurater.



Figuur 75: Verwerking met trackingprogramma van de assemblage van een speelgoedauto

Hierbij is het duidelijk dat het tracken van de objecten en de data-associatie ervan werken, maar niet van hoge kwaliteit is. Een grote factor voor de lage kwaliteit is dat de detectie zeer onbetrouwbaar is. Zo waren objecten in beweging moeilijk tot zelfs niet detecteerbaar waardoor de Kalman-filter aan de nodige informatie ontbrak om tijdig het bewegingsmodel te updaten. Het foute bewegingsmodel had vervolgens een negatief effect op de data-associatie aangezien hierbij de voorspelde locaties voor niet-gedetecteerde markers niet accuraat zijn.

4.2.5 Conclusie

Aangezien deze casus een benadering biedt voor het probleem bij tracking van onderdelen, is het in die zin gelukt. Het knelpunt van deze casus is net de detectie waarvoor de benadering is gebeurd. Door in plaats van de markers een andere betere detectiemethode te gebruiken, gaan als gevolg hiervan de tracker en de data-associatie van de objecten accurater verlopen. Aangezien detectie de eerste stap van het trackingsproces is, is de kwaliteit hiervan essentieel. Want des te beter de detectie, des te meer data er voor de volgende stappen ter beschikking is.

Ook door het afgelegde traject mee in rekening te nemen voor de associatie van objecten aan trajecten zal de data-associatie beter verlopen, want NN is een simpele en snelle techniek om een kostmatrix te verkrijgen. Het Hongaars algoritme werkt, maar de kwaliteit hiervan is volledig afhankelijk van de opgestelde kostmatrix, en de snelheid is afhankelijk van de grootte van die kostmatrix.

4.3 Detectie en tracking bij een mobiele robot

Ook deze casus dient volledig als ondersteunend onderzoek voor een doctoraatonderzoek dat lopende is op het onderzoekscentrum ACRO. In tegenstelling tot de vorige casus zal deze casus niet beschikken oven resultaten en discussie. Dit omdat er voor deze casus op het moment van deze masterproef geen financiële middelen voorzien waren om het volledig uit te werken. In de Methodiek is er wel een voorstel voor de aanpak uitgeschreven.

4.3.1 Situering

Het doctoraatsonderzoek van Peter Aerts specificeert zich op SLAM (Simultaneous Localisation and Mapping) en pakt het probleem aan van het opbouwen van een kaart in een ongekende ruimte zonder de locatie van de robot te kennen. Aangezien dit een belangrijk onderzoeksgebied in de robotica is zijn er al verscheiden SLAM-algoritmes ontwikkeld maar deze gaan ervan uit dat de wereld statisch is. Hiertegenover staat het onderzoeksgebied rond lifelong SLAM, hierin wordt omgegaan met een niet-statische omgeving waarin er bewegende entiteiten aanwezig zijn. Daarom streeft men er in dit vakgebied naar om het proces te optimaliseren door te anticiperen op veranderingen in de omgeving. Hiervoor wordt in het eerste geval de omgeving gesegmenteerd om zo aan de hand van de informatie de omgeving te labelen. Hierdoor is het mogelijk om eigenschappen toe te wijzen aan entiteiten die uit de omgeving zijn gelabeld en hiermee hun bewegingsgedrag te modeleren. Vervolgens wordt er een kaart opgebouwd die rekening houdt met de bewegingsmodellen van de objecten Daardoor kan de mobiele robot anticiperen op deze bewegingsmodellen en botsingsvrij navigeren in de omgeving.

Voor deze casus komt padplanning en navigatie niet aan bod, enkel het segmenteren van de omgeving en het tracken van de objecten zodat hiervan bewegingsmodellen opgesteld kunnen worden. Om voor de lezer een beter beeld te krijgen op het begrip mobiele robot is er in Figuur 76 een Kobuki Turtlebot 2 afgebeeld, dit is een kleinschalige mobiele robot.

Figuur 76: Kobuki Turtlebot 2 [63]

4.3.2 Probleemstelling

Het tracken van objecten in een bewegende omgeving in combinatie met het gebruik van een bewegende camera is zeer complex. Dit omdat het bewegingsmodel van objecten niet rechtstreeks uit het beeldmateriaal berekenbaar zijn. Het bewegingsmodel kan enkel opgesteld worden door de positie en pose van de robot te kennen en vervolgens die van de objecten ten opzichte van de robot. Ook is het nodig om de positie van de objecten niet enkel in 2D te achterhalen maar ook in 3D.

Voor deze casus is er gekozen om het trackingsprobleem te versimpelen, daarom is het probleem opgedeeld in vijf deelproblemen.

- Detecteren van objecten;
- Classificeren van de objecten;
- Bepalen van de positie (3D) van de objecten ten opzichte van de robot;
- Toepassen van data-associatie op de objecten en trajecten;
- Opstellen van de bewegingsmodellen.

Elk van deze problemen moeten opgelost zijn om aan tracking voor levenslang SLAM te voldoen. Er is helaas gebleken dat deze masterproef geen oplossing kan bieden voor alle vijf de problemen omdat ze niet binnen het werkveld van de masterproef vallen. Bovenop dit algemeen trackingsprobleem zal het doctoraatsonderzoek ook nog lokalisatie en het in kaart brengen van de omgeving uitvoeren.

4.3.3 Methodiek

Vanwege de complexiteit van het probleem is het vinden van één perfecte oplossing niet evident, zeker aangezien deze masterproef niet gekend is met de volledige kennis van het lifelong SLAM onderzoeksgebied. Daarom probeert deze casus één of meerdere oplossingen of aanpakken voor te stellen voor elk van de vijf bovenvermelde deelproblemen. Ook is het belangrijk om te weten dat de mobiele robot mogelijk een extra feature heeft, Zijnde de lasserscanner. Hiermee is het mogelijk om de omgeving rondom de robot te vast te leggen in een 3D puntenwolk.

4.3.3.1. Detecteren van objecten

Zoals hierboven vermeld staat is het mogelijk dat mobielen robots beschikken over een laserscanner. Deze extra feature geeft extra informatie die een camera niet heeft, zijnde de diepte (3D beelden), en hiermee het mogelijk om objecten van elkaar te onderscheiden op basis van hun 3D grenzen. Bij de 3D grenzen zijn dit de diepteverschillen die de grenzen declareren in tegenstelling tot de 2D beelden waar intensiteitsverschillen de grenzen declareren. Een andere optie is om stereovisie te gebruiken om de diepte te achterhalen, maar hiervoor moeten

uiteraard twee beelden verwerkt worden. Op dit moment is het verwerken van één beeld al een zeer intensief proces, en door stereovisie zijn er nog geen objecten gedetecteerd maar enkel een diepte map opgesteld. Dit geeft dus drie opties: enkel een laser scanner, gebruik van stereovisie (twee camera's) of de combinatie van camera en laser scanner. De derde optie geeft uiteraard het probleem dat de camerabeelden en laser scannerbeelden op elkaar afgesteld moeten worden.

Voor detectiemethodes gebaseerd op 3D beelden kan deze masterproef geen voorstellen doen aangezien dit ook buiten het onderzoeksveld van de masterproef valt. Maar voor de detectie aan de hand van camerabeelden zijn er voldoende voorbeelden uitgelegd in de literatuurstudie. Zoals de opensource software OpenCV die veel technieken heeft voor objectdetectie, en ook neurale netwerken (worden later pas besproken).

4.3.3.2. Classificeren van de objecten

Voor classificatie aan de hand van enkel de camerabeelden stelt deze masterproef voor om een neuraal netwerk te gebruiken. Ondertussen bestaan er al veel verschillende neurale netwerken en is het onderzoek hierrond nog zeer actief, in de literatuurstudie staan er verschillende voorbeelden van detectie en classificatie zoals de YOLO-detector. Deze detectors/netwerken zijn uiteraard veel te uitgebreid aangezien ze ook de objecten detecteren aan de hand van de features, terwijl er enkel voor dit deelprobleem een classificator nodig is. Daarom is er de optie om wel of niet de 3D beelden te gebruiken en een detector te gebruiken voor detectie en classificatie. Daarnaast is het ook mogelijk om de detectie en classificatie apart te houden.

Voor classificatie van objecten is het mogelijk om dit te doen aan de hand van enkel 3D beelden maar hierover heeft deze masterproef geen kennis. Ondanks dit bestaan er ondertussen wel methodes zoals [64].

Een derde mogelijkheid is camerabeeldinformatie en lasser scannerinformatie te combineren en hiermee een neuraal netwerk op te stellen die zowel detectie als classificatie doet. Dit betekend een adaptatie van de bestaande neurale netwerken.

4.3.3.3. Bepalen van de positie van de objecten ten opzichte van de robot

In het geval dat de objecten gedetecteerd zijn, is voor het vinden van de positie geen complex algoritme nodig. Want de diepte (die noodzakelijk is) en de positie ten opzichte van de robot zijn al gekend voor elk punt toebehorend tot het object individueel. Van deze toebehorende punten neemt men dan een soort gemiddelde die de positie van het object voorstelt.

4.3.3.4. Toepassen van data-associatie op de objecten en trajecten

Voor data-associatie zijn er drie voorstellen om het deelprobleem op te lossen:

- Multi hypothese theorie in combinatie met het Viterbi algoritme
- Multi hypothese theorie in combinatie met de GRASP-methode [51]
- Hongaars algoritme

Vooraleer een van deze drie algoritmes toepasbaar is moeten er eerst waarschijnlijkheden opgesteld worden. Hiervoor zal het verschijningsmodel in combinatie met het bewegingsmodel van de objecten gebruikt een geschikte keuze zijn. In het algemeen zullen alle drie de voorstellen associatie kunnen toepassen maar de snelheid is wel verschillend. Zo is het Hongaars algoritme de snelste optie aangezien deze enkel een kostmatrix oplost. De MHT + GRASP aanpak is trager en intensiever want het MHT-algoritme houdt veel meer informatie bij over de vorige frames wat het Hongaars algoritme niet doet. De GRASP-methode is op zijn beurt sneller dan het Viterbi algoritme want deze heeft een maximale ingestelde looptijd en neemt dan de beste gevonden oplossing. Terwijl het Viterbi algoritme voor elk traject doorheen de MHT-diagram de "kortste" weg zoekt dit zijnde ook de meest verwachtbare weg. De kwaliteit van de data-associatie is zowel afhankelijk van de gekozen methode als van de kwaliteit van de waarschijnlijkheden. Dus des te beter het bewegingsmodel, des te beter de waarschijnlijkheden, des te beter de data-associatie is.

4.3.3.5. Opstellen van de bewegingsmodellen

Voor de bewegingsmodellen is het mogelijk om de Kalman-filter te gebruiken, maar de Kalman-filter zal enkel een goeie voorspelling kunnen maken voor objecten die een lineaire beweging ondernemen. Dus voor fietsers, auto's en andere vervoersmiddelen zal de Kalman-filter een accurate benadering doen, maar voor voetgangers, honden en andere niet-lineair bewegende objecten niet. Daarom is het beter om de Extended Kalman-filter te gebruiken of het opstellen van de bewegingsmodellen voor elk type object individueel.

4.3.4 Conclusie

Aangezien het doel van deze masterproef enkel onderzoek gericht is, zijn er geen financiële en materiële middelen voorzien om deze casus uit te werken. Daarom is het enkel mogelijk om voorstellen voor aanpak uit te schrijven. Daarom is het ook niet mogelijk om een uitspraak te doen over de kwaliteit van de oplossingen, maar deze methodes zijn wel al in andere papers besproken en geëvalueerd. Ook bevat de literatuurstudie informatie over de voorstellen van deze casus.

5 Conclusie

Uit de literatuurstudie is gebleken dat detectie een zeer computationeel intensief proces is. Daarom is een benadering uitgevoerd door middel van markerdetectie, vervolgens is de keuze gemaakt om de markerdetectie van OpenCV te gebruiken. Hiervan is dan ook een nauwkeurigheidstudie uitgevoerd. Het resultaat hiervan is dat het ChArUco-paneel de nauwkeurigste is gevolgd door het ArUco-paneel, de Diamantmarker en als minst nauwkeurige de ArUco-markers.

In casus 1 is er gebruik gemaakt van de Diamantmarker om de intrinsieke parameters van een gidsrobot te schatten. Dit werd gerealiseerd door de pose en positie van de marker te berekenen aan de hand van de Diamantmarker. Voor casus 2 is een volledig trackingsproces opgebouwd met als detector een benadering met behulp van de ArUco-markers, als tracker een Kalman-filter en voor data-associatie het Hongaars algoritme met de NN-techniek (Nearest Neighbour) voor het opstellen van de kostmatrix. Voor de laatste casus is het enkel mogelijk om voorstellen te doen aangezien het segmenteren van de omgeving nood heeft aan een niet-marker gebaseerde detector. Als tracker voor deze casus is de Extended Kalman-filter of het opbouwen van individuele bewegingsmodellen voor elk type object aangeraden. Voor de data-associatie zijn drie voorstellen gedaan : multi hypothese theorie in combinatie met het Viterbi algoritme, multi hypothese theorie in combinatie met de GRASP-methode, Hongaars algoritme.

Toekomstig werk

Aangezien deze masterproef als onderzoek dient voor twee doctoraatsonderzoeken, zijn deze de logische toekomstige werken. Maar toekomstige masterproeven in het vakgebied computervisie hebben ook baat aan de uitgebreide literatuurstudie en de uitgebreidere uitleg rond markerdetectie.

Bronnenlijst

[1] "ACRO – Acro." [Online]. Available: https://iiw.kuleuven.be/onderzoek/acro. [Accessed: 18-Feb-2019].

[2] R. T. Collins, "Introduction to Data Association [cursus]," *Penn State University: CSE Department*, pp. 993–1003, 2011.

[3] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I-511-I–518, 2001.

[4] OpenCV, "Introduction to Support Vector Machines — OpenCV 2.4.13.7 documentation," *Doxygen*, 2014. [Online]. Available: https://docs.opencv.org/2.4.13.7/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html. [Accessed: 11-Apr-2019].

[5] V. N. Vapnik, *Statistical learning theory NewYork*. New York: Wiley, 1998.

[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning Data Mining, Inference, and Prediction*, 2nd ed. New York: Springer, 2004.

[7] M. S. B. Maind and M. P. Wankar, "Research Paper on Basic of Artificial Neural Network," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 2, no. 1, pp. 96–100, 2014.

[8] Y. Lecun, L. Eon Bottou, Y. Bengio, and P. H. Abstract|, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[10] R. Girshick, "Fast R-CNN," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, pp. 1440–1448, 2015.

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[12] M. Abadi *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," *Google Research*, 2016. [Online]. Available: https://arxiv.org/abs/1603.04467. [Accessed: 15-Feb-2019].

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 29, pp. 779–788, 2016.

[14] F. C. Monteiro and A. Campilho, "Region and graph-based motion segmentation," in

Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2008, vol. 5112 LNCS, pp. 609–618.

[15] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," in *Proceedings of Imaging Understanding Workshop*, 1981, pp. 121–130.

[16] I. Culjak, D. Abram, T. Pribanic, H. Dzapo, and M. Cifrek, "A brief introduction to OpenCV," *MIPRO, 2012 Proceedings of the 35th International Convention*, pp. 1725–1730, 2012.

[17] OpenCV Team, "OpenCV: Face Detection using Haar Cascades," *doxygen*, 2016. [Online]. Available: https://docs.opencv.org/3.4.3/d7/d8b/tutorial_py_face_detection.html. [Accessed: 11-Feb-2019].

[18] J. Sánchez, N. Monzón, and A. Salgado, "An Analysis and Implementation of the Harris Corner Detector," *Image Processing On Line*, vol. 8, pp. 305–328, 2018.

[19] J. Shi and C. Tomasi, "Good Features to Track," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[20] OpenCV, "Shi-Tomasi Corner Detector & Good Features to Track — OpenCV 3.0.0-dev documentation," *Doxygen*, 2014. [Online]. Available: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_shi_tomasi/py_shi_tomasi.html#shi-tomasi. [Accessed: 11-Apr-2019].

[21] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," in *Computer Vision – ECCV 2006 Lecture Notes in Computer Science*, 2006, pp. 430–443.

[22] OpenCV, "Introduction to SIFT (Scale-Invariant Feature Transform)," *Doxygen*, 2014. [Online]. Available: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_sift_intro/py_sift_intro.html#sift-intro. [Accessed: 10-Apr-2019].

[23] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60(2), pp. 91–110, 2004.

[24] OpenCV, "Introduction to SURF (Speeded-Up Robust Features) — OpenCV 3.0.0-dev documentation," *Doxygen*, 2014. [Online]. Available: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html. [Accessed: 10-Apr-2019].

[25] H. Bay, A. Ess, T. Tuytelaars, and L. Vangool, "Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 404–417, 2006.

[26] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," *Computer Vision – ECCV 2010 Lecture Notes in Computer Science*, vol. 5, pp. 778–792, 2010.

[27] OpenCV, "OpenCV: BRIEF (Binary Robust Independent Elementary Features)," *Doxygen*, 2014. [Online]. Available: https://docs.opencv.org/3.4.3/dc/d7d/tutorial_py_brief.html. [Accessed: 10-Apr-2019].

[28] E. Rublee, V. Rabaud, and K. Konolige, "ORB : an efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision*, 2011, pp. 2564–2571.

[29] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[30] W. B. Barbara Frank, Cyrill Stachniss, Giorgio Grisetti, Kai Arras, "[SLIDES] Robotics 2 : Camera Calibration," *Camera*.

[31] A. Babinec, L. Jurišica, P. Hubinský, and F. Duchoň, "Visual localization of mobile robot using artificial markers," in *Procedia Engineering*, 2014, vol. 96, pp. 1–9.

[32] OpenCV, "OpenCV: Detection of ArUco-markers," *Doxygen*, 2019. [Online]. Available: https://docs.opencv.org/3.4/db/da9/tutorial_aruco_board_detection.html. [Accessed: 25-Feb-2019].

[33] OpenCV, "Detection of ChArUco Corners," *Doxygen*, 2017. [Online]. Available: https://docs.opencv.org/4.0.1/df/d4a/tutorial_charuco_detection.html. [Accessed: 25-Feb-2019].

[34] OpenCV, "OpenCV: Detection of Diamantmarkers," *Doxygen*, 2019. [Online]. Available: https://docs.opencv.org/master/d5/d07/tutorial_charuco_diamond_detection.html. [Accessed: 25-Feb-2019].

[35] A. Bakker and K. Gravemeijer, "Statistische minitools voor de ontwikkeling van het begrip verdeling," *Onderwijs Research Dagen*. 2001.

[36] E. Demeester, "Autonomous mobile robots locomotion, kinematics, perception, localisation, navigation [cursus]," *Diepenbeek: Gezamenlijke opleiding Industriële Ingenieurswetenschappen UHasselt & KU Leuven*, 2017.

[37] R. Faragher, "Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation [Lecture Notes]," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 128–132, 2012.

[38] H. Grabner, M. Grabner, and H. Bischof, "Real-Time Tracking via On-line Boosting," *Proceedings of British Machine Vision Conference (BMVC)*, vol. 1, no. 1, pp. 47–56, 2006.

[39] Z. Wang, S. Yoon, S. J. Xie, Y. Lu, and D. S. Park, "Visual tracking with semi-supervised online weighted multiple instance learning," *Visual Computer*, vol. 32, no. 3, pp. 307–320, 2016.

[40] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[41] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-Based Object Tracking," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003, vol. 25, no. 5, pp. 564–577.

[42] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual Object Tracking using Adaptive Correlation Filters," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2544–2550, 2010.

[43] A. Lukežič, T. Vojíř, L. Čehovin Zajc, J. Matas, and M. Kristan, "Discriminative Correlation Filter Tracker with Channel and Spatial Reliability," *International Journal of Computer Vision*, vol. 126, no. 7, pp. 671–688, 2018.

[44] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: Automatic detection of tracking failures," in *Proceedings - International Conference on Pattern Recognition*, 2010, pp. 2756–2759.

[45] D. Held, S. Thrun, and S. Savarese, "Learning to Track at 100 FPS with Deep," *Computer Vision – ECCV 2016 Lecture Notes in Computer Science*, pp. 749–765, 2016.

[46] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

[47] J. M. Kim, Chanho; Li, Fuxin; Ciptadi, Arridhana; Rehg, "Multiple Hypothesis Tracking Revisisted," *Iccv*, vol. 19, no. 1, 2015.

[48] B. F. J. Manly, *Multivariate statistical methods : a primer*. Chapman and Hall, 1986.

[49] "Euclidean distance and Mahalanobis distance. | Download Scientific Diagram." [Online]. Available: https://www.researchgate.net/figure/Euclidean-distance-and-Mahalanobis-distance_fig2_222399694. [Accessed: 29-Nov-2018].

[50] J. P. M. Silva and K. A. Sakallah, "GRASP—A New Search Algorithm for Satisfiability," *The Best of ICCAD*, pp. 73–89, 2003.

[51] X. Ren, Z. Huang, D. Liu, and J. Wu, "Multiple object video tracking using GRASP-MHT," *Information Fusion (FUSION)*, pp. 330–337, 2012.

[52] G. D. Forney Jr, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[53] A. Azim and O. Aycard, "Multiple pedestrian tracking using viterbi data association," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2010, pp. 706–711.

[54] H. W. Kuhn, "The Hungarian method for the assignment problem Cited by me," *Naval research logistics quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955.

[55] O. Javed, K. Shafique, and M. Shah, "Appearance Modeling for Tracking in Multiple Non-Overlapping Cameras," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR05)*, 2005, vol. 2, pp. 26–33.

[56] OpenCV, "ArUco Marker Detection," *Doxygen*, 2016. [Online]. Available: https://docs.opencv.org/3.2.0/d9/d6a/group__aruco.html#ga061ee5b694d30fa2258dd4f

13dc98129.

[57] J. Ferrão, P. Dias, and A. J. R. Neves, "Detection of ArUco-markers Using the Quadrilateral Sum Conjuncture," in *Image Analysis and Recognition*, 2018, pp. 363–369.

[58] Windows development center, "GetTickCount function (sysinfoapi.h) | Microsoft Docs," *Microsoft*. [Online]. Available: https://docs.microsoft.com/en-us/windows/desktop/api/sysinfoapi/nf-sysinfoapi-gettickcount. [Accessed: 27-Mar-2019].

[59] EMIS - VITO, "Definities en terminologie," *WAC, Compendium voor analyse van water*, pp. 1–4, 2010.

[60] D. R. L. Koomans and ing. W. Bouwmeester, "Nauwkeurigheid van de metingen van de Wegenscanners," 2016.

[61] MVTec Software GmbH, "Machine Vision in World Coordinates," 2003.

[62] Z. Tang, R. Grompone von Gioi, P. Monasse, and J.-M. Morel, "High-precision camera distortion measurements with a 'calibration harp,'" *Journal of the Optical Society of America A*, vol. 29, no. 10, p. 2134, 2012.

[63] CLEARPATH ROBOTICS, "TURTLEBOT 2 PERSONAL ROBOT, USER MANUEL," 2013.

[64] W. Luo and R. Urtasun, "Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3569–3577.

Bijlagen

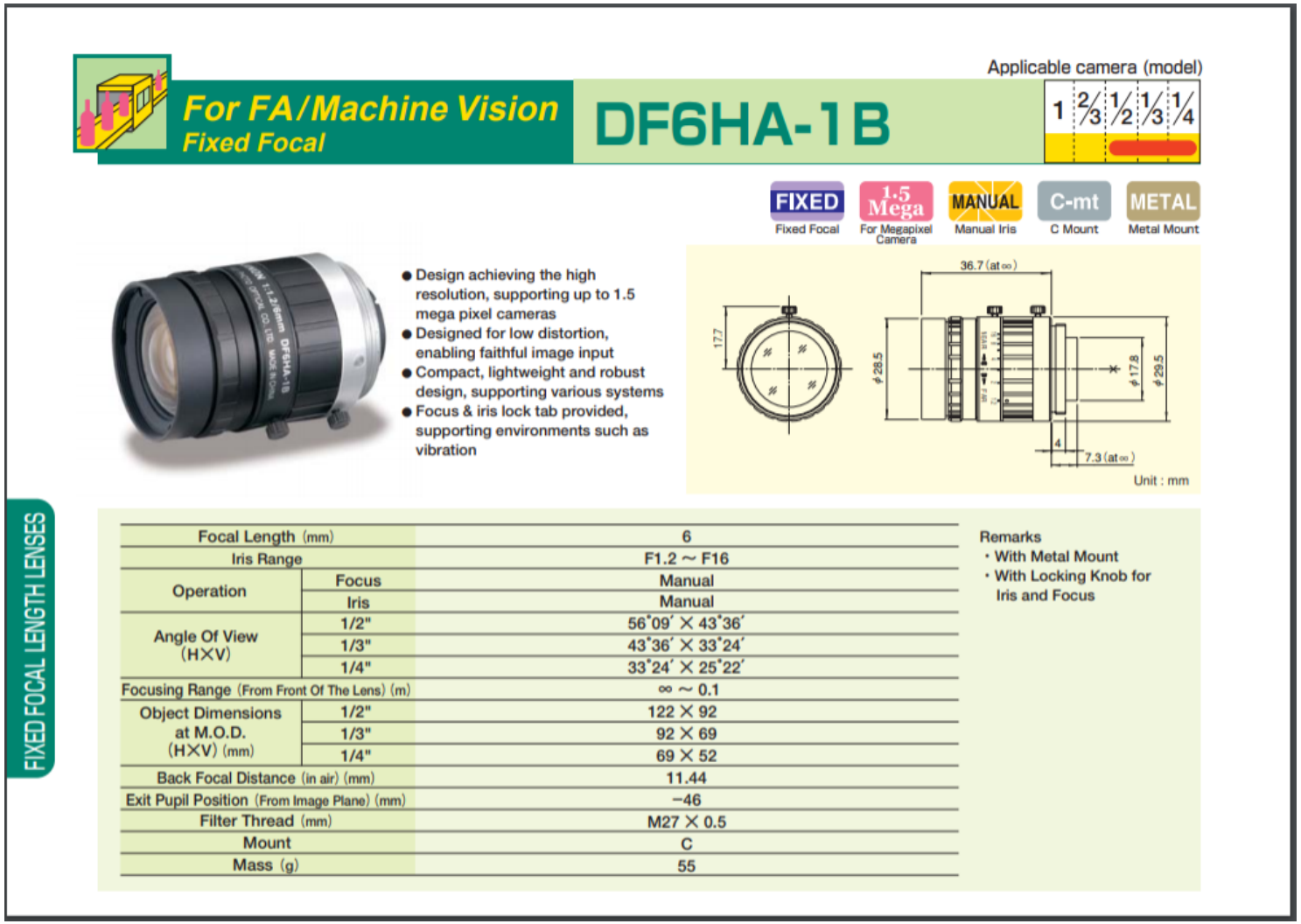
Lijst van tabellen

Lijst van figuren

```
1.  void detectMarkers(InputArray _image, const Ptr<Dictionary> &_dictionary, OutputArra
    yOfArrays _corners,
2.                     OutputArray _ids, const Ptr<DetectorParameters> &_params,
3.                     OutputArrayOfArrays _rejectedImgPoints, InputArrayOfArrays camMat
    rix, InputArrayOfArrays distCoeff) {
4.
5.      CV_Assert(!_image.empty());
6.
7.      Mat grey;
8.      _convertToGrey(_image.getMat(), grey);
9.
10.     /// STEP 1: Detect marker candidates
11.     vector< vector< Point2f > > candidates;
12.     vector< vector< Point > > contours;
13.     vector< int > ids;
14.
15.     /// STEP 1.a Detect marker candidates :: using AprilTag
16.     if(_params->cornerRefinementMethod == CORNER_REFINE_APRILTAG)
17.         _apriltag(grey, _params, candidates, contours);
18.
19.     /// STEP 1.b Detect marker candidates :: traditional way
20.     else
21.         _detectCandidates(grey, candidates, contours, _params);
22.
23.     /// STEP 2: Check candidate codification (identify markers)
24.     _identifyCandidates(grey, candidates, contours, _dictionary, candidates, ids, _p
    arams,
25.                         _rejectedImgPoints);
26.
27.     /// STEP 3: Filter detected markers;
28.     _filterDetectedMarkers(candidates, ids, contours);
29.
30.     // copy to output arrays
31.     _copyVector2Output(candidates, _corners);
32.     Mat(ids).copyTo(_ids);
33.
34.     /// STEP 4: Corner refinement :: use corner subpix
35.     if( _params->cornerRefinementMethod == CORNER_REFINE_SUBPIX ) {
36.         CV_Assert(_params->cornerRefinementWinSize > 0 && _params-
    >cornerRefinementMaxIterations > 0 &&
37.                   _params->cornerRefinementMinAccuracy > 0);
38.
39.         //// do corner refinement for each of the detected markers
40.         // for (unsigned int i = 0; i < _corners.cols(); i++) {
41.         //    cornerSubPix(grey, _corners.getMat(i),
42.         //                 Size(params.cornerRefinementWinSize, params.cornerRefinem
    entWinSize),
43.         //                 Size(-1, -
    1), TermCriteria(TermCriteria::MAX_ITER | TermCriteria::EPS,
44.         //                                                  params.cornerRefinementMaxIter
    ations,
45.         //                                                  params.cornerRefinementMinAccu
    racy));
46.         //}
47.
48.         // this is the parallel call for the previous commented loop (result is equi
    valent)
49.         parallel_for_(Range(0, _corners.cols()),
50.                       MarkerSubpixelParallel(&grey, _corners, _params));
51.     }
52.
53.     /// STEP 4, Optional : Corner refinement :: use contour container
54.     if( _params->cornerRefinementMethod == CORNER_REFINE_CONTOUR){
```

```
55.
56.         if(! _ids.empty()){
57.
58.             // do corner refinement using the contours for each detected markers
59.             parallel_for_(Range(0, _corners.cols()), MarkerContourParallel(contours,
    candidates, camMatrix.getMat(), distCoeff.getMat()));
60.
61.             // copy the corners to the output array
62.             _copyVector2Output(candidates, _corners);
63.         }
64.     }
65. }
```

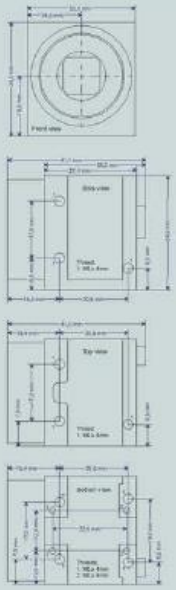
Figuur b. 1: Code van de functie detectMarker() (OpenCV)

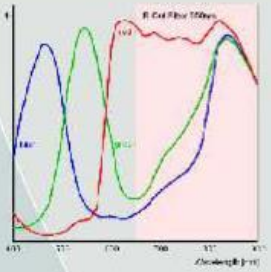


Figuur b. 2: Datasheet Fujinion 1:1.2/6mm DF6HA-1B lens

- 752 x 480 pixels (WVGA)
- High quality CMOS sensors with square pixels, Global Shutter
- Up to 87 frames/s in full frame mode, over 100 frames/s using Area of Interest (AOI)
- Digital output
- uEye SDK (compatible to the FALCON SDK) and demo programs in scope of supply
- C-Mount
- Ultra small size and less weight: 34 x 32 x 27,4 mm (HxWxD), 62g
- Camera control via USB 2.0
- Mini-B USB AND screwable Micro D-SUB connector for industrial use
- Driver and Interfaces: Windows and Linux SDK, Twain, Direct Show (WDM), ActiveX, ActivVisionTools, Common Vision Blox, HALCON and NeuroCheck
- OEM modules available

USB2.0 camera





CAMERA SPECIFICATIONS	
	UI-1220-M/C
Resolution (pixels)	752x480
M - Model (b/w)	Yes
C - Model (colour)	Bayer RGB Pattern
Chip type / Chip size	1/3 Inch/5,4 mm
Cellsize (HxV) in µm	6 x 6
Shutter	Global
Scanning System	Progressive Scan
Sensor dynamics	60 dB
Frames per second	87
Binning	hor. & vert.
Subsampling	--
AOI	hor. & vert.
Mirror /Flip	hor. & vert.
Pixel Clock Frequency (Min/Max)	5/40 MHz
Shutter speed	60µs - 1600ms
Hardware Gain	0-6dB
Hardware Trigger	async
Lens mount	C-Mount
IR filter	yes, removable
Interface	USB 2.0
Power supply	< 1 Watts
Operating temperature	0° - 50° Celsius
Regulations	CE, FCC Class B
Size (HxWxD) in mm	34 x 32 x 27,4
Weight	62 g

The uEye CMOS cameras are ultra compact cameras with USB 2.0 interface for industrial use. They are equipped with high quality sensors, industrial Design, C-Mount, a standard USB and additionally a screwable USB interfaces which offers also the trigger and digital output. Ready for professional use.
With it's USB 2.0 architecture, the cameras are ideally suited for use with PC, Notebook and Embedded Systems in image processing and factory automation.

The powerful -free of charge- uEye SDK and the yet available drivers and Interfaces for popular machine vision software allow e asy integration in applications. It's so easy!

AVAILABLE DRIVER AND INTERFACES

Windows 2000 und XP: uEye SDK, TWAIN, ActiveX, Direct Show (WDM), ActivVisionTools, Common Vision Blox, HALCON, NeuroCheck. Linux: uEye SDK

ACCESSORIES:

- USB cable
- Lenses
- USB2.0 Interfaces
- Software

The uEye cameras on the internet: www.net-usa-inc.com

Figuur b. 3: Datasheet uEye Ui 122xLE-M/C

Tabel b. 1: Data van de metingen op een constant beeld van dichtbij van Marker 34

ID	Rotatie			Translatie			Frame
	X	Y	Z	X	Y	Z	
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	1
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	2
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	3
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	4
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	5
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	6
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	7
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	8
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	9
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	10
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	11
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	12
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	13
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	14
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	15
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	16
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	17
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	18
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	19
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	20
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	21
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	22
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	23
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	24
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	25
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	26
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	27
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	28
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	29
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	30
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	31
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	32
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	33
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	34
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	35
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	36
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	37
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	38
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	39
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	40
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	41
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	42

34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	43
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	44
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	45
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	46
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	47
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	48
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	49
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	50
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	51
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	52
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	53
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	54
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	55
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	56
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	57
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	58
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	59
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	60
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	61
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	62
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	63
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	64
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	65
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	66
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	67
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	68
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	69
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	70
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	71
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	72
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	73
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	74
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	75
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	76
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	77
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	78
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	79
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	80
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	81
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	82
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	83
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	84
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	85
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	86
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	87
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	88

34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	89
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	90
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	91
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	92
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	93
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	94
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	95
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	96
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	97
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	98
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	99
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	100
34	176,983	3,67089	-89,695	0,085654	-0,06945	0,368457	101
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	102
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	103
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	104
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	105
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	106
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	107
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	108
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	109
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	110
34	177,407	3,10121	-90,0057	0,085113	-0,06884	0,365711	111
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	112
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	113
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	114
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	115
34	176,694	1,01141	-89,48	0,085228	-0,06906	0,366229	116

Tabel b. 2: Data van de metingen op een constant beeld van dichtbij van Marker 17

ID	Rotatie			Translatie			Frame
	X	Y	Z	X	Y	Z	
17	178,04	4,74472	-89,8011	-0,01666	-0,00015	0,368045	1
17	175,824	3,46956	-90,2196	-0,01658	-7,50E-05	0,367642	2
17	186,862	2,9783	-90,1051	-0,01655	-0,00017	0,369604	3
17	173,909	0,04242	-90,6917	-0,01672	-0,00029	0,368232	4
17	173,909	0,04242	-90,6917	-0,01672	-0,00029	0,368232	5
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	6
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	7
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	8
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	9
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	10
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	11
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	12

17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	13
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	14
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	15
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	16
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	17
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	18
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	19
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	20
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	21
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	22
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	23
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	24
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	25
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	26
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	27
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	28
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	29
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	30
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	31
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	32
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	33
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	34
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	35
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	36
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	37
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	38
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	39
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	40
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	41
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	42
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	43
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	44
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	45
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	46
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	47
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	48
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	49
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	50
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	51
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	52
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	53
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	54
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	55
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	56
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	57
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	58

17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	59
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	60
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	61
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	62
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	63
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	64
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	65
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	66
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	67
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	68
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	69
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	70
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	71
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	72
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	73
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	74
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	75
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	76
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	77
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	78
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	79
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	80
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	81
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	82
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	83
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	84
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	85
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	86
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	87
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	88
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	89
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	90
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	91
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	92
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	93
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	94
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	95
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	96
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	97
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	98
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	99
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	100
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	101
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	102
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	103
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	104

17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	105
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	106
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	107
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	108
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	109
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	110
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	111
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	112
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	113
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	114
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	115
17	177,345	2,69868	-89,6533	-0,01682	-0,00017	0,369378	116

Tabel b. 3: Data van de metingen op een constant beeld van dichtbij van Marker 6

ID	**Rotatie**			**Translatie**			**Frame**
	X	**Y**	**Z**	**X**	**Y**	**Z**	
6	178,201	2,19785	-89,6936	-0,08351	0,033786	0,366862	1
6	178,201	2,19785	-89,6936	-0,08351	3,38E-02	0,366862	2
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	3
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	4
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	5
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	6
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	7
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	8
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	9
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	10
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	11
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	12
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	13
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	14
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	15
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	16
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	17
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	18
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	19
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	20
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	21
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	22
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	23
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	24
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	25
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	26
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	27
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	28

6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	29
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	30
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	31
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	32
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	33
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	34
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	35
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	36
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	37
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	38
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	39
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	40
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	41
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	42
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	43
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	44
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	45
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	46
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	47
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	48
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	49
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	50
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	51
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	52
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	53
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	54
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	55
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	56
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	57
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	58
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	59
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	60
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	61
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	62
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	63
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	64
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	65
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	66
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	67
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	68
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	69
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	70
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	71
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	72
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	73
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	74

6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	75
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	76
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	77
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	78
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	79
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	80
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	81
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	82
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	83
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	84
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	85
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	86
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	87
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	88
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	89
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	90
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	91
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	92
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	93
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	94
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	95
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	96
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	97
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	98
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	99
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	100
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	101
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	102
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	103
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	104
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	105
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	106
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	107
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	108
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	109
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	110
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	111
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	112
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	113
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	114
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	115
6	177,988	0,633697	-90,0313	-0,08335	0,033795	0,366135	116

Tabel b. 4: Nauwkeurigheid marker 34 voor constant beeld van dichtbij

Marker 34						
	Rotatie			Translatie		
	X	Y	Z	X	Y	Z
Gemiddelde	176,778	1,3979	-89,542	0,085251	-0,06907	0,366357
Standaarddeviatie	0,2089	0,8847	0,1542	0,000122	0,000125	0,000629
Maximum	177,407	3,67089	-89,48	0,085654	-0,06884	0,368457
Minimum	176,694	1,01141	-90,006	0,085113	-0,06945	0,365711
Maximaal positieve afwijking	0,62911	2,27299	0,062	0,000402	0,000225	0,0021
Maximaal negatieve afwijking	0,08389	0,38649	0,4637	0,000138	0,000378	0,000646

Tabel b. 5: Nauwkeurigheid marker 17 voor constant beeld van dichtbij

Marker 17						
	Rotatie			Translatie		
	X	Y	Z	X	Y	Z
Gemiddelde	177,361	2,680	-89,681	-0,01681	-0,00018	0,369334
Standaarddeviatie	1,008	0,406	0,151	3,89E-05	1,81E-05	0,000251
Maximum	186,862	4,745	-89,653	-0,01655	-7,5E-05	0,369604
Minimum	173,909	0,042	-90,692	-0,01682	-0,00029	0,367642
Maximaal positieve afwijking	9,501	2,065	0,028	0,000264	0,000101	0,00027
Maximaal negatieve afwijking	3,452	2,637	1,010	7,65E-06	0,000116	0,001692

Tabel b. 6: Nauwkeurigheid marker 6 voor constant beeld van dichtbij

Marker 6						
	Rotatie			Translatie		
	X	Y	Z	X	Y	Z
Gemiddelde	177,992	0,661	-90,025	-0,08335	0,033794	0,366148
Standaarddeviatie	0,028	0,204	0,044	2,11E-05	1,15E-06	9,5E-05
Maximum	178,201	2,198	-89,694	-0,08335	0,033795	0,366862
Minimum	177,988	0,634	-90,031	-0,08351	0,033786	0,366135
Maximaal positieve afwijking	0,209	1,537	0,332	2,79E-06	1,52E-07	0,000714
Maximaal negatieve afwijking	0,004	0,027	0,006	0,000159	8,65E-06	1,25E-05

Tabel b. 7: Nauwkeurigheid marker 34 voor constant beeld op afstand

Marker 34						
	Rotatie			Translatie		
	X	Y	Z	X	Y	Z
Gemiddelde	181,733	3,736	180,156	-0,03813	-0,11126	0,74568
Standaarddeviatie	1,890	3,441	0,413	0,000138	0,000675	0,003857
Maximum	183,531	8,371	180,621	-0,03794	-0,11053	0,758921
Minimum	173,394	-1,868	178,852	-0,03893	-0,11342	0,740584
Maximaal positieve afwijking	1,798	4,635	0,465	0,000187	0,000734	0,013241
Maximaal negatieve afwijking	8,339	5,604	1,304	0,000797	0,002155	0,005096

Tabel b. 8: Nauwkeurigheid marker 17 voor constant beeld op afstand

Marker 17						
	Rotatie			Translatie		
	X	Y	Z	X	Y	Z
Gemiddelde	179,480	-0,442	179,979	0,029176	-0,00973	0,744014
Standaarddeviatie	3,701	1,583	0,182	4,93E-05	7,2E-05	0,002462
Maximum	191,930	-0,119	180,632	0,029232	-0,00937	0,744775
Minimum	166,586	-12,160	179,370	0,029013	-0,00987	0,735879
Maximaal positieve afwijking	12,450	0,322	0,653	5,57E-05	0,000356	0,000761
Maximaal negatieve afwijking	12,894	11,718	0,609	0,000163	0,000136	0,008135

Tabel b. 9: Nauwkeurigheid marker 6 voor constant beeld op afstand

Marker 6						
	Rotatie			Translatie		
	X	Y	Z	X	Y	Z
Gemiddelde	177,968	-2,083	179,330	0,062769	0,057331	0,741486
Standaarddeviatie	7,168	4,678	0,380	0,000157	7,8E-05	0,0014
Maximum	194,666	0,842	180,274	0,06358	0,057711	0,749788
Minimum	170,734	-12,909	178,496	0,062701	0,057042	0,739511
Maximaal positieve afwijking	16,698	2,925	0,944	0,00081	0,00038	0,008302
Maximaal negatieve afwijking	7,234	10,827	0,834	6,83E-05	0,000289	0,001975

Tabel b. 10: Gegevens Diamantmarkers met foute markerdetectie

ID	Rotatie			Translatie			Frame
	X	Y	Z		X	Y	Z
[11, 13, 14, 16]	0,0211555	3,09785	0,0711534	-0,0476925	-0,0775312	0,730138	1
[1, 3, 4, 6]	0,00483836	2,95489	-0,289978	-0,0459395	0,0620159	0,731639	1
[1, 3, 4, 6]	0,00429351	2,95689	-0,271486	-0,0460566	0,0621161	0,733016	2
[11, 13, 14, 16]	0,0283957	2,96346	0,174386	-0,0475674	-0,077358	0,729008	2
[1, 3, 4, 6]	0,00617075	2,9643	-0,2877	-0,0460249	0,0620934	0,732674	3
[11, 13, 14, 16]	0,0219979	3,01876	0,172376	-0,0476694	-0,0774763	0,729965	3
[11, 13, 14, 16]	0,0216233	3,02965	0,165559	-0,0476866	-0,0775098	0,73026	4
[1, 3, 4, 6]	0,0056483	2,9569	-0,27061	-0,0460568	0,0621078	0,732987	4
[1, 3, 4, 6]	0,00649464	2,96006	-0,274901	-0,0460582	0,0621098	0,732998	5
[11, 13, 14, 16]	0,02572	2,98607	0,1839	-0,047552	-0,0773247	0,728795	5
[6, 11, 4, 13]	0,319966	1,64407	0,812548	-0,0221181	-0,0027923	0,35038	6
[1, 3, 4, 6]	0,00605397	2,95737	-0,276394	-0,0460329	0,0620847	0,732692	7
[11, 13, 14, 16]	0,0187283	3,1108	0,157532	-0,0477839	-0,0775527	0,730768	7
[1, 3, 4, 6]	0,00538972	2,96069	-0,289594	-0,0460426	0,0621145	0,732903	8
[11, 13, 14, 16]	0,0209819	3,03514	0,176504	-0,047684	-0,0775104	0,730312	8
[1, 3, 4, 6]	0,00591913	2,95694	-0,280199	-0,0460383	0,0620946	0,732807	9
[11, 13, 14, 16]	0,0168095	3,1523	0,165115	-0,0477118	-0,0774007	0,729514	9
[1, 3, 4, 6]	0,00580978	2,95565	-0,277721	-0,0460284	0,062086	0,732685	10
[11, 13, 14, 16]	0,0214081	3,03284	0,177804	-0,0476859	-0,077506	0,730294	10
[1, 3, 4, 6]	0,00588491	2,95614	-0,273942	-0,0460422	0,0620911	0,732763	11
[11, 13, 14, 16]	0,0256102	2,99379	0,14562	-0,047642	-0,077462	0,729586	11
[1, 3, 4, 6]	0,0060115	2,95766	-0,275816	-0,0460514	0,062104	0,732921	12
[11, 13, 14, 16]	0,0224595	3,00395	0,155499	-0,0476382	-0,0774574	0,729732	12
[1, 3, 4, 6]	0,00552183	2,95403	-0,277185	-0,0460306	0,0620718	0,732578	13
[11, 13, 14, 16]	0,0174872	3,12983	0,149773	-0,047769	-0,0775251	0,730402	13
[1, 3, 4, 6]	0,00589068	2,95431	-0,278028	-0,0460157	0,0620495	0,732365	14
[11, 13, 14, 16]	0,0127885	3,15756	0,163384	-0,0477119	-0,077443	0,729547	14
[1, 3, 4, 6]	0,00553318	2,96024	-0,289215	-0,0460137	0,0620943	0,732632	15
[11, 13, 14, 16]	0,0259639	2,99165	0,146002	-0,0476297	-0,0774634	0,729492	15
[6, 11, 4, 13]	0,319993	1,64468	0,811618	-0,0221237	-0,0028165	0,350202	16
[11, 13, 14, 16]	0,0140094	3,12707	0,158659	-0,0477046	-0,0774582	0,729821	17
[1, 3, 4, 6]	0,00584928	2,96612	-0,28038	-0,0460383	0,0620537	0,732436	17
[1, 3, 4, 6]	0,00538678	2,95484	-0,272933	-0,0460538	0,0621013	0,73293	18
[11, 13, 14, 16]	0,0253746	2,99118	0,150479	-0,0476331	-0,0774542	0,729554	18
[1, 3, 4, 6]	0,00550546	2,95576	-0,274691	-0,0460374	0,0620884	0,732729	19
[11, 13, 14, 16]	0,0282774	2,98597	0,157987	-0,0476352	-0,0773899	0,729496	19
[1, 3, 4, 6]	0,00553839	2,95394	-0,273904	-0,0460423	0,0620887	0,732809	20
[11, 13, 14, 16]	0,0268583	2,98186	0,185631	-0,047573	-0,077323	0,728872	20
[1, 3, 4, 6]	0,00559554	2,96201	-0,286845	-0,0460212	0,0620909	0,732688	21
[11, 13, 14, 16]	0,0164732	3,13139	0,144255	-0,0477444	-0,077519	0,730204	21
[1, 3, 4, 6]	0,00604068	2,9609	-0,270909	-0,0460733	0,0621249	0,733158	22
[11, 13, 14, 16]	0,0208515	3,03059	0,178635	-0,0476916	-0,0774942	0,730302	22

[1, 3, 4, 6]	0,00570823	2,95368	-0,271207	-0,0460358	0,0620848	0,73267	23
[11, 13, 14, 16]	0,0257875	2,99098	0,184812	-0,0476059	-0,0773893	0,729486	23
[11, 13, 14, 16]	0,0223094	3,02479	0,175695	-0,0476654	-0,0774966	0,730163	24
[1, 3, 4, 6]	0,00585851	2,967	-0,285175	-0,046044	0,0620588	0,73247	24
[1, 3, 4, 6]	0,00604529	2,95762	-0,279076	-0,0460295	0,0620816	0,732636	25
[11, 13, 14, 16]	0,0230325	2,99747	0,157949	-0,0476108	-0,0774253	0,729429	25
[1, 3, 4, 6]	0,00646997	2,95497	-0,280083	-0,0460246	0,0620507	0,732446	26
[11, 13, 14, 16]	0,0218489	3,02305	0,1788	-0,0476738	-0,0774895	0,730187	26
[1, 3, 4, 6]	0,00579892	2,97124	-0,282498	-0,0460767	0,0621031	0,732922	27
[11, 13, 14, 16]	0,0175675	3,11897	0,138423	-0,0477378	-0,0775171	0,730355	27
[1, 3, 4, 6]	0,00578474	2,95769	-0,277571	-0,046048	0,0620817	0,732805	28
[11, 13, 14, 16]	0,0227067	3,00721	0,169084	-0,047643	-0,0774777	0,729907	28
[1, 3, 4, 6]	0,00698462	2,9559	-0,287914	-0,0460084	0,0620308	0,732231	29
[11, 13, 14, 16]	0,0139324	3,1314	0,167179	-0,0477011	-0,0774532	0,72975	29
[1, 3, 4, 6]	0,00526656	2,96051	-0,283061	-0,0460245	0,0620936	0,732762	30
[11, 13, 14, 16]	0,0162688	3,13772	0,145753	-0,0477541	-0,0774879	0,730104	30
[11, 13, 14, 16]	0,0248003	3,01302	0,153708	-0,0476888	-0,0774984	0,730079	31
[1, 3, 4, 6]	0,00555452	2,97099	-0,288416	-0,0460569	0,0620946	0,732706	31
[1, 3, 4, 6]	0,00535333	2,96239	-0,276713	-0,0460577	0,0621269	0,733098	32
[11, 13, 14, 16]	0,0179786	3,06203	0,21595	-0,0477185	-0,077506	0,73055	32
[11, 13, 14, 16]	0,0208608	3,04124	0,180028	-0,0477054	-0,0775259	0,730519	33
[1, 3, 4, 6]	0,00544074	2,96435	-0,279175	-0,0460564	0,0620796	0,732658	33
[1, 3, 4, 6]	0,00400275	2,95865	-0,275874	-0,0460454	0,062111	0,73287	34
[11, 13, 14, 16]	0,0180205	3,10631	0,143787	-0,0477487	-0,0775243	0,730463	34
[6, 11, 4, 13]	0,319893	1,64409	0,81262	-0,0221184	-0,002792	0,350364	35
[1, 3, 4, 6]	0,00625732	2,96228	-0,293496	-0,0460068	0,0620784	0,732536	36
[11, 13, 14, 16]	0,0259188	2,99021	0,148116	-0,047632	-0,0774588	0,729576	36
[1, 3, 4, 6]	0,00507833	2,96241	-0,278674	-0,046058	0,0621314	0,733129	37
[11, 13, 14, 16]	0,0258885	2,99729	0,19073	-0,0476345	-0,0773917	0,729555	37
[1, 3, 4, 6]	0,00573296	2,95535	-0,276704	-0,0460378	0,0620987	0,732794	38
[11, 13, 14, 16]	0,0280335	2,98336	0,171761	-0,0476276	-0,0773831	0,729362	38
[6, 11, 4, 13]	0,320996	1,64496	0,811957	-0,0221437	-0,0027917	0,350458	39
[6, 11, 4, 13]	0,320919	1,64551	0,811068	-0,0221332	-0,0028133	0,350207	40
[11, 13, 14, 16]	0,0220412	3,02044	0,175589	-0,047672	-0,0774937	0,730137	41
[1, 3, 4, 6]	0,00592295	2,96645	-0,285395	-0,0460367	0,062054	0,73236	41
[1, 3, 4, 6]	0,00602202	2,95822	-0,273688	-0,0460507	0,0621	0,732872	42
[11, 13, 14, 16]	0,0251891	3,00481	0,148049	-0,0476862	-0,0775066	0,730137	42
[1, 3, 4, 6]	0,00532485	2,96261	-0,285228	-0,0460338	0,0621094	0,732895	43
[11, 13, 14, 16]	0,0175475	3,1291	0,154161	-0,0477657	-0,0775131	0,730334	43
[1, 3, 4, 6]	0,00569881	2,96684	-0,281938	-0,0460518	0,0620686	0,732562	44
[11, 13, 14, 16]	0,0258978	3,00046	0,179165	-0,0476405	-0,0774172	0,729738	44
[1, 3, 4, 6]	0,00578366	2,95858	-0,274174	-0,0460527	0,0620929	0,732899	45
[11, 13, 14, 16]	0,0177253	3,12829	0,140835	-0,0477745	-0,0775132	0,730282	45
[6, 11, 4, 13]	0,319868	1,64408	0,812631	-0,0221152	-0,0027894	0,350298	46
[1, 3, 4, 6]	0,00565487	2,95819	-0,274503	-0,0460544	0,0621001	0,732936	47
[11, 13, 14, 16]	0,0172503	3,13038	0,149553	-0,0477684	-0,0775227	0,730407	47

[11, 13, 14, 16]	0,019839	3,03879	0,199809	-0,0477027	-0,07751	0,730503	48
[1, 3, 4, 6]	0,0056576	2,96687	-0,275962	-0,0460633	0,0620837	0,732734	48
[1, 3, 4, 6]	0,00507452	2,95653	-0,270858	-0,0460552	0,062112	0,732983	49
[11, 13, 14, 16]	0,0209527	3,03377	0,184751	-0,0476897	-0,0775034	0,730312	49
[1, 3, 4, 6]	0,00511938	2,95293	-0,273982	-0,0460304	0,0620781	0,732539	50
[11, 13, 14, 16]	0,0214639	3,03462	0,169211	-0,0476916	-0,0775233	0,730404	50
[1, 3, 4, 6]	0,00597565	2,95689	-0,277925	-0,046048	0,0620939	0,732781	51
[11, 13, 14, 16]	0,0219624	3,01584	0,170737	-0,0476733	-0,0774749	0,729976	51
[11, 13, 14, 16]	0,022961	2,99621	0,167712	-0,0476291	-0,0774493	0,729574	52
[1, 3, 4, 6]	0,00581518	2,95583	-0,266336	-0,0460625	0,0621059	0,732964	52
[1, 3, 4, 6]	0,00521557	2,95824	-0,283657	-0,0460201	0,0620886	0,732646	53
[11, 13, 14, 16]	0,0173983	3,12773	0,16374	-0,0477694	-0,0775242	0,730461	53
[1, 3, 4, 6]	0,00518597	2,95641	-0,268465	-0,0460479	0,0621062	0,732898	54
[11, 13, 14, 16]	0,0259806	2,99175	0,139815	-0,0476472	-0,077473	0,729747	54
[1, 3, 4, 6]	0,00579974	2,96152	-0,290597	-0,0460228	0,0620907	0,732657	55
[11, 13, 14, 16]	0,0167443	3,13189	0,153765	-0,0477646	-0,0775212	0,730369	55
[6, 11, 4, 13]	0,321007	1,64552	0,811003	-0,0221381	-0,0028177	0,350322	56
[6, 11, 4, 13]	0,320414	1,6416	0,811236	-0,0221964	-0,0027991	0,350672	57
[6, 11, 4, 13]	0,321016	1,64495	0,811937	-0,0221437	-0,0027918	0,350464	58
[6, 11, 4, 13]	0,320015	1,64408	0,812512	-0,0221209	-0,0027947	0,350444	59
[1, 3, 4, 6]	0,00619274	2,95998	-0,272646	-0,0460741	0,0621225	0,733138	60
[11, 13, 14, 16]	0,0286902	2,97928	0,158047	-0,047613	-0,0773824	0,729326	60
[6, 11, 4, 13]	0,319883	1,6441	0,812634	-0,0221196	-0,0027929	0,350385	61
[11, 13, 14, 16]	0,0263782	2,98149	0,176228	-0,0475675	-0,0773389	0,728881	62
[1, 3, 4, 6]	0,00529909	2,96785	-0,272285	-0,0460699	0,0620939	0,732835	62
[1, 3, 4, 6]	0,00510373	2,96071	-0,286548	-0,046041	0,0621087	0,732955	63
[11, 13, 14, 16]	0,0220621	3,00159	0,158514	-0,0476271	-0,0774574	0,729686	63
[11, 13, 14, 16]	0,016918	3,13124	0,156329	-0,0477816	-0,0775291	0,730575	64
[1, 3, 4, 6]	0,00565043	2,97093	-0,283915	-0,046071	0,0620958	0,7328	64
[1, 3, 4, 6]	0,00571981	2,95857	-0,276361	-0,0460584	0,0621106	0,733016	65
[11, 13, 14, 16]	0,0258905	2,99422	0,141244	-0,0476497	-0,0774807	0,729736	65
[11, 13, 14, 16]	0,0292325	2,97524	0,165136	-0,047598	-0,0773729	0,729162	66
[1, 3, 4, 6]	0,00493414	2,96577	-0,275452	-0,0460574	0,0620802	0,732695	66
[11, 13, 14, 16]	0,0205207	3,03462	0,169798	-0,0476998	-0,0775099	0,730323	67
[1, 3, 4, 6]	0,0055566	2,96648	-0,284867	-0,0460457	0,0620491	0,73238	67
[1, 3, 4, 6]	0,00518881	2,96207	-0,284839	-0,0460354	0,0621027	0,732832	68
[11, 13, 14, 16]	0,0138987	3,1284	0,158962	-0,0477226	-0,0774884	0,730019	68
[1, 3, 4, 6]	0,0056978	2,96254	-0,288439	-0,0460281	0,0620924	0,732704	69
[11, 13, 14, 16]	0,0230236	2,99693	0,158775	-0,0476077	-0,0774131	0,729287	69
[1, 3, 4, 6]	0,0049634	2,96022	-0,283789	-0,0460303	0,0621118	0,732833	70
[11, 13, 14, 16]	0,0228988	2,994	0,159452	-0,0476067	-0,0774252	0,729413	70
[1, 3, 4, 6]	0,00621753	2,9552	-0,277681	-0,0460181	0,0620632	0,732519	71
[11, 13, 14, 16]	0,0276311	2,97641	0,178421	-0,0475535	-0,0773122	0,728702	71
[1, 3, 4, 6]	0,00559278	2,9675	-0,28082	-0,0460684	0,0620745	0,732676	72
[11, 13, 14, 16]	0,0158183	3,13741	0,132441	-0,0477374	-0,0775045	0,730246	72
[1, 3, 4, 6]	0,00604771	2,95445	-0,274318	-0,0460367	0,0620731	0,73261	73

[11, 13, 14, 16]	0,022618	2,99611	0,163445	-0,0476142	-0,077424	0,729393	73
[6, 11, 4, 13]	0,321007	1,64496	0,811949	-0,0221446	-0,0027924	0,350478	74
[1, 3, 4, 6]	0,00514255	2,95965	-0,283164	-0,0460284	0,0620982	0,732753	75
[11, 13, 14, 16]	0,0260458	2,99193	0,140759	-0,0476411	-0,077463	0,729563	75
[1, 3, 4, 6]	0,00612493	2,95651	-0,279023	-0,0460155	0,062062	0,732479	76
[11, 13, 14, 16]	0,0264105	2,99387	0,133083	-0,0476437	-0,0774711	0,729676	76
[1, 3, 4, 6]	0,00440728	2,96243	-0,277696	-0,0460654	0,062145	0,733186	77
[11, 13, 14, 16]	0,0210634	3,03863	0,177108	-0,0477024	-0,0775199	0,730411	77
[1, 3, 4, 6]	0,00529354	2,96061	-0,280766	-0,0460326	0,0620992	0,732758	78
[11, 13, 14, 16]	0,0166504	3,1369	0,149575	-0,0477608	-0,0774939	0,730141	78
[1, 3, 4, 6]	0,00605825	2,95978	-0,296149	-0,0459977	0,0620683	0,732384	79
[11, 13, 14, 16]	0,0271686	2,97719	0,179703	-0,0475644	-0,0773233	0,728789	79
[1, 3, 4, 6]	0,00643723	2,95975	-0,273164	-0,0460557	0,0621038	0,732946	80
[11, 13, 14, 16]	0,0273646	2,99754	0,160398	-0,0476636	-0,0774348	0,729879	80
[1, 3, 4, 6]	0,00557179	2,95478	-0,273708	-0,0460314	0,0620814	0,732652	81
[11, 13, 14, 16]	0,0251833	2,98374	0,179558	-0,0475673	-0,0773354	0,728953	81
[1, 3, 4, 6]	0,00665339	2,96009	-0,278711	-0,046041	0,0620892	0,732764	82
[11, 13, 14, 16]	0,0255589	2,99378	0,14684	-0,047638	-0,0774617	0,7296	82
[1, 3, 4, 6]	0,00499736	2,95961	-0,27995	-0,0460369	0,0621094	0,732857	83
[11, 13, 14, 16]	0,0168095	3,13426	0,146284	-0,0477635	-0,0775351	0,730448	83
[1, 3, 4, 6]	0,00543071	2,95574	-0,275267	-0,0460447	0,0621014	0,732884	84
[11, 13, 14, 16]	0,0217453	3,01748	0,17239	-0,0476622	-0,0775022	0,730105	84
[1, 3, 4, 6]	0,00528366	2,95461	-0,271562	-0,0460409	0,0620842	0,732722	85
[11, 13, 14, 16]	0,0269602	2,98996	0,176134	-0,047593	-0,0773728	0,729288	85
[11, 13, 14, 16]	0,0199355	3,03159	0,195151	-0,0476992	-0,0775356	0,730669	86
[1, 3, 4, 6]	0,0057107	2,96641	-0,280057	-0,0460703	0,0621048	0,732876	86
[1, 3, 4, 6]	0,00516408	2,9602	-0,281998	-0,0460239	0,062099	0,732765	87
[11, 13, 14, 16]	0,0262403	2,99307	0,13554	-0,0476504	-0,0774874	0,729811	87
[11, 13, 14, 16]	0,0256974	2,99123	0,149171	-0,047646	-0,0774706	0,729675	88
[1, 3, 4, 6]	0,00561113	2,96589	-0,280405	-0,0460512	0,0620691	0,732577	88
[11, 13, 14, 16]	0,0206205	3,04941	0,168017	-0,0477054	-0,0775091	0,730274	89
[1, 3, 4, 6]	0,00530858	2,96823	-0,27831	-0,0460701	0,0620993	0,732814	89
[1, 3, 4, 6]	0,00542831	2,95943	-0,285073	-0,0460154	0,062084	0,732577	90
[11, 13, 14, 16]	0,021035	3,03052	0,177922	-0,0476842	-0,0775122	0,730324	90
[1, 3, 4, 6]	0,00402472	2,95802	-0,274692	-0,0460462	0,0621236	0,732961	91
[11, 13, 14, 16]	0,025661	2,99054	0,146981	-0,0476458	-0,0774814	0,729696	91
[1, 3, 4, 6]	0,00546808	2,95577	-0,269635	-0,0460597	0,0621003	0,73297	92
[11, 13, 14, 16]	0,0230875	3,01236	0,166415	-0,0476421	-0,0774689	0,729858	92
[1, 3, 4, 6]	0,00506906	2,958	-0,274713	-0,0460531	0,0621127	0,732908	93
[11, 13, 14, 16]	0,0231277	2,99273	0,16153	-0,0476176	-0,0774295	0,729519	93
[1, 3, 4, 6]	0,00652286	2,96092	-0,27716	-0,0460559	0,0621018	0,732934	94
[11, 13, 14, 16]	0,0276918	2,98324	0,161768	-0,0476341	-0,0774094	0,729582	94
[1, 3, 4, 6]	0,00592904	2,95568	-0,267935	-0,0460408	0,0620886	0,732776	95
[11, 13, 14, 16]	0,0251093	3,00237	0,147945	-0,0476638	-0,0774697	0,72977	95
[1, 3, 4, 6]	0,00557553	2,95622	-0,275498	-0,0460363	0,0620821	0,73273	96
[11, 13, 14, 16]	0,0200569	3,04673	0,190561	-0,0477096	-0,0775199	0,730499	96

[1, 3, 4, 6]	0,00559807	2,96546	-0,272773	-0,0460742	0,0620934	0,732826	97
[11, 13, 14, 16]	0,027183	2,98674	0,181648	-0,0475944	-0,0773363	0,729082	97
[1, 3, 4, 6]	0,00585934	2,95641	-0,276516	-0,0460454	0,0620904	0,732786	98
[11, 13, 14, 16]	0,0246198	3,00646	0,195874	-0,0476285	-0,077393	0,729735	98
[11, 13, 14, 16]	0,0172228	3,12643	0,151	-0,0477771	-0,0775497	0,730646	99
[1, 3, 4, 6]	0,0050923	2,96662	-0,282429	-0,0460716	0,0620794	0,732633	99
[1, 3, 4, 6]	0,00700964	2,95878	-0,283773	-0,0460165	0,0620614	0,73246	100
[11, 13, 14, 16]	0,0234397	3,01972	0,172584	-0,0476784	-0,077487	0,730038	100
[11, 13, 14, 16]	0,0215672	3,02905	0,177642	-0,0476916	-0,0775087	0,730323	101
[1, 3, 4, 6]	0,00541655	2,96803	-0,281229	-0,0460631	0,0620832	0,732722	101
[11, 13, 14, 16]	0,021247	3,03423	0,178649	-0,0476939	-0,0775207	0,730372	102
[1, 3, 4, 6]	0,00560779	2,96819	-0,279405	-0,0460666	0,0620868	0,732785	102
[1, 3, 4, 6]	0,00570729	2,95643	-0,274239	-0,0460444	0,0620945	0,732778	103
[11, 13, 14, 16]	0,0167574	3,11528	0,149564	-0,0477465	-0,0775096	0,73041	103
[1, 3, 4, 6]	0,00615691	2,96318	-0,292534	-0,0460311	0,062103	0,732752	104
[11, 13, 14, 16]	0,0163346	3,1425	0,15293	-0,0477535	-0,0775001	0,730189	104
[1, 3, 4, 6]	0,00516256	2,96091	-0,282642	-0,0460281	0,062085	0,732606	105
[11, 13, 14, 16]	0,0220934	3,00234	0,166411	-0,0476193	-0,0774118	0,72935	105
[11, 13, 14, 16]	0,0248688	3,02115	0,142875	-0,0476942	-0,0775065	0,730062	106
[1, 3, 4, 6]	0,00576479	2,96807	-0,279044	-0,0460681	0,062083	0,732702	106
[1, 3, 4, 6]	0,00594928	2,96048	-0,272035	-0,046078	0,0621454	0,733244	107
[11, 13, 14, 16]	0,0262262	2,98775	0,138769	-0,0476314	-0,0774565	0,729589	107
[6, 11, 4, 13]	0,319938	1,64409	0,812583	-0,0221199	-0,0027935	0,350405	108
[1, 3, 4, 6]	0,00482051	2,95936	-0,285114	-0,0460246	0,0620938	0,732694	109
[11, 13, 14, 16]	0,0225295	2,998	0,162526	-0,0476225	-0,0774468	0,729647	109
[1, 3, 4, 6]	0,00548162	2,95554	-0,272545	-0,0460497	0,0621048	0,73286	110
[11, 13, 14, 16]	0,0172391	3,12899	0,156929	-0,0477748	-0,077529	0,73053	110

Tabel b. 11: Niet-verwerkte resultaten CSV-bestand

rotatie			translatie			Time
X	Y	Z	X	Y	Z	
-3.13725	0.0135583	0.0434106	-0.0336625	0.123974	0.462469	0.00024576
-3.13641	0.0137445	0.0418173	-0.0336454	0.123972	0.4625	0.211283
-3.13483	0.0140632	0.0434191	-0.0336088	0.123989	0.462712	0.416558
-3.13685	0.0134945	0.0374485	-0.0336723	0.123983	0.462288	0.600656
-3.13219	0.0135549	0.0286331	-0.0336955	0.124004	0.462557	0.796283
-3.13648	0.0138064	0.0519435	-0.0336433	0.123994	0.46283	0.974527
-3.13908	0.0135689	0.0470741	-0.0336697	0.123969	0.462422	1.18017
-3.13592	0.0140182	0.0461773	-0.0336398	0.12399	0.462691	1.38509
-3.13838	0.0138688	0.0453306	-0.0336161	0.124011	0.462547	1.58956
-3.14094	0.0130789	0.0489352	-0.0336995	0.123963	0.46228	1.78844
-3.13942	0.013331	0.0486133	-0.0336923	0.123976	0.46244	1.99301
-3.13766	0.01352	0.0430255	-0.0336739	0.123966	0.462403	2.18774
-3.13815	0.013352	0.0458861	-0.0336885	0.123965	0.462421	2.39194

-3.13842	0.0133053	0.0450299	-0.0336915	0.123964	0.462366	2.56859
-3.13651	0.0135486	0.037959	-0.033663	0.123987	0.462401	2.75207
-3.13625	0.0138712	0.0387521	-0.0336181	0.124002	0.46256	2.96565
-3.13578	0.014301	0.041269	-0.0336222	0.123996	0.462532	3.18493
-3.1369	0.0137543	0.0506154	-0.0336591	0.123981	0.462724	3.40711
-3.13144	0.0139421	0.0316786	-0.0336338	0.124007	0.462894	3.61688
-3.13865	0.0133127	0.0464746	-0.0336928	0.123963	0.462414	3.80622
-3.13788	0.0134839	0.0445798	-0.0336728	0.123973	0.462441	4.03307
-3.14015	0.0131625	0.0494149	-0.0336956	0.123969	0.462405	4.22795
-3.13505	0.0141392	0.0433651	-0.0336147	0.124014	0.462739	4.46738
-3.13302	0.0137518	0.0326804	-0.0336677	0.124006	0.462636	4.6807
-3.13712	0.0136156	0.036986	-0.0336613	0.123979	0.462254	4.86326
3.14799	-0.0143265	-0.0412089	-0.0336165	0.124004	0.462791	5.06739
-3.14018	0.013278	0.049023	-0.0336944	0.123963	0.462315	5.25049
-3.13887	0.0133508	0.0475642	-0.0336902	0.123971	0.462482	5.46822
-3.14069	0.0131059	0.048931	-0.0336991	0.123954	0.462284	5.692
-3.13871	0.0133784	0.0459109	-0.0336773	0.123974	0.462397	6.08593
-3.13967	0.0133939	0.0498103	-0.0336656	0.12397	0.462567	6.31238
-3.13819	0.0136831	0.0538257	-0.033674	0.123985	0.462703	6.49304
-3.13675	0.0135464	0.0414648	-0.0336651	0.123978	0.46247	6.69055
-3.13582	0.0140616	0.0473707	-0.0336372	0.123998	0.462775	6.89897
-3.13922	0.013297	0.0487847	-0.0337002	0.123969	0.462479	7.08935
-3.141	0.0131394	0.0499242	-0.0337149	0.12396	0.462304	7.2778
-3.13968	0.0132384	0.0490237	-0.0337027	0.123952	0.462411	7.47046
-3.13504	0.0141325	0.0439999	-0.0335936	0.123988	0.462778	7.67091
-3.13905	0.0133869	0.0466081	-0.0336359	0.12399	0.462515	7.84764
-3.13467	0.0139145	0.0446811	-0.0335978	0.12399	0.462856	8.05192
-3.13039	0.014821	0.0428362	-0.0335899	0.124037	0.463208	8.23635
-3.13226	0.0147169	0.0510182	-0.0336036	0.124035	0.463314	8.41583
-3.13585	0.0142212	0.0465574	-0.0335969	0.12403	0.462851	8.61128
-3.13389	0.0133763	0.0309626	-0.0337444	0.124033	0.46252	8.79888

```
/*
By downloading, copying, installing or using the software you agree to this
license. If you do not agree to this license, do not download, install,
copy or use the software.
                          License Agreement
                For Open Source Computer Vision Library
                        (3-clause BSD License)
Copyright (C) 2013, OpenCV Foundation, all rights reserved.
Third party copyrights are property of their respective owners.
Redistribution and use in source and binary forms, with or without modification,
are permitted provided that the following conditions are met:
  * Redistributions of source code must retain the above copyright notice,
    this list of conditions and the following disclaimer.
  * Redistributions in binary form must reproduce the above copyright notice,
    this list of conditions and the following disclaimer in the documentation
    and/or other materials provided with the distribution.
  * Neither the names of the copyright holders nor the names of the contributors
    may be used to endorse or promote products derived from this software
```

```
19.     without specific prior written permission.
20. This software is provided by the copyright holders and contributors "as is" and
21. any express or implied warranties, including, but not limited to, the implied
22. warranties of merchantability and fitness for a particular purpose are
23. disclaimed. In no event shall copyright holders or contributors be liable for
24. any direct, indirect, incidental, special, exemplary, or consequential damages
25. (including, but not limited to, procurement of substitute goods or services;
26. loss of use, data, or profits; or business interruption) however caused
27. and on any theory of liability, whether in contract, strict liability,
28. or tort (including negligence or otherwise) arising in any way out of
29. the use of this software, even if advised of the possibility of such damage.
30. */
31. #include <opencv2/highgui.hpp>
32. #include <opencv2/ArUco.hpp>
33. #include <iostream>
34. #include <fstream>
35. #include <iostream>
36. #include <opencv2/calib3d/calib3d.hpp>
37. # define M_PI 3.14159265358979323846  /* pi */
38.
39. using namespace std;
40. using namespace cv;
41. namespace {
42.     const char* about = "Basic marker detection";
43.     const char* keys =
44.         "{d        |   0    | dictionary: DICT_4X4_50=0, DICT_4X4_100=1, DICT_4X4_250=2,"
45.         "DICT_4X4_1000=3, DICT_5X5_50=4, DICT_5X5_100=5, DICT_5X5_250=6, DICT_5X5_1000=7, "
46.         "DICT_6X6_50=8, DICT_6X6_100=9, DICT_6X6_250=10, DICT_6X6_1000=11, DICT_7X7_50=12,"
47.         "DICT_7X7_100=13, DICT_7X7_250=14, DICT_7X7_1000=15, DICT_ARUCO_ORIGINAL = 16,"
48.         "DICT_APRILTAG_16h5=17, DICT_APRILTAG_25h9=18, DICT_APRILTAG_36h10=19, DICT_APRILTAG_36h11=20}"
49.         "{v        | wit_licht.flv   | Input from video file, if ommited, input comes from camera }"
50.         "{ci       | 1     | Camera id if input doesnt come from video (-v) }"
51.         "{c        | camera_ueye(CB)4    | Camera intrinsic parameters. Needed for camera pose }"
52.         "{l        | 0.0300    | Marker side lenght (in meters). Needed for correct scale in camera pose }"
53.         "{dp       |       | File of marker detector parameters }"
54.         "{r        |       | show rejected candidates too }"
55.         "{refine   |       | Corner refinement: CORNER_REFINE_NONE=0, CORNER_REFINE_SUBPIX=1,"
56.         "CORNER_REFINE_CONTOUR=2, CORNER_REFINE_APRILTAG=3}";
57. }
58. static bool readCameraParameters(string filename, Mat &camMatrix, Mat &distCoeffs) {
59.     FileStorage fs(filename, FileStorage::READ);
60.     if (!fs.isOpened())
61.         return false;
62.     fs["camera_matrix"] >> camMatrix;
63.     fs["distortion_coefficients"] >> distCoeffs;
64.     return true;
65. }
66. static bool readDetectorParameters(string filename, Ptr<ArUco::DetectorParameters> ¶ms) {
67.     FileStorage fs(filename, FileStorage::READ);
68.     if (!fs.isOpened())
69.         return false;
70.     fs["adaptiveThreshWinSizeMin"] >> params->adaptiveThreshWinSizeMin;
71.     fs["adaptiveThreshWinSizeMax"] >> params->adaptiveThreshWinSizeMax;
72.     fs["adaptiveThreshWinSizeStep"] >> params->adaptiveThreshWinSizeStep;
73.     fs["adaptiveThreshConstant"] >> params->adaptiveThreshConstant;
74.     fs["minMarkerPerimeterRate"] >> params->minMarkerPerimeterRate;
75.     fs["maxMarkerPerimeterRate"] >> params->maxMarkerPerimeterRate;
76.     fs["polygonalApproxAccuracyRate"] >> params->polygonalApproxAccuracyRate;
77.     fs["minCornerDistanceRate"] >> params->minCornerDistanceRate;
78.     fs["minDistanceToBorder"] >> params->minDistanceToBorder;
79.     fs["minMarkerDistanceRate"] >> params->minMarkerDistanceRate;
80.     fs["cornerRefinementMethod"] >> params->cornerRefinementMethod;
81.     fs["cornerRefinementWinSize"] >> params->cornerRefinementWinSize;
82.     fs["cornerRefinementMaxIterations"] >> params->cornerRefinementMaxIterations;
83.     fs["cornerRefinementMinAccuracy"] >> params->cornerRefinementMinAccuracy;
```

```
    fs["markerBorderBits"] >> params->markerBorderBits;
    fs["perspectiveRemovePixelPerCell"] >> params->perspectiveRemovePixelPerCell;
    fs["perspectiveRemoveIgnoredMarginPerCell"] >> params->perspectiveRemoveIgnoredMarginPerCell;
    fs["maxErroneousBitsInBorderRate"] >> params->maxErroneousBitsInBorderRate;
    fs["minOtsuStdDev"] >> params->minOtsuStdDev;
    fs["errorCorrectionRate"] >> params->errorCorrectionRate;
    return true;
}

// Calculates rotation matrix to euler angles
// The result is the same as MATLAB except the order
// of the euler angles ( x and z are swapped ).
static Vec3f rotationMatrixToEulerAngles(Mat &R)
{
    float sy = sqrt(R.at<double>(0, 0) * R.at<double>(0, 0) + R.at<double>(1, 0) * R.at<double>(1, 0));
    bool singular = sy < 1e-6; // If
    float x, y, z;
    if (!singular)
    {
        x = atan2(R.at<double>(2, 1), R.at<double>(2, 2));
        y = atan2(-R.at<double>(2, 0), sy);
        z = atan2(R.at<double>(1, 0), R.at<double>(0, 0));
    }
    else
    {
        x = atan2(-R.at<double>(1, 2), R.at<double>(1, 1));
        y = atan2(-R.at<double>(2, 0), sy);
        z = 0;
    }
    return Vec3f(x, y, z);
}

int main(int argc, char *argv[]) {
    CommandLineParser parser(argc, argv, keys);
    parser.about(about);

    if (argc < 1) {
        parser.printMessage();
        return 0;
    }

    int dictionaryId = parser.get<int>("d");
    bool showRejected = parser.has("r");
    bool estimatePose = parser.has("c");
    float markerLength = parser.get<float>("l");

    Ptr<ArUco::DetectorParameters> detectorParams = ArUco::DetectorParameters::create();
    if (parser.has("dp")) {
        bool readOk = readDetectorParameters(parser.get<string>("dp"), detectorParams);
        if (!readOk) {
            cerr << "Invalid detector parameters file" << endl;
            return 0;
        }
    }

    if (parser.has("refine")) {
        //override cornerRefinementMethod read from config file
        detectorParams->cornerRefinementMethod = parser.get<int>("refine");
    }
    std::cout << "Corner refinement method (0: None, 1: Subpixel, 2:contour, 3: AprilTag 2): " << detectorParams-
       >cornerRefinementMethod << std::endl;

    int camId = parser.get<int>("ci");
    String video;
    if (parser.has("v")) video = parser.get<String>("v");
```

```
148.
149.    if (!parser.check()) {
150.        parser.printErrors();
151.        return 0;
152.    }
153.    Ptr<ArUco::Dictionary> dictionary =ArUco::getPredefinedDictionary(ArUco::PREDEFINED_DICTIONARY_NAME(dictionaryId));
154.
155.    Mat camMatrix, distCoeffs;
156.    if (estimatePose) {
157.        bool readOk = readCameraParameters(parser.get<string>("c"), camMatrix, distCoeffs);
158.        if (!readOk) {
159.            cerr << "Invalid camera file" << endl;
160.            return 0;
161.        }
162.    }
163.    VideoCapture inputVideo;
164.    int waitTime;
165.    if (!video.empty()) {
166.        inputVideo.open(video);
167.        waitTime = 100;
168.    }
169.    else {
170.        inputVideo.open(camId);
171.        waitTime = 10;
172.    }
173.
174.    ofstream outFile;
175.    outFile.open("Data_AM_linear2.csv");
176.    outFile << "ID" << "     " << "rotatie" << "          " << "translatie" << "          " << "Time" << endl;
177.    outFile << "  " << "y:  " << "p:  " << "r   " << "x:  " << "y:  " << "z   " << endl;
178.    double start = (double)getTickCount();
179.
180.    while (inputVideo.grab()) {
181.        Mat image, imageCopy;
182.        double pretick = (double)getTickCount();                    // deze neemt de tick voor de afbeelding wordt genomen
183.        inputVideo.retrieve(image);                                 // nemen van de foto
184.        double posttick = (double)getTickCount();                   // deze neemt de tick voor de afbeelding wordt genomen
185.        double fototick = (posttick + pretick) / 2;                 // het gemiddelde tussen de pre en posttick ==> wanneer foto geno
          men is
186.        double diff = (fototick - start) / getTickFrequency();      // tijd tussen start en deze foto ==> actuele tijd
187.        flip(image, image, 1);                                      // enkel voor de uEye camera
188.        rotate(image, image, 1);                                    // enkel voor de uEye camera
189.
190.        vector< int > ids;
191.        vector< vector< Point2f > > corners, rejected;
192.        vector< Vec3d > rvecs, tvecs;
193.        Mat rotM;
194.        Vec3f ABY;
195.
196.        // detect markers and estimate pose
197.        ArUco::detectMarkers(image, dictionary, corners, ids, detectorParams, rejected);
198.        if (estimatePose && ids.size() > 0)
199.            ArUco::estimatePoseSingleMarkers(corners, markerLength, camMatrix, distCoeffs, rvecs,tvecs);
200.        // draw results
201.        image.copyTo(imageCopy);
202.        if (ids.size() > 0) {
203.            ArUco::drawDetectedMarkers(imageCopy, corners, ids);
204.
205.            if (estimatePose) {
206.                for (unsigned int i = 0; i < ids.size(); i++) {
207.                    ArUco::drawAxis(imageCopy, camMatrix, distCoeffs, rvecs[i], tvecs[i], markerLength * 0.5f);
208.                    outFile << ids[i] << "  ";
209.                    Rodrigues(rvecs[i], rotM);                  // toepassen van rodrigues om rotatiematrix te verkrijgen
210.                    ABY = rotationMatrixToEulerAngles(rotM);    // rotatiematrix naar Yaw pitch roll te rekenen
211.                    ABY = ABY * 180 / M_PI;                     // naar graden omzetten
```

```
                    outFile << ABY(0) << "  " << ABY(1) << "    " << ABY(2) << "    ";
                    outFile << tvecs[i](0) << " " << tvecs[i](1) << "   " << tvecs[i](2) << "   " << diff << endl;
                }
            }
        }
        if (showRejected && rejected.size() > 0)
            ArUco::drawDetectedMarkers(imageCopy, rejected, noArray(), Scalar(100, 0, 255));

        imshow("out", imageCopy);
        char key = (char)waitKey(10);
        if (key == 27) break;
    }
    outFile.close();
    return 0;
}
```

Figuur b. 4: Aangepaste OpenCV-code voor ArUco markerdetectie