

# **Camera tracking van door radar gedetecteerde objecten**

**Gerard van Walraven**  
0950584

**8 september 2021**

**Begeleiders Hogeschool Rotterdam**

Ornella Schavemaker-Piva  
Renee van Doorn

**Begeleider en opdrachtgever Sens2Sea**

Geert Mosterdijk

# 1 Voorwoord

Deze scriptie is geschreven voor mijn afstuderen aan de opleiding Technische Informatica op de Hogeschool Rotterdam en in opdracht van Sens2Sea B.V. Het afstudeerproject liep van februari 2021 tot en met september 2021.

Boven iedereen wil ik Geert Mosterdijk bedanken voor de kans om deze stage met alle tegenslagen van dien onder zijn begeleiding af te ronden. Toen bekend werd dat mijn originele stage niet door zou gaan kon ik diezelfde dag, met de hulp van Tieneke Muller, nog bij Sens2Sea terecht om een mogelijke opdracht te bespreken. Dankzij de kans die Geert mij heeft geboden heb ik dit project met heel veel vertrouwen kunnen afronden.

Naast Geert wil ik ook graag Claus van de Weem, Tim Leijsen en Henk van Gijn bedanken voor de begeleiding vanuit Sens2Sea B.V. Zonder hun hulp had ik veel langer vast gezeten op sommige van de problemen waar ik tijdens de realisatie van het proof of principle tegen aan liep.

Ook wil ik Marten Haaksema en Jelle Koopmans bedanken voor hun hulp bij de ontwerpen van de ophanging en het 3d-printen van de uiteindelijke ophanging. Als ik hun hulp niet had gehad was ik waarschijnlijk nog steeds bezig met 3d-printen.

Allerlaatst wil ik de docenten bedanken voor de begeleiding en feedback tijdens mijn afstuderen en de flexibiliteit toen mijn eerste stage plek niet door kon gaan.

Gerard van Walraven

Waddinxveen, 6 september 2021

## 2 Samenvatting

De zee en oppervlaktewateren zijn in Nederland van groot belang. Daarom is het belangrijk dat problemen op het water snel worden opgelost. Een cruciaal onderdeel daarvan is het op tijd herkennen van objecten in het water, op basis daarvan kan worden bepaald welke handeling er moet worden ondernomen en met welke urgentie. Is er bijvoorbeeld een persoon te water geraakt dan moet deze met redelijke spoed uit het water worden gehaald. Maar een losgeslagen tak buiten de vaarweg heeft een minder grote prioriteit.

Met de huidige systemen moeten bemanningsleden op schepen of aan wal zelf met een verrekijker op zoek gaan naar door een radar gedetecteerde objecten. Dit kan veel tijd kosten die niet altijd beschikbaar is. Daarom moet het systeem worden uitgebreid met een zelfrichtende camera. Om dit te realiseren is de volgende onderzoeksraag opgesteld: Hoe kan je op basis van radar input camera's positioneren om objecten in zee vast te leggen op grote afstanden?

Tijdens het project is vanaf nul een geheel nieuw systeem gemaakt voor het juist verwerken van radar inputs en het richten van een camera. Het systeem is daarnaast uitgebreid met een user interface om de gebruiker controle te geven over het systeem en de beelden live te kunnen volgen. Het eindresultaat is een volledig functioneel proof of principle dat gebruikt kan worden als voorbeeld van de mogelijkheden van deze technologie.

Op basis van dit proof of principle zijn nog heel erg veel uitbreidingen mogelijk. De camera en de besturing van de zoom moeten in de toekomst vervangen worden door een professionele camera. Daarnaast is het ook van belang dat er in de toekomst een stabilisatiesysteem wordt ingebouwd zodat de camera ook op een schip goed kan functioneren.

# Inhoudsopgave

<b>1 Voorwoord</b>	<b>1</b>
<b>2 Samenvatting</b>	<b>2</b>
<b>3 Begrippenlijst</b>	<b>5</b>
<b>4 Probleemstelling</b>	<b>6</b>
4.1 Context . . . . .	6
4.2 Beperkingen huidige radar . . . . .	7
<b>5 Opdrachtomschrijving</b>	<b>8</b>
5.1 Onderzoeksvragen . . . . .	8
5.2 Scope . . . . .	8
<b>6 Systeem architectuur</b>	<b>9</b>
<b>7 Keuze camera</b>	<b>10</b>
7.1 Eisen van de camera . . . . .	10
7.2 PTZ camera's . . . . .	11
7.3 Losse sensor met lens . . . . .	12
7.4 Losse Zoom camera's . . . . .	13
7.5 Camcorders . . . . .	13
7.6 Vergelijking . . . . .	14
7.7 Uiteindelijke keuze . . . . .	14
<b>8 Keuze motoren</b>	<b>14</b>
8.1 DC motor (brushed en brushless) . . . . .	15
8.2 servomotor . . . . .	15
8.3 steppermotor . . . . .	15
<b>9 Binnenhalen radar gegevens</b>	<b>15</b>
<b>10 Kijkrichting camera bepalen</b>	<b>16</b>
10.1 Berekenen pan . . . . .	16
10.2 Bereken afstand . . . . .	17
10.3 Berekenen tilt . . . . .	18
10.4 Eigen offsets . . . . .	19
<b>11 Ophanging</b>	<b>19</b>
11.1 Eigen ontwerp . . . . .	19
11.2 Ontwerp van een derde . . . . .	21
<b>12 Steppermotoren</b>	<b>22</b>
12.1 Stepper-drivers en microstepping . . . . .	22
<b>13 Voeding</b>	<b>23</b>

<b>14 Aansturing</b>	<b>24</b>
14.1 Hardware . . . . .	24
14.2 AccelStepper library . . . . .	24
14.3 Beweging op de assen . . . . .	25
14.4 Nulpunt bepalen . . . . .	25
14.5 Bepalen stappen per graad . . . . .	26
<b>15 Controller</b>	<b>27</b>
<b>16 User interface</b>	<b>28</b>
16.1 Commando's . . . . .	28
16.2 Communicatie met de controller . . . . .	29
16.3 Implementatie in de controller . . . . .	30
<b>17 Event-based besturing</b>	<b>32</b>
17.1 Message class . . . . .	32
17.2 Communicatie classes . . . . .	32
17.3 Multithreading . . . . .	33
17.4 Global handlers en modes . . . . .	34
17.5 Status . . . . .	34
17.6 Meerdere user interfaces . . . . .	34
17.7 Betere uitbreidbaarheid . . . . .	34
<b>18 Conclusie</b>	<b>35</b>
<b>19 Aanbevelingen</b>	<b>35</b>
19.1 Bijlage A: Test verslag . . . . .	38

### 3 Begrippenlijst

**bearing** De richting waarin een object zich voortbeweegt relatief aan het geografische noorden. 5

**camerasysteem** Het gehele systeem bestaande uit: de camera, het richtsysteem en de aansturing van het richtsysteem. 6–8

**radarsysteem** Een systeem dat met behulp van radio golven de geografische locatie, bearing en snelheid van een object kan bepalen. In dit geval een specifiek systeem gemaakt door Sens2Sea B.V.. 6–8, 15

## 4 Probleemstelling

De zee en oppervlakte wateren in Nederland zijn voor ons van groot economisch en recreatief belang. Daarom is het belangrijk dat problemen op het water kunnen worden voorkomen en snel worden opgelost, mocht het wel gebeuren. Huidige radarsystemen kunnen objecten op het water wel detecteren en positioneren maar herkennen is niet mogelijk. Radarsystemen kunnen namelijk op kleine schaal vrijwel geen vormen herkennen. Dit is echter wel nodig om te bepalen welke handeling er moet worden ondernomen en met welke urgentie dit moet gebeuren.

Door het gebrek aan herkenning van de radarsystemen is het nu aan de bemanning van een schip of het personeel aan wal om met een verrekijker te zoeken naar het gedetecteerde object. Dit is op enkele honderden meters nog wel haalbaar, maar op afstanden in de kilometers is dit praktisch onmogelijk. Naast dat dit heel lastig is duurt dit ook erg lang, en zoals later wordt uitgelegd is deze tijd niet altijd beschikbaar.

Om dit probleem op te lossen moeten met een camera losse beelden gemaakt kunnen worden. Om de camera op de objecten te richten is een richtsysteem nodig. De stabiliteit en precisie van dit richtsysteem hebben directe invloed op de maximale afstand waarop de camera objecten kan vastleggen, daarom is het van belang dat de systemen zo precies mogelijk zijn en van stabilisatie worden voorzien in zowel de software als hardware.

Hoewel er wel systemen bestaan, denk hierbij aan camerasytemen die worden toegepast voor het bestrijden van piraterij, is er binnen de scope van de opdrachtgever nog geen systeem dat op meerdere plekken kan worden toegepast om vanuit meerdere hoeken een duidelijk beeld van een object te geven.

Zonder het gebruik van meerdere camera's op meerdere posities kunnen objecten voor een tijdje uit het beeld van de camera verdwijnen, dit kan er voor zorgen dat de gebruiker niet altijd goed zicht heeft op het object dat wordt gevolgd. Daarnaast kunnen meerdere camera's een triangulatie meting mogelijk maken voor een hele nauwkeurige bepaling van de positie van objecten.

### 4.1 Context

Om een goed beeld te krijgen van het nut van het systeem is het goed om een voorbeeld voor te stellen. Een van de mogelijke toepassingen van het systeem is het redden van te water geraakte personen, drenkelingen. Rond de haven van Rotterdam raken met enige regelmaat mensen te water, daarbij is het essentieel dat deze personen zo snel mogelijk uit het water worden gehaald. Het zoeken naar de personen is jammer genoeg makkelijker gezegd dan gedaan, zeker als deze personen na bijvoorbeeld een klap op het water buiten bewustzijn zijn geraakt.

Het is echter wel van levensbelang dat een persoon heel snel uit het water wordt gehaald. Bij een watertemperatuur van 5 graden Celsius is, volgens schattingen van de KNRM, de overlevingskans na een uur vrijwel nul [5]. Deze tijd wordt nog korter als de persoon buiten bewustzijn is of geestelijk onwel is. Het camerasytemen zou kunnen helpen in de zoektocht en de reddingswerkers de mogelijkheid geven om zich volledig te richten op het redden van een persoon.

Naast het gebruik voor het vinden van objecten of mensen is het ook een oplossing voor het leed dat sommige dieren lijden op zee. Het heien van de fundering voor windmolens kan tot wel 260 dB aan geluid produceren [6]. Een geluid van dit niveau kan zware negatieve effecten hebben voor dieren in de buurt van de bron [10]. Het camerasystrem zou de schepen kunnen ondersteunen in het spotten van zoogdieren die aan het oppervlak moeten komen om adem te halen. Hierdoor kan het heien worden stilgelegd als het nodig is.

Tot slot is het systeem ook goed geschikt voor het herkennen van boten en schepen. Denk hierbij aan grote containerschepen tot kleine vissersbootjes of zelfs rubberen bootjes. Deze systemen zouden kunnen worden ingezet voor het automatisch identificeren van alle boten en schepen op zee of op het oppervlaktewater. Deze systemen zouden potentieel ook smokkelaars en illegale visserij kunnen opsporen.

## 4.2 Beperkingen huidige radar

Hoewel de huidige radarsystemen die gebouwd zijn door Sens2Sea B.V. wel objecten in het water kunnen detecteren en volgen is er nog geen manier om deze vast te leggen op beeld en te kunnen identificeren. De radar trackt uit zichzelf alle potentiële objecten op het wateroppervlak. De gebruiker kan uit één van de getrackte objecten kiezen om deze met de camera te volgen, deze krijgt dan een speciale markering. Het is echter lastig voor de gebruiker om het object te identificeren, de radar kan alleen aangeven op welke hoek, relatief aan het noorden, het object zich bevindt en hoe ver weg het is zoals te zien in figuur 1. De gebruiker zal dan zelf met een verrekijker of het blote oog op zoek moeten gaan naar het object om het te identificeren. Deze objecten variëren ook nog eens in grootte van een plastic fles of een kanoer tot een bruinvis die adem komt halen of een zeecontainer. Grote objecten zijn vaak relatief makkelijk te spotten voor de gebruiker, maar kleinere objecten zijn bijzonder lastig te vinden aangezien deze zich vaak achter de golven bevinden.



Figuur 1: Voorbeeld radar beelden

## 5 Opdrachtomschrijving

### 5.1 Onderzoeks vragen

Hoe kan je op basis van radar input camera's positioneren om objecten in zee vast te leggen op grote afstanden?

- Welke camera is ideaal voor gebruik in het camerasysteem?
- Hoe haal je de gegevens van de radar binnen?
- Hoe zet je de radargegevens om in nauwkeurige hoeken voor de pan en tilt?
- Hoe zijn de camera's zo nauwkeurig mogelijk te richten?
- Hoe maak je een user interface voor simpele aansturing en het tonen van camera beelden?
- Wat is de beste manier om met alle onderdelen met elkaar te laten communiceren?
- Hoe kan het systeem worden uitgebreid om met meerdere camera's te werken en mogelijk vanaf meerdere locaties?

### 5.2 Scope

De volgende onderdelen vallen wel binnen de scope van het project en worden behandeld:

- Prototype camerasysteem op land in verbinding met stationaire radar
- Keuze voor de camera
- Communicatie met het radar systeem
- Richten camerasysteem
- Met meerdere camera's kunnen werken om hetzelfde of verschillende objecten te volgen
- Communicatie met en implementatie user interface
- Documentatie van het hele systeem
- Testen camerasysteem

Buiten de scope vallen de volgende onderdelen:

- Radarsysteem en tracking op basis van radar gegevens
- Identificatie objecten door middel van software
- Zelf maken zoom aansturing van de camera

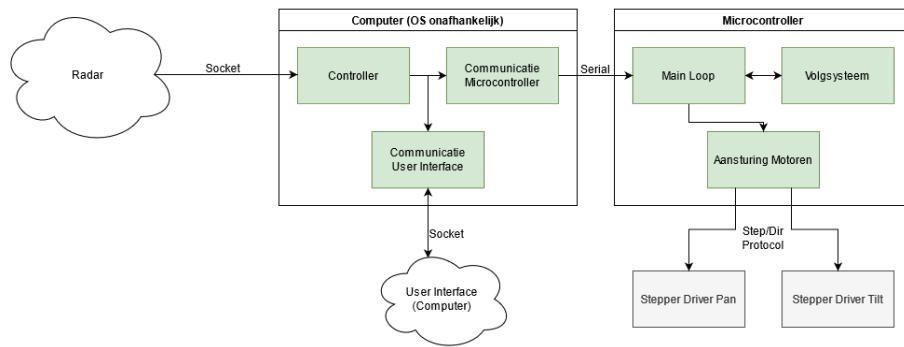
## 6 Systeem architectuur

Het te maken systeem bestaat uit 3 onderdelen zoals te zien in figuur 2: De controller die draait op een computer, het richtsysteem dat draait op een microcontroller en de user interface die ook op een computer draait.

De controller dient als het brein van het systeem, alle berekeningen vinden daar plaats en het handelt alle inputs van zowel de radar als de user interface af.

Het richtsysteem zorgt ervoor dat de commando's van de controller worden uitgevoerd door de motoren die de camera moeten bewegen. Hierbij maakt het systeem ook berekeningen zoals hoeveel afstand de motor moet afleggen om de gewenste positie te bereiken.

De user interface verbind het systeem en de gebruiker, hier is het beeld van de camera te zien en kan de gebruiker het systeem handmatig besturen of instellingen aanpassen.



Figuur 2: Diagram systeemarchitectuur

## 7 Keuze camera

Om de keuze voor de camera te maken is er gekeken naar verschillende mogelijkheden van bestuurbare camera's en losse camera's.

### 7.1 Eisen van de camera

Om te bepalen welke camera de beste is, zijn de opties vergeleken op een aantal punten.

Het eerste punt zijn de uitbreidingsmogelijkheden van de camera. De camera moet de mogelijkheid hebben om, tegen een hogere prijs, betere kwaliteit en/of grotere optische zoom te bieden. De opdrachtgever gaf aan om voor de Proof of Principle een camera goed genoeg is met 30x zoom. Daar komt bij dat de camera minimaal over een 1080p of betere resolutie moet beschikken om scherpe beelden te kunnen maken. Hogere resoluties zijn beter maar niet nodig aangezien het gemiddelde scherm geen hogere resolutie kan laten zien.

Ten tweede is de bestuurbaarheid van de camera belangrijk, omdat het mechanisch aansturen van lenzen buiten de scope van het project valt, moet dit ingebouwd zijn in de camera.

Als laatste is de prijs van de camera belangrijk, het prototype is puur bedoeld als Proof of Principle en het budget is beperkt. Daarom moet het type camera beschikken over zowel goedkope opties als duurdere varianten met een hogere kwaliteit. De camera moet minder dan 500 euro kosten, voor het Proof of Principle.

Naam	Prioriteit	Omschrijving
Uitbreidbaarheid	Hoog	De camera moet tegen betrekking goed uitbreidbaar zijn in onder andere kwaliteit en zoom
Bestuurbaarheid	Gemiddeld	De camera moet zonder te veel aanpassingen door de microcontroller worden aangestuurd op de zoom, focus en waar van toepassing ook de pan en tilt
Prijs	Hoog	De prijs van de camera moet niet te hoog liggen voor het proof of principle

## 7.2 PTZ camera's

Op de markt zijn al meerde bestaande Pan/Tilt/Zoom (PTZ) camera's te bestellen, vaak te besturen over het internet. Deze worden vaak met servo's bestuurd en zijn bedoeld voor beveiliging en TV/Livestream productie. Het aanbod van PTZ camera's is heel breed en er zijn zeker uitbreidingsmogelijkheden voor betere kwaliteit.

Omdat de PTZ camera's al beschikken over een ophanging met aansturing, is dat deel van het project al opgelost. Hierdoor is er meer tijd over voor andere mogelijke uitbreidingen van het project. Deze extra is tevens ook een nadeel aangezien het richtsysteem van de camera vrijwel niet aangepast kan worden.

Het grote nadeel van PTZ camera's is de beperkte precisie van de camera's. Alle PTZ camera's van minder dan 500 euro hebben een zoom van minder dan 30x en een beperkte precisie in Pan en Tilt. Een goed voorbeeld hiervan is de ArduCam die eerst gepland was om voor het project te gebruiken. Deze camera heeft maar een 3x zoom en is daardoor onbruikbaar op lange afstanden, daarnaast zijn de pan en tilt van de camera niet nauwkeurig door het gebruik van simpele servo's. Ook hebben deze oplossingen vaak een beperkte hoek voor de Pan. Omdat dit type camera's al over een richtsysteem beschikken komt er in dit specifieke geval een extra eis bij, namelijk dat de camera 270° kan draaien op de x-as. Op deze manier beperkt de camera keuze het zicht niet. Hoewel de bestuurbaarheid van deze camera's dus erg goed is gaat het wel ten kosten van een stukje precisie.

De kosten van PTZ camera's die wel erg precies zijn en aan een goede zoom en resolutie voldoen zijn vaak erg hoog en daarom per definitie te duur voor het Proof of Principle, de camera in figuur 3 kost 1400 euro en de prijs loopt op voor camera's met een hogere resolutie, zoom en precisie op de assen. Over het algemeen zijn PTZ camera's wel goed bruikbaar als het project een serieuzere vorm aan neemt.



Figuur 3: Logitech Rally PTZ camera

### 7.3 Losse sensor met lens

Een ander alternatief zou zijn om losse camera sensoren te gebruiken met een aparte lens, voorbeelden van beide zijn te zien in figuur 4. Een van de voordelen van dit systeem is dat zowel de lens als de sensor makkelijk vervangen kunnen worden.

Het grote voordeel van deze oplossing is het zelf kunnen samenstellen van lens en sensor. Hiermee kan in theorie voor elke situatie een aparte oplossing worden gemaakt met een apart maximaal bereik gebaseerd op de zoom van de lens, de resolutie van de camera en de precisie van het Pan/Tilt systeem. Voor de sensoren bestaan er veel goedkope en goed werkende oplossingen zoals de Raspberry Pi HQ Camera die direct werkt met een Raspberry Pi.

Deze oplossing heeft echter twee grote nadelen, de eerste is het vinden van goed werkende en passende hardware. Het vinden van goed werkende losse lenzen met een extern bestuurbare zoom is een grote uitdaging. Daarnaast moet er voor elke oplossing een losse interface worden toegepast in de code die de camera aanstuurt aangezien niet alle lenzen op dezelfde standaard werken. Daarom is de uitbreidbaarheid beperkt omdat het hele systeem vaak moet worden vervangen en er maar beperkte passende opties zijn van lens en sensor combinaties.

Het tweede nadeel is het feit dat deze manier van oplossen vrij ver buiten de scope van het project valt. Daardoor zou er veel tijd worden besteed aan het samenstellen en toepassen van deze oplossing die ook besteed had kunnen worden aan nieuwe features. Dit maakt het besturen van de zoom van deze camera's heel erg lastig.

Ook qua prijs valt dit type camera tegen, professionele lenzen zijn erg duur en de markt voor losse sensoren is niet heel breed. Voor een Proof of Principle liggen de kosten nog net te hoog voor deze combinatie.



Figuur 4: Losse camera sensor (links) en lens (rechts)

## 7.4 Losse Zoom camera's

Een andere realistische oplossing voor dit project is het gebruik van losse zoom camera's die bedoeld zijn voor makkelijke gebruik in beveiligingscamera's. Deze camera's zijn normaal bedoeld voor toepassing in de PTZ camera's zoals eerder vermeld, maar zonder het Pan/Tilt systeem er omheen. De markt is heel erg breed voor dit type camera's en er zijn heel veel alternatieven beschikbaar.

Deze camera's hebben vaak al een bestaande interface die makkelijk te gebruiken is via het Pelco-D protocol. Hiermee kan de zoom van de camera gemakkelijk worden bestuurd vanaf een extern apparaat. Het nadeel van dit type camera is dat ze vaak met composiet signalen de video uitzenden. Dit moet eerst omgezet worden voordat het bruikbaar is voor de meeste toepassingen. Dit maakt de bestuurbaarheid van dit type camera's heel goed.

Het nadeel van deze camera's is dat ze binnen Europa lastig te verkrijgen zijn. Hierdoor ligt de prijs van dit type camera erg hoog in vergelijking met de andere oplossingen.



Figuur 5: Voorbeeld losse zoom camera

## 7.5 Camcorders

Camcorders zijn camera's bedoeld voor het opnemen van video's. Ze beschikken vaak over een zoom functionaliteit en ingebouwde stabilisatie. Er bestaan veel verschillende soorten camcorders in verschillende prijscategorieën. De goedkopere beschikken over beperkte optische zoom, terwijl duurdere varianten over hele grote zoom capaciteit beschikken.

Het enige nadeel van dit type camera is dat het digitaal aansturen van de zoom niet altijd ingebouwd zit. Dit veroorzaakt wel hinder, maar is niet dramatisch aangezien de knop voor het zoomen losgehaald kan worden en door middel van bijvoorbeeld een relais aangestuurd kan worden.

Er is een grote hoeveelheid aan goedkope opties voor camcorders die binnen Nederland te koop zijn.

## 7.6 Vergelijking

Van de 3 criteria (uitbreidbaarheid, duurzaamheid en prijs) is de volgende tabel op te stellen voor alle camera soorten.

	Uitbreidbaarheid	Bestuurbaarheid	Prijs
PTZ Camera	++	++	--
Losse sensor met lens	+/-	--	-
Losse zoom camera	+	++	--
Camcorders	++	+/-	++

## 7.7 Uiteindelijke keuze

Hoewel qua uitbreidbaarheid en bestuurbaarheid de PTZ camera en losse zoom camera het beste uit de vergelijking komen is van beide de prijs te hoog voor het Proof of Principle. Daarom is gekozen voor een camcorder aangezien die relatief nog een goede keuze is en qua prijs heel gunstig is.

De specifieke camcorder die uiteindelijk is gebruikt voor het Proof of Principle is de Panasonic HC-V180 omdat deze camera beschikt aan alle minimale eisen en een goede prijs kwaliteit verhouding bied met een maximale zoom van 40x en 1080p kwaliteit.

Daarnaast heeft de camera positieve extra's waarom voor deze specifieke camcorder is gekozen. De camera beschikt over software stabilisatie en de mogelijkheid om over de HDMI port live beeld te sturen zonder enige icoontjes of statussymbolen over het beeld te laten zien.

## 8 Keuze motoren

In het richtsysteem zijn twee motoren nodig om de camera in de pan en tilt hoek te kunnen draaien. Om de keuze te kunnen maken voor het beste type motor zijn de volgende punten van belang:

Naam	Prioriteit	Omschrijving
Onbeperkt draaien	Onmisbaar	Om de resolutie van het systeem te verhogen word er gebruik gemaakt van tandwielen, daarom mag de motor geen beperkte draaihoek hebben. Er zijn namelijk meerdere rotaties nodig om het grotere tandwiel te laten bewegen.
Controleerbare snelheid	Onmisbaar	Omdat het richtsysteem als doel heeft om in 1 seconde naar een bepaald punt te bewegen moet de snelheid van de motor makkelijk aan te passen zijn.
Precisie	Hoog	Als de zoom van de camera maximaal is heeft elke kleine beweging van de camera een grote impact op het beeld. Daarom is het belangrijk dat de motor naar exact de juiste positie beweegt want elk kleine verschil kan betekenen dat het object niet op het beeld meer is.
Resolutie	Hoog	Zoals bij de precisie is het voor het beeld ook belangrijk dat de resolutie van de motor hoger is dan 100 stappen per graad.

Met deze eisen zijn de volgende motoren overwogen voor gebruik in het richtsysteem.

## 8.1 DC motor (brushed en brushless)

Hoewel DC motoren onbeperkt kunnen draaien en de snelheid te controleren is zijn ze niet geschikt door het gebrek aan precisie. De motoren kunnen namelijk geen bepaalde positie vasthouden. Er zou een extra systeem om de motor heen gebouwd moeten worden om bewegingen te corrigeren. Dat valt ver buiten de scope van het project en daarom is dit type motor niet te gebruiken [3, 4].

## 8.2 servomotor

Er bestaan twee type servomotoren, positionele en continu draaiende servomotoren. De positionele servomotor val gelijk af aangezien deze niet aan de eisen van onbeperkt draaien en controleerbare snelheid voldoet. De continu draaiende servomotor kan dat wel en de snelheid is te regelen. Het grote nadeel in dit systeem zit hem in de precisie van de motor. De motor corrigeert de positie constant door de ingebouwde dc motor heen tegen te laten draaien wanneer de motor onbedoeld beweegt. Doordat dit constant gebeurt kan er echter een soort bibberend effect ontstaan waarbij de motor steeds een beetje heen en terug beweegt. Hoewel dit maar hele kleine bewegingen zijn kan dit bij maximale zoom van de camera een heel groot effect hebben op het beeld. Daarom is ook dit type motor niet bruikbaar voor het proof of principle [2, 3, 4].

## 8.3 steppermotor

De laatst bekende motor is de steppermotor. Deze motor bestaat uit een permanente magneet met tanden en een set spoelen. Door afwisselend stroom over de spoelen te laten lopen kan de schaft met de magneten worden gedraaid door het magnetische veld van de spoelen. Hierdoor kan de motor heel precies worden gedraaid. De resolutie is door microstepping toe te passen ook te verhogen. In tegenstelling tot de servomotor houd de steppermotor zijn positie vast door middel van het magnetische veld van de spoelen. Hij maakt tijdens stilstand dus geen kleine bewegingen. Daarnaast kan de motor onbeperkt draaien en is de snelheid van de motor goed te besturen. Dit maakt de steppermotor ideaal voor gebruik in het proof of principle [2, 3, 4].

# 9 Binnenhalen radar gegevens

In het radarsysteem heeft de gebruiker een overzicht van de omgeving van de radar met daarop alle gedetecteerde objecten duidelijk aangegeven. De gebruiker kan daaruit één of meerdere objecten selecteren om te volgen. Die gegevens moeten door de camera's worden gebruikt om ook door de camera's te worden gevolgd.

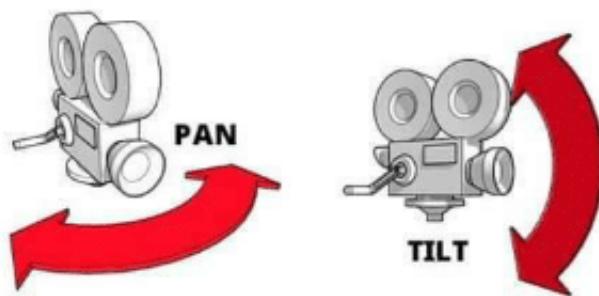
Omdat de radar in veel gevallen op een andere locatie draait als de camera's die ermee verbinden, is het nodig om via het internet de gegevens door te sturen. Omdat de functionaliteit al bestaat bij de radarsystemen wordt daarbij een websocket gebruikt. Vanuit het radarsysteem wordt een simpele string gestuurd met daarin verwerkt de geografische locatie van het object en de geschatte snelheid en koers. Het format is als volgt: "latitude,longitude,snelheid,koers".

Omdat een directe bedrade verbinding niet altijd mogelijk en moet het systeem over het internet kunnen communiceren. Om dit probleem op te lossen draait er op de computer bij de radar een simpele TCP/IP socket. Alle camera systemen kunnen hiermee verbinden, welk object elke camera moet volgen wordt door de server bij de radar bepaald. Hiervoor wordt het string format zoals hierboven beschreven gebruikt.

De string van het radarsysteem bestaat uit door komma's gescheiden waarden. Deze string word in de lokale code van de camera gesplitst en verwerkt. De doorgestuurde string bestaat uit 4 onderdelen: de breedtegraad, lengtegraad, snelheid en koers van het gevolgde object. Alle waarden kunnen worden omgezet in doubles waarna ze gebruikt kunnen worden in de berekening naar een pan en tilt hoek voor het camerasysteem.

## 10 Kijkrichting camera bepalen

Nu de coördinaten van het te volgen object bekend zijn is het tijd om deze om te zetten in een kijkhoek voor de camera. Die hoek bestaat uit twee assen. De pan of x-as die de horizontale beweging van de camera bepaald en de tilt of de z-as die de verticale beweging bepaald. Beide assen zijn duidelijk te zien in figuur 6.



Figuur 6: Pan en tilt uitgebeeld

De kijkrichting van de camera bestaat dus uit twee hoeken die berekend moeten worden. Daarvoor worden de volgende berekeningen gebruikt.

### 10.1 Berekenen pan

De pan is de horizontale hoek waarop de camera word gedraaid. Hierbij zijn geografische coördinaten relevant omdat hiermee de horizontale positie van beide objecten word aangegeven.

Om de juiste richting van de camera te bereiken moet eerst de gewenste geografische richting worden bepaald. Dit kan worden gedaan met de uitwerking hier boven. Hier wordt met behulp van twee geografische coördinaten de hoek van het tweede punt ten opzichte van het eerste punt berekend. Deze hoek zal het punt worden waar de camera naartoe draait.

Om de hoek tussen de twee punten uit te rekenen moeten de twee punten eerst van coördinaten worden omgezet naar een twee dimensionale lijn. Hiervoor moeten de twee coördinaten omgezet worden in twee punten waarvan 1 op de oorsprong en de ander een punt daarbuiten  $[9, 1]$ .

Het punt daar buiten kan worden berekend met de volgende twee formules:

$$x = \sin(\Delta\lambda) \cdot \cos(\phi_2)$$

$$y = \cos(\phi_1) \cdot \sin(\phi_2) - \sin(\phi_1) \cdot \cos(\phi_2) \cdot \cos(\Delta\lambda)$$

Daarna kan met de arctan2 functie de hoek van de lijn tussen het punt en de oorsprong berekend worden:

$$\theta = \text{atan2}(x, y)$$

Hierna kan de hoek van radialen worden omgezet naar een hoek in graden. Deze hoek is wel tussen -180 en 180 graden dus moet omgezet worden door middel van de volgende berekening:

$$\text{hoek} = \frac{\theta + 360}{360}$$

Normaal gesproken moet er nu nog rekening gehouden worden met de kromming van de aarde. De berekende hoek geeft de initiële hoek aan vanaf het eerste punt tot het tweede punt. Omdat de camera niet werkt op afstanden waarbij de kromming van de aarde relevant is, de objecten zijn dan namelijk niet te zien achter de horizon [1].

## 10.2 Bereken afstand

Om de berekening van de tilt van de camera uit te voeren moet eerst de afstand tot het object bekend zijn.

De hoek tussen twee punten op een cirkel is met de volgende vergelijking te berekenen:

$$\theta = \frac{d}{r}$$

Waarbij:  $\theta$  de hoek in radialen is,  $d$  de afstand tussen twee punten op een cirkel en  $r$  de radius van de cirkel.

Voor het berekenen van de hoek tussen twee punten wordt de haversine formule gebruikt. De formule is als volgt:

$$\text{hav}(\theta) = \text{hav}(\Delta\phi) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \text{hav}(\Delta\lambda)$$

Hierbij zijn de haversines in te vullen als:

$$\text{hav}(x) = \sin^2\left(\frac{x}{2}\right)$$

En kan de gehele functie herschreven worden naar de volgende functie:

$$a = \text{hav}(\theta) = \sin^2(\Delta\phi) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2(\Delta\lambda)$$

Hiermee is de hoek in radialen te berekenen met de volgende berekening:

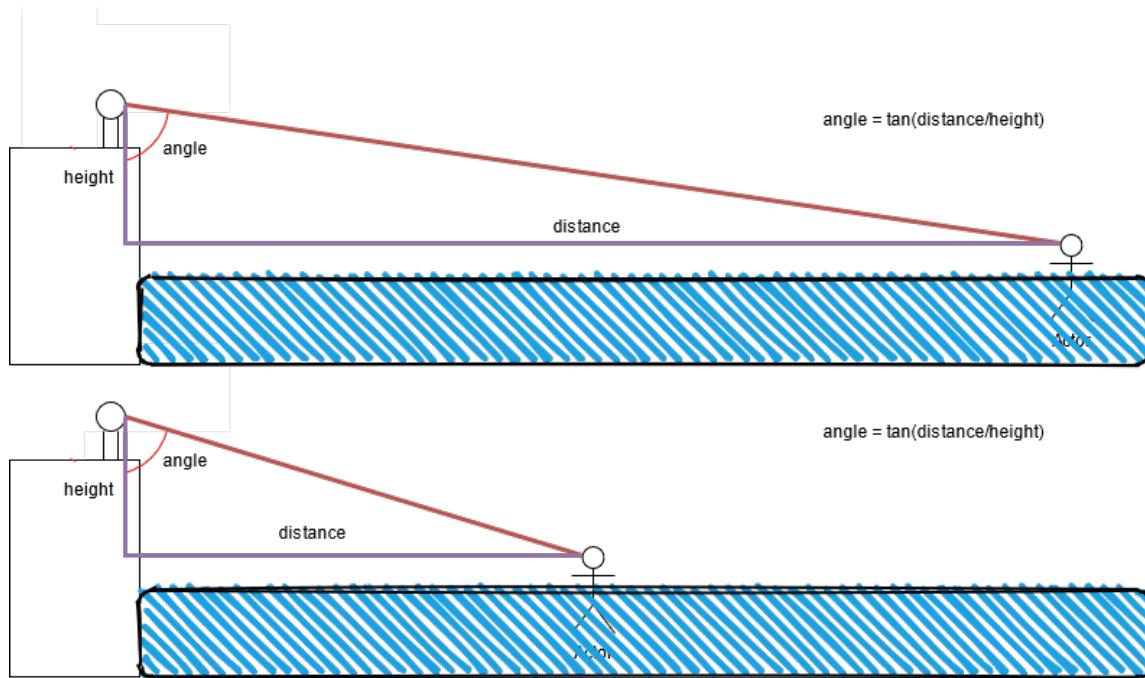
$$\theta = 2 \cdot \text{atan}2(\sqrt{a}, \sqrt{(1-a)})$$

Deze berekende hoek is de hoek tussen de twee punten op de cirkel, in dit geval de aardbol, gezien vanaf het middelpunt. Omdat de hoek en de radius van de aarde bekend zijn is hiermee nu de afstand tussen de twee punten uit te rekenen met behulp van de eerste formule [1]:

$$d = r \cdot \theta$$

### 10.3 Berekenen tilt

De tilt van de camera is de hoek die de camera maakt in de z-as, dus de verticale hoek van de camera. Deze verandert op basis van de afstand van het object zoals te zien is in figuur 7.



Figuur 7: Verschil Tilt op basis van afstand

Nu de afstand is berekend kan ook de hoek van de tilt berekend worden door middel van de volgende berekening:

$$\theta = \arctan\left(\frac{o}{a}\right)$$

Hierna hoeft deze hoek alleen nog maar omgezet te worden in graden en dan kan hij gebruikt worden door de camera om de verticale beweging mee te regelen.

## 10.4 Eigen offsets

De berekende hoeken zijn echter niet in elke situatie ideaal. De camera zal zich op basis van de berekeningen bewegen naar het middelpunt van het gedetecteerde object en direct op het zeeoppervlak waar deze is gedetecteerd. Neem het voorbeeld van een schip zoals te zien in figuur 8. De camera zal automatisch naar het midden van het schip bij het wateroppervlak richten zoals in het rode vierkant. Het is in dit geval echter relevanter om de identificatie van het schip bij de boeg in beeld te hebben.



Figuur 8: Verschil Tilt op basis van afstand

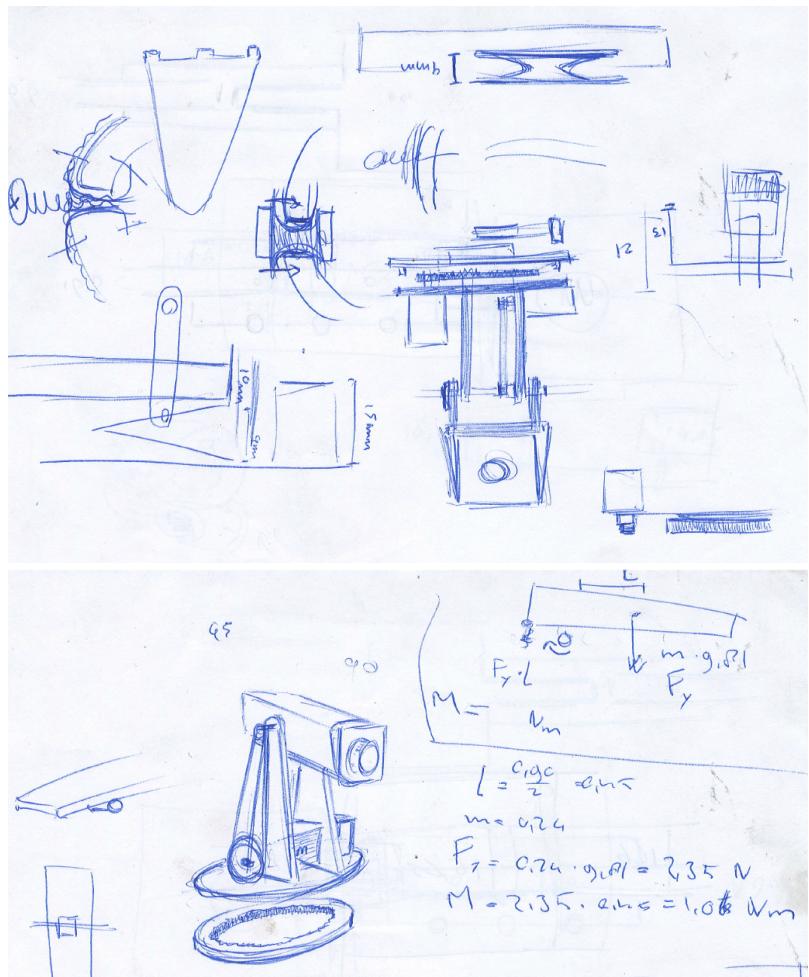
Om dit probleem op te lossen is het gebruik van een eigen offset cruciaal. De gebruiker kan op deze manier handmatig de camera bijstellen zonder dat deze stopt met het volgen van het object. De offsets van de gebruiker moeten bij elke nieuwe berekening van de pan en tilt worden meegenomen. Op deze manier heeft de gebruiker de vrijheid om de camera te richten op elk onderdeel van het object zonder de controle volledig over te moeten nemen.

De gebruiker moet door gebruik te maken van de user interface de camera in real-time kunnen bijstellen door middel van de pijltjestoetsen op het toetsenbord. Zo krijgt de gebruiker direct feedback op zijn input waardoor de offsets gemakkelijk in te stellen zijn.

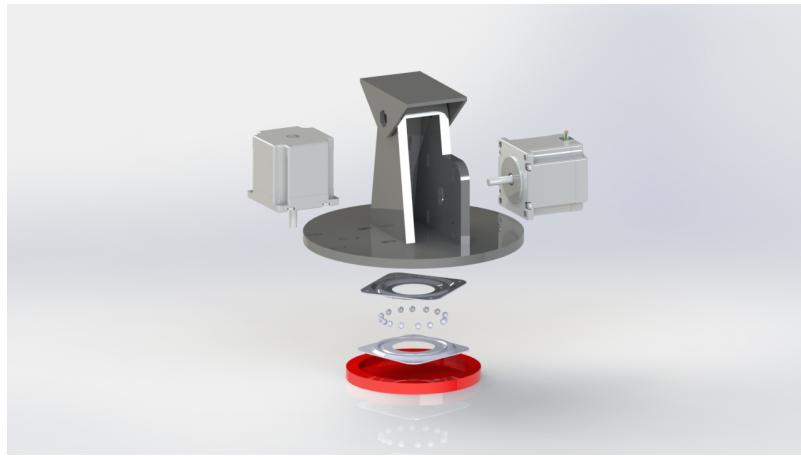
# 11 Ophanging

## 11.1 Eigen ontwerp

In de eerste fase van het ontwerp is er gewerkt aan een eigen ontwerp voor de ophanging van de camera omdat het in principe relatief simpel moest gaan worden. Het was echter al snel duidelijk dat het ontwerp niet zo simpel was gemaakt als eerst gedacht. In de figuren hieronder zijn de eerste uitwerkingen te zien in schetsen en 3d ontwerpen.



Figuur 9: Concept schetsen originele ontwerp



Figuur 10: Exploded view van eerste 3d ontwerp

## 11.2 Ontwerp van een derde

Omdat het eerste ontwerp te lastig bleek om te implementeren en omdat het project dreigde in tijdsnood te komen door het zelf maken van de ophanging is gekozen om een ontwerp van Isaac Chasteau te gebruiken. Deze relatief simpele ophanging is vrijwel volledig 3d geprint, hierdoor is hij snel te printen zonder te veel werk. Het ontwerp werkt op 360 graden op beide assen waardoor het geen beperking oplevert voor het systeem. Het design kon vrijwel volledig direct worden toegepast, alleen het tandwiel van de z-as moest worden vergroot aangezien de originele tandwiel ratio te klein was om zo precies mogelijk te richten. Een foto van de uiteindelijke ophanging is te zien in figuur 11, hierbij zijn de oranje onderdelen de aangepaste onderdelen.



Figuur 11: Ophanging op basis van het ontwerp van Chasteau

## 12 Steppermotoren

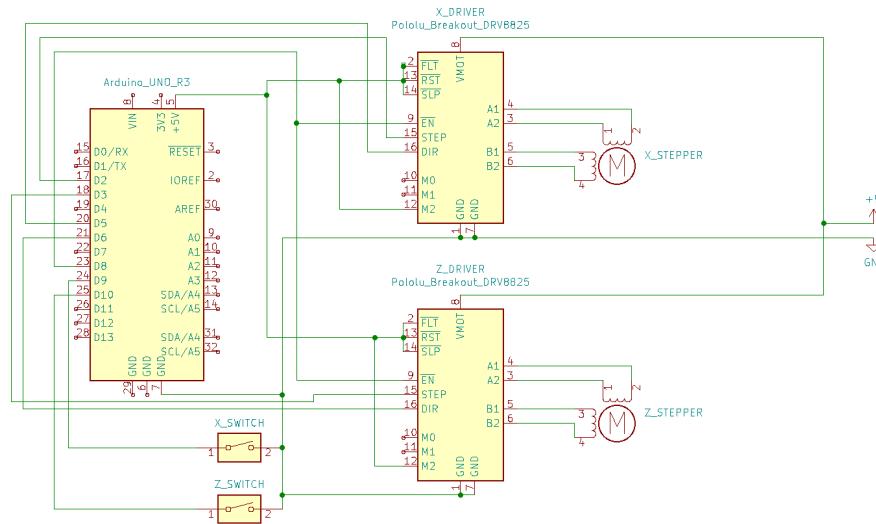
De uiteindelijke keuze voor de steppermotoren is gevallen op een 0.55 Nm motor met het NEMA-17 formaat. De keuze voor 0.55 Nm is gemaakt omdat de camera praktisch in het midden van de ophanging hangt. Daardoor is de koppel die nodig is om de camera op zijn plek te houden of te bewegen niet heel hoog.

De keuze van het NEMA-17 formaat is gemaakt omdat deze gebruikt zijn in de ophanging die is gebruikt. Zo hoefde er niet aanpassingen worden gemaakt aan het ontwerp. Daarnaast zijn deze motoren makkelijk te verkrijgen in Nederland.

### 12.1 Stepper-drivers en microstepping

Om de steppermotoren goed te laten functioneren zijn stepper-drivers nodig om de spoelen van de motor aan te sturen. In dit geval is de DRV8825 omdat deze door de leverancier van de motoren werden aangeraden als goede driver voor de motoren.

Met deze drivers kan ook microstepping worden toegepast om de motor extra resolutie te geven bij het draaien en dus preciezere bewegingen kan maken. Dit gaat op hogere snelheden wel ten kosten van de koppel van de motor. In het camerasystsem is 1/16 microstepping toegepast. Dat houdt in dat elke normale stap van de motor is onderverdeeld in kleinere stappen. De resolutie van de motor verandert dan van 200 stappen per omwenteling naar 3200 stappen per omwenteling. De drivers zijn gemakkelijk aan te sturen via het Step/Dir protocol.



Figuur 12: Diagram aansluiting steppermotoren, drivers en nulpunt schakelaars

## 13 Voeding

Om te bepalen wat voor soort voeding er nodig is is het belangrijk om eerst te bepalen welke onderdelen er zijn en wat voor spanning en stroom de onderdelen nodig hebben. Deze onderdelen zijn als volgt:

- Camera, 5V 1A
- Raspberry Pi 4B, 5V 3A
- Arduino Uno, 5V 1A
- 2 Steppermotoren, 12-24V 6A

De keuze is vooral door 2 factoren bepaald. Ten eerste het gewenste voltage van de steppermotoren, daarvoor is het belangrijk om te weten wat het effect is van een hoger voltage op de motoren. Zoals bij veel motoren heeft het voltage bij de steppermotoren eigenlijk alleen te maken met de snelheid van de motor [7]. Omdat het systeem niet snel hoeft te draaien omdat de afstand tot de objecten groot is, is dit niet heel relevant. Een lager voltage heeft daarnaast ook geen negatief effect op de houdkoppel van de motor.

Om de redenen beschreven hierboven is besloten om de motoren van 12V te voorzien. Dan is nog de vraag of het beter is om een 5V voeding te gebruiken en het voltage omhoog te brengen voor de steppermotoren of een 12V voeding te gebruiken en het voltage te verlagen naar 5V. Omdat de steppermotoren het meeste stroom gebruiken van alle componenten en omdat het efficiënter is om voltage te verlagen dan te verhogen [8], is gekozen voor een 12V voeding.



Figuur 13: Foto van de gekozen voeding

Tijdens het uitwerken bleek echter dat de spanning te hoog was om genoeg kracht te leveren om de steppermotoren met 1/16 microstepping consistent te laten bewegen. De motor bleef veel hangen en

maakte grote sprongen tussen stappen. Hierdoor werd het beeld op camera heel onduidelijk.

Omdat de kracht die de steppermotor kon leveren op deze lage snelheid voornamelijk wordt bepaald door de stroom is dat het eerste vermoeden waar onderzoek naar gedaan is. Omdat de weerstand op de spoelen vast staat was de enige manier om de stroom te verhogen door de spanning te verlagen.

Met een variabele voeding is de spanning langzaam terug geschroefd terwijl de steppermotor aan het draaien was. Vanaf 10V begon de steppermotor consistent te bewegen en bij 9V gaf de camera een stabiel beeld bij langzame beweging. Onder 9V stopte de drivers met werken dus is gekozen om 9V te gebruiken voor de steppermotoren in plaats van de eerdere 12V.

## 14 Aansturing

Om de camera te richten met behulp van de richtsystemen is een snelle en precieze aansturing nodig van de steppermotoren. De aansturing krijgt de positie over 1 seconde van het object binnen en moet hier binnen 1 seconde naartoe bewegen voordat de volgende positie binnen komt.

### 14.1 Hardware

Eerst moest er in het project een besluit genomen worden over de hardware die gebruikt zou gaan worden. De keuze is uiteindelijk op de Arduino gevallen voor het richtsysteem. De hoofdreden hiervoor was de beschikbaarheid en ervaring met Arduino's. De familie van Arduino's komen in heel veel vormen maar gebruiken allemaal dezelfde onderliggende aansturing. Daardoor is het heel makkelijk te wisselen tussen Arduino's mocht het nodig zijn. Het eerste prototype is gemaakt met een Arduino Uno aangezien die direct beschikbaar was.

De controller van het camerasysteem moet in principe op elke PC kunnen draaien. Het doel is echter dat het systeem comfortabel op een kleine single-board-computer kan draaien als een Raspberry Pi. Deze systemen kunnen in de toekomst gemakkelijk op of in de behuizing van de camera worden bevestigd.

### 14.2 AccelStepper library

Om dit voor elkaar te krijgen wordt er gebruik gemaakt van de AccelStepper library. Deze library houdt de timing van stappen bij voor de aansturing en heeft ook een ingebouwde mogelijkheid om de beweging te versnellen en te verlangzamen. Van deze functie wordt uiteindelijk geen gebruik gemaakt aangezien het timing problemen veroorzaakte.

De AccelStepper library gebruikt een maximale snelheid om te bepalen hoe hard de motor draait en probeert deze snelheid altijd zo snel mogelijk te bereiken. Dit doet de library op basis van de opgegeven maximale versnelling. In het geval van de aansturing is de maximale snelheid het aantal stappen dat nodig is om het opgegeven punt te bereiken, omdat de library werkt met stappen per seconde zal de motor de opgegeven positie dan binnen 1 seconde bereiken. Om er voor te zorgen dat de motor altijd op dezelfde snelheid beweegt is de maximale versnelling heel erg hoog, hierdoor bereikt de motor na de eerste stap al de gewenste snelheid.

### 14.3 Beweging op de assen

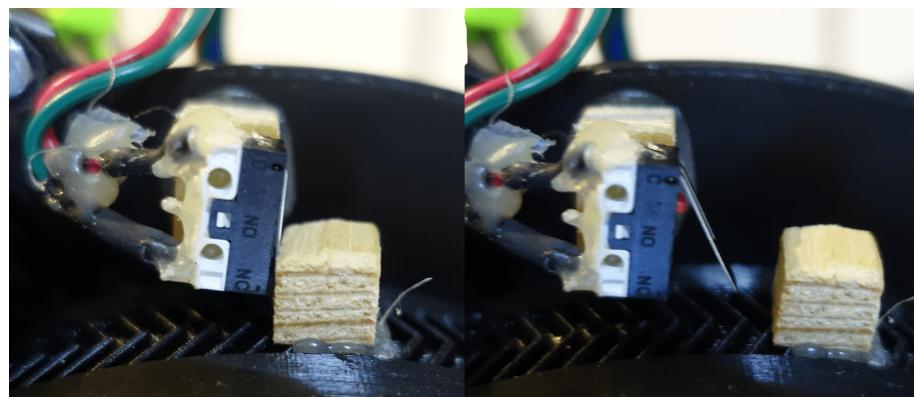
Het camerasysteem beweegt op de x- en z-as door middel van de eerder beschreven steppermotoren. Het systeem heeft een limiet op hoe ver het in een richting kan draaien totdat de draden in de knoop raken of te kort worden en de motoren tegenhouden. Daarom hebben beiden assen een duidelijk limiet waarbinnen zij kunnen draaien. Daarnaast maakt het limiet het makkelijker om het nulpunt te bepalen.

Omdat het voor de camera niet relevant is om achter zich te kijken maar wel links en recht van de camera moet de x-as 270 graden kunnen draaien. Met een nulpunt links achter bij de elektronica. Hierdoor kan de camera gemakkelijk de volledige 270 graden draaien zonder dat kabels in de knoop raken of te kort zijn.

Om alle mogelijkheden op de z-as open te houden moet deze 180 graden kunnen draaien met recht naar beneden als nulpunt van de as. Hierdoor blijven de kabels die van achteren uit de camera lopen altijd aan dezelfde kant van de ophanging en dus nooit per ongeluk in het zicht van de camera. Door 180 graden te kunnen draaien blijven alle opties wel open voor deze as aangezien hij van recht naar beneden tot recht naar boven kan kijken en dus geen dode hoek toevoegt.

### 14.4 Nulpunt bepalen

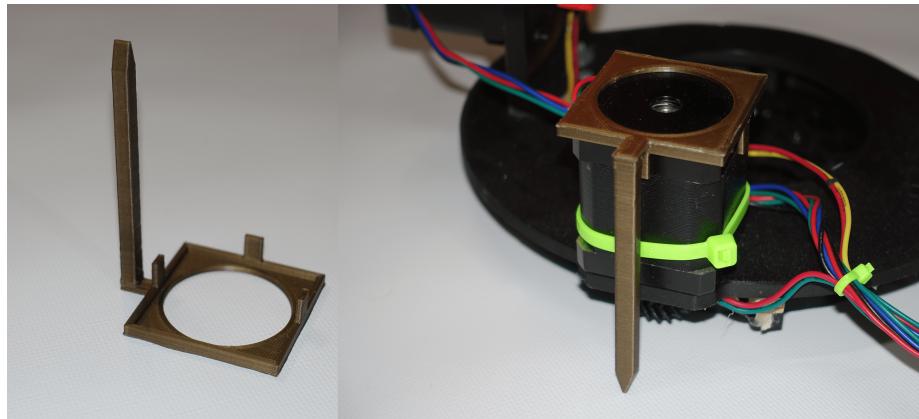
Omdat bij het opstarten de aansturing geen idee heeft op welke hoek de steppermotoren zich bevinden, aangezien ze met de hand verplaatst kunnen worden als het camerasysteem niet actief is, moet de aansturing eerst een bepaald nulpunt vinden. Dit wordt gedaan met behulp van twee schakelaars die zich op de basis van de camera bevinden. Twee uitstekende stopblokken op de tandwielen van de x- en z-as komen hiermee in aanraking als de steppermotoren in de negatieve richting bewegen.



Figuur 14: Foto's van een gesloten en open schakelaar bij het nulpunt van de x-as

De schakelaars kunnen op twee momenten worden geactiveerd: tijdens de nulpunt bepaling bij het opstarten en tijdens normale operatie als door een fout de steppermotoren te ver terug bewegen en in contact komen met de schakelaar. Zowel tijdens het opstarten als tijdens normale operatie registreert het systeem bij aanraking een nieuw nulpunt voor de respectievelijke as en beweegt daarna een aantal stappen weg van het nulpunt.

De nulpunten van de x- en z-as worden voor beide assen op een andere manier bepaald. Omdat het koppelstuk van de ophanging bij de x-as rotatiesymmetrisch is kan het nulpunt van de x-as worden bepaald door het koppelstuk met het middelpunt in het midden van een gradenboog en 2 van de armen op de as van de gradenboog te positioneren. Daarna kan door middel van een ge-3D-print opzetstuk, zoals te zien in figuur 15, de steppermotor die zich direct achter de camera bevindt, de steppermotor met opzetstuk exact naar 45 graden gedraaid worden. De positie kan dan door de steppermotor worden vastgehouden door er stroom op te zetten waarna de schakelaar en het stopblok kunnen worden gemonteerd.



Figuur 15: Foto's van het x-as opzetstuk

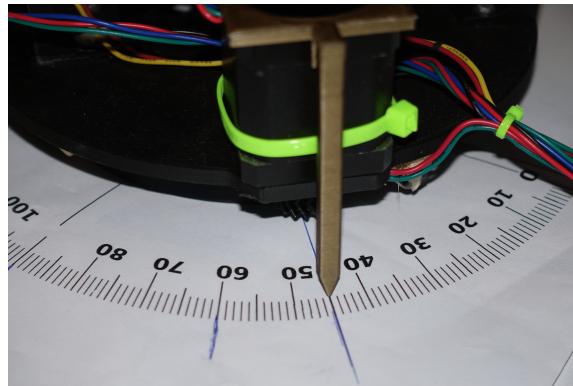
Om het nulpunt van de z-as te bepalen moet de ophanging op een waterpas oppervlak staan. Daarna kan door de arm waterpas te positioneren met de camera naar beneden richtend het nulpunt worden bepaald. Daarna kan de steppermotor van stroom worden voorzien om hem in positie te houden en de schakelaar en het stopblok te monteren.

#### 14.5 Bepalen stappen per graad

Nu de steppermotoren precies kunnen worden bewogen en het nulpunt kan worden bepaald is het laatste wat nodig is het bepalen van het aantal graden per stap. Dit wordt voor beide assen op een verschillende manier gedaan.

##### X-as

De meting op de x-as word gedaan door middel van een op papier geprinte gradenboog en een ge-3D-print opzetstuk voor de x-as. Na nul graden op de gradenboog te hebben geplaatst, kan de x-as worden gedraaid door het handmatig invoeren van een aantal stappen vanaf het nulpunt. Door een bepaalde hoek te kiezen en daar door middel van trial and error naartoe te bewegen kan het aantal stappen om naar die hoek te draaien worden bepaald. Daarna is het aantal stappen per graad te berekenen door het aantal stappen te delen door het aantal graden dat is gedraaid. Bij 1/16 microstepping is dat op de x-as ongeveer 300 stappen per graad beweging.



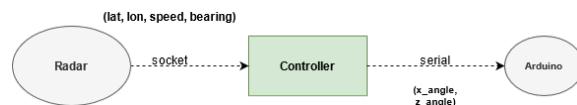
Figuur 16: Foto van de x-as op de gradenboog

### Z-as

Het bepalen van de stappen per graad op de z-as is vergelijkbaar met dat op de x-as alleen het meten van de hoek loopt anders. Omdat de z-as beweegt tussen recht naar boven richten en recht naar onder richten kan een kleine waterpas worden gebruikt voor de meting. Door een waterpas op de arm van de z-as te plaatsen kan door middel van trial en error via het handmatig veranderen van het aantal stappen vanaf het nulpunt bepaald worden hoeveel stappen er nodig zijn om 180 graden te draaien. Daarmee kan dan weer het aantal stappen per graad worden berekend. Bij 1/16 microstepping is dat op de z-as ongeveer 301 stappen per graad beweging.

## 15 Controller

Nu de radargegevens binnengehaald kunnen worden, de hoeken van de assen kunnen worden berekend en de camera precies gericht kan worden moet alles nog worden samengebracht in een gezamenlijke controller. De eerste versie van de controller is in staat om een locatie van de radar te ontvangen, door middel van de eerder beschreven berekeningen een x- en z-hoek uit te rekenen en deze naar de Arduino te sturen.



Figuur 17: Diagram werking eerste controller

De eerste versie van de controller werkt volledig en reageert snel en precies op de locatie die hij van de radar binnen krijgt. Het systeem is in deze fase alleen maar via de terminal uit te lezen en om de camerabeelden te bekijken moet de gebruiker via een externe applicatie de camera openen.

## 16 User interface

Nu de controller de radarbeelden juist in een beweging om kon zetten moest er ook voor de gebruiker een visueel systeem komen om het camera beeld te kunnen zien en op de hoogte gehouden worden van de status van het camerasysteem. Daarnaast moet de gebruiker in staat zijn de x- en z-offsets bij te stellen en zelf de camera te bewegen.

Voor het uitlezen van de beelden van de camera is de OpenCV library toegepast. Deze library kan niet alleen de beelden van de camera ontvangen maar deze ook omzetten in RGB arrays voor de user interface. Hierdoor zijn geen andere libraries meer nodig voor de beeldverwerking.

Om de data en het beeld aan de gebruiker te tonen is gebruik gemaakt van PyGame. Deze simpele user interface heeft alle benodigde onderdelen om zowel het beeld van de camera als status tekst weer te geven. PyGame heeft daarnaast een ingebouwde functionaliteit om alle events in het gemaakte venster af te handelen. Op die manier kan de gebruiker door middel van het toetsenbord commando's geven aan de camera.



Figuur 18: Afbeelding van de user interface en de camera

### 16.1 Commando's

De commando's van de UI zijn gesplitst in twee categorieën: de globale commando's die betrekking hebben tot de algemene werking van de controller en de mode specifieke commando's deze hebben betrekking tot het echte richtsysteem van de camera.

Tot nu toe is de enige globale commando het *mode\_selection* commando. Deze geeft de controller aan in welke mode hij moet werken. Het systeem bestaat voor nu uit twee modes: radargestuurd en handmatig.

De andere commando's van de user interface zijn afhankelijk van de mode waar de controller zich in bevindt. De radargestuurde mode heeft een extra commando om de offsets zoals eerder beschreven door te geven. In de handmatige mode kan met dezelfde knoppen de camera handmatig worden bestuurd door de gewenste hoek aan te passen.

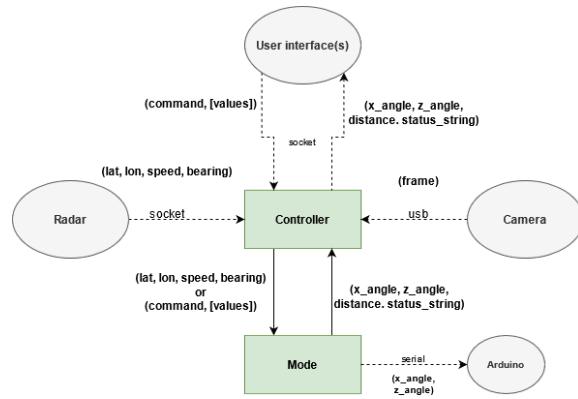
Beide commando's in beide modes hebben de mogelijkheid om de stap grote aan te passen, dit is een intern geregelde variabele in de user interface en hoeft niet naar de controller te worden gestuurd.

Alle commando's van de user interface worden in het volgende format gestuurd:  
*"commando;value,value2,...".*

## 16.2 Communicatie met de controller

Omdat het camerasysteem en de user interface niet op dezelfde plek draaien moet er een verbinding gelegd worden tussen de twee onderdelen. Daarvoor zijn net zoals bij de radar communicatie sockets gebruikt. Sockets werken met het TCP protocol dus elk bericht dat word ontvangen is ook zeker juist. De communicatie bestaat uit twee losse sockets met verschillende functies.

Via de eerste socket stuurt de controller elke loop zijn huidige status met daar in vermeld: de x- en z-hoek van de camera, de afstand tot het gevogde object en een string die de huidige status van de controller in tekst aangeeft.



Figuur 19: Foto van de x-as op de gradenboog

De user interface gebruikt de UI socket om de commando's van de gebruiker door te sturen, zoals het veranderen van de huidige mode van de controller.

Daarnaast moet de gebruiker ook het beeld van de camera kunnen zien, daarvoor word een losse camera socket gebruikt. Met OpenCV word er een frame van de camera uitgelezen die daarna via een externe library, pickle, in een string word omgezet om gestuurd te worden over de socket. Deze string kan door pickle aan de ontvangende kant weer worden omgezet in een object. Omdat de string vrij groot is kan deze niet in een keer worden gestuurd. Daarom moet er een systeem komen om door te geven hoe groot de afbeelding zou gaan worden. Hiervoor wordt door de controller eerst een Integer struct met daarin de grootte in bytes gestuurd.

De ontvanger stopt elke loop de ontvangen data van de camera socket in een data array. Wanneer deze de grote van een Integer struct heeft bereikt word deze uitgelezen en uit de data array verwijderd. Deze struct bevat de grootte van de frame in bytes, de ontvanger wacht dan totdat de data array dezelfde grootte als beschreven of meer bevat. Daarna kan de data van de frame er uit gehaald worden en via pickle weer worden omgezet in een echt frame object.

Het frame object kan daarna via OpenCV worden omgezet in een array van RGB waardes die direct door PyGame kan worden verwerkt om het te laten zien in het PyGame venster.

### 16.3 Implementatie in de controller

De modes van de controller worden toegepast als losse objecten, elke mode bestaat uit 3 functies: De start functie zorgt ervoor dat alle verbindingen zijn gemaakt zodat de mode goed kan functioneren, de main functie handelt de real time radar gegevens en commando's van de gebruiker af en de stop functie sluit alle gemaakte verbindingen die niet meer nodig zijn.

De controller bestaat zoals in de versie zonder user interface uit een main loop waar alles in gebeurt, zoals ook te zien in figuur 20. Deze main loop stuurt het hele programma aan en vanuit hier word ook alle communicatie geregeld elke loop gebeuren de volgende dingen:

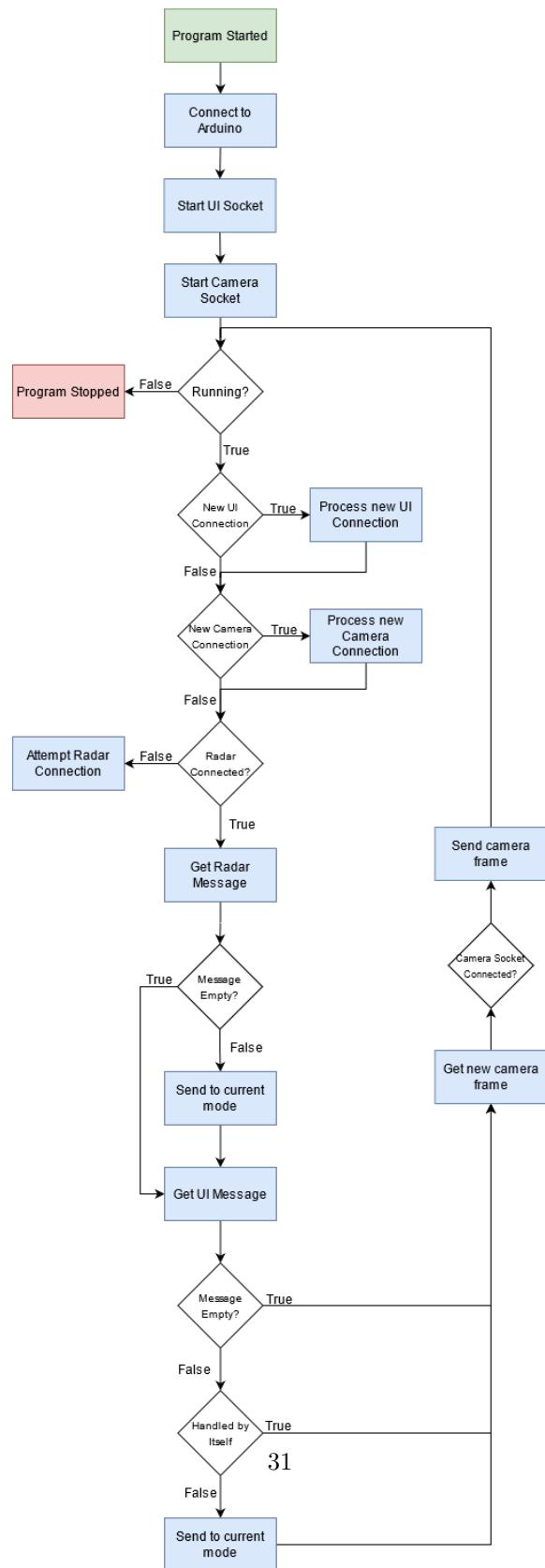
De controller controleert of er nieuwe verbindingen zijn gemaakt met de sockets voor de user interface en het camera beeld. Dit moet gebeuren omdat de controller functioneert als server, de inkomende verbindingen zullen explicet moeten worden geaccepteerd voordat ze te gebruiken zijn.

Daarna controleert de controller of er nieuwe data vanuit de radar binnen is gekomen en stuurt deze door naar het mode object dat op dat moment in gebruik is.

De controller haalt daarna mogelijke berichten van de user interface op. Als deze ook een echt bericht bevatten kunnen er twee dingen gebeuren. De controller handelt zelf gelijk mode selectie commando af en gaat door met de loop of het commando is geen mode selectie en moet afgehandeld worden zoals de huidige mode dat beschrijft. Elke van de twee modes heeft een eigen class met daarin zijn eigen manier om deze commando's af te handelen. Zodra het commando is afgehandeld wordt er door de mode zelf actie ondernomen.

Dan stuurt de controller de nieuwe status van de controller naar de user interface. Dit word op de user interface getoond onder in beeld.

Uiteindelijk haalt de controller een frame op van de camera en stuurt deze samen met een struct met daarin de grootte in bytes van de frame naar de user interface, deze word daar zoals eerder beschreven getoond.



Figuur 20: Vereenvoudigde Flow Chart main loop van de controller met user interface integratie

Met de toevoeging van een user interface aan het systeem was de controller een stuk complexer geworden. De enkele controller class was nu verantwoordelijk voor het omzetten van radar informatie, het communiceren met de hardware, het ontvangen en versturen van commando's van de API en het binnenvullen van camerabeelden van de camera en het versturen naar de user interface.

De extra complexiteit in de controller maakte het lastig om bugs op te lossen die naar boven kwamen. Daarnaast kon door de manier van communiceren wat de status van het systeem was naar de user interfaces er maar één user interface verbonden zijn met de controller.

## 17 Event-based besturing

Na het toevoegen van de communicatie met de user interface in de controller bleek dat dit niet zo soepel liep als gehoopt. Beide processen waren single threaded en zorgde ervoor dat de controller vaak bleef hangen in de communicatie met de user interface in plaats van het afhandelen van nieuwe radargegevens. Daarnaast waren er, hoewel de code in de basis functioneerde, veel bugs bij de communicatie en afhandeling daarvan. Daarom is gekozen om vrijwel alle code opnieuw te schrijven. Deze keer zijn alle onderdelen in classes verdeeld en functies duidelijk beschreven met typeaanduiding voor zowel parameters als return waarden.

De volledige code is ook herschreven met het gebruik van standaard engineering units in gedachte. Zo worden op alle plekken dezelfde eenheden voor de grootheden gebruikt. Denk aan meters voor afstand en graden voor hoeken.

De nieuwe event-based versie verschilt met de eerdere versie op 5 grote vlakken:

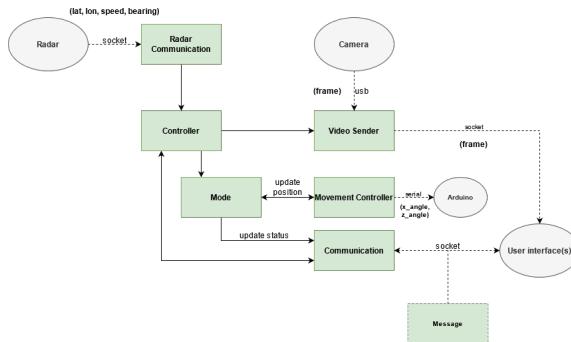
### 17.1 Message class

Het eerste punt dat is aangepakt in de nieuwe versie is hoe commando's werden verstuurd. Waar hiervoor een simpele string werd gebruikt worden de berichten, ook wel events, nu ondergebracht in een Message object. Dit object bestaat uit twee onderdelen. De naam of het doorgegeven commando en de meegegeven waarden.

Het Message object kan zichzelf omzetten naar bytes om te kunnen worden verzonden en de constructor van de class accepteert dezelfde bytes om precies hetzelfde object opnieuw te maken met alleen de ontvangen bytes. Hierbij wordt ook de pickle library gebruikt. Door deze functionaliteit zijn de objecten makkelijk over sockets te versturen zonder dat extra functies voor nodig zijn in de communicatie classes.

### 17.2 Communicatie classes

De communicatie class is gesplitst in 3 versies: de algemene communicatie met de user interface, de communicatie met de radar en de video verbinding met de user interface. Dit is ook te zien in figuur 21. De 3 versies werken alle 3 met sockets maar er zijn wel verschillen in de manier waarop de classes omgaan met inkomende data:



Figuur 21: Diagram nieuwe event based controller

### Algemene communicatie

De algemene communicatie class werkt met het eerder beschreven Message object en haalt deze binnen op dezelfde manier als hoe de frames voorheen werden binnengehaald. Voor elk bericht dat wordt verzonden via deze class wordt een struct verzonden met de lengte van het bericht in bytes. Hierdoor weet de ontvangende kant exact waar het bericht begint en ophoudt.

De ontvangen berichten worden door de communicatie class in een Queue geplaatst, hier blijven de berichten totdat de controller klaar is om ze af te handelen.

### Radar communicatie

Aan de radar communicatie is in de nieuwe versie niet zo veel veranderd aangezien deze altijd goed heeft gewerkt. Alle communicatie is naar een eigen class overgeplaatst. Daarnaast maakt de radar communicatie van de strings die worden ontvangen vanuit de radar een bericht in een Message object. Op die manier kunnen ze als event afgehandeld worden.

### Video communicatie

Ook bij de video communicatie is niet veel veranderd, alle code is naar een eigen class gebracht. Hierbij is het Queue principe niet overgenomen omdat alleen de laatste frame relevant is om te laten zien.

## 17.3 Multithreading

De grootste verandering voor de snelheid en efficiëntie van de controller was het toevoegen van multithreading aan de video zender en ontvanger. De zender en ontvanger objecten hebben een eigen main loop die door de threading module van python worden gecontroleerd. De objecten bestaan nog wel als variabelen in de controller en user interface objecten. Op die manier kan het laatst ontvangen frame worden verkregen door de user interface en kunnen de loops worden gestopt vanaf buiten af.

Het video zender object haalt zelf de frames op van de camera en zorgt dat deze verstuurd worden naar de ontvanger. De controller hoeft de zender dus alleen maar te starten en te stoppen daarbuiten is er geen overzicht nodig vanuit de controller.

## 17.4 Global handlers en modes

Met de nieuwe Message objecten worden events gekenmerkt via de naam van de Message. Hier is het nieuwe event handling systeem omheen gebouwd. Zowel de controller als de modes hebben functies met dezelfde namen als de namen die door de user interface en de radar communicatie worden meegegeven aan de events.

De controller zoekt eerst of de naam overeen komt met een van de globale handlers, deze handlers zijn verantwoordelijk voor de algemene taken van de controller zoals het verwisselen van modes of het resetten van het systeem.

Als de controller zelf geen handler heeft voor het event word het doorgegeven aan het mode object dat op dat moment actief is. De mode objecten werken niet meer zoals eerst met een losse start, main en stop functie maar functioneren nu volledig vanuit hun handle functie.

Deze handle functie zoekt net zoals bij de controller naar een functie die overeenkomt met het commando dat via het event is gegeven. Als die overeenkomst word gevonden worden alle waardes die meegegeven zijn in het Message object doorgegeven aan de gevonden functie. Deze functie genereert dan een nieuw Status object die naar alle user interfaces moet worden gestuurd.

## 17.5 Status

Alles met betrekking door de status van de controller word nu exclusief door de controller bijgehouden. Waar voorheen de controller en de user interface een eigen losse status hadden gebruikt de user interface nu de status zoals verkregen van de controller. Hier maakt de user interface zelf geen aanpassingen aan, na elk commando wacht hij af totdat een nieuwe status door de controller word aangeleverd.

Op deze manier is het niet meer mogelijk voor de controller en de user interface om niet gelijk te lopen op bepaalde punten.

Het status object word als parameter in een Message object verzonden naar de user interfaces via de algemene communicatie class.

## 17.6 Meerdere user interfaces

Doordat de controle over modes en de status nu zijn verplaatst van de user interface naar de controller is het ook mogelijk geworden om met meerdere user interfaces te verbinden. Bij elke verandering door welke user interface dan ook word de nieuwe status naar alle verbonden user interfaces gestuurd. Mocht een status bericht niet aankomen bij de user interface dan word de verbinding automatisch verbroken.

## 17.7 Betere uitbreidbaarheid

De herschreven controller bestaat nu uit losse classes die stuk voor stuk veranderd of vervangen kunnen worden zonder dat het effect heeft op de rest van het systeem. Op deze manier zijn de communicatie classes bijvoorbeeld makkelijk te veranderen of vervangen door een nieuw protocol of library toe te passen zonder dat de rest van de controller aangepast hoeft te worden.

## 18 Conclusie

Het doel van het project was om een camerasysteem te ontwerpen en te realiseren die op basis van radargegevens een object op het wateroppervlak kan volgen. Daarvoor zijn twee losse systemen ontworpen en gerealiseerd, het richtsysteem en de controller. Het richtsysteem is in staat om een camera precies te richten op de x- en z-as. De controller is in staat om de radargegevens om te zetten naar een bruikbare hoek voor het richtsysteem. Door de combinatie is de camera door middel van de radargegevens precies te richten op een door de radar gedetecteerd object op het wateroppervlak.

De controller is daarnaast in staat om te communiceren met een user interface om de gebruiker besturing over de camera te bieden en de camerabeelden te tonen aan de gebruiker.

Het gerealiseerde proof of principle is klaar om als proef toegepast te worden op de kade en kan met minimale aanpassingen en met de toevoeging van een stabilisatie systeem ook op schepen worden toegepast. Jammer genoeg kon het systeem, door vertraging bij het havenbedrijf Rotterdam, nog niet buiten een simulatie worden getest in de haven van Rotterdam.

## 19 Aanbevelingen

Nog niet alle punten uit het project zijn af gekomen of nog niet helemaal zoals ik had gehoopt, de volgende punten zijn daarvan de voornaamste:

### Beter afhandelen threading

De manier waarop threading nu werkt in het project is functioneel alleen nog niet zo netjes als mogelijk. Vooral het afsluiten van threads geeft vaak exceptions dit komt omdat de thread vaak nog niet helemaal op de hoogte is van het feit dat hij moet sluiten als de garbage collector hem forceert om de sluiten.

### Communicatie in losse threads

Om de controller nog soepeler te laten lopen kan ook het ontvangen van berichten in een losse thread worden verwerkt. Hierdoor is de controller geen tijd kwijt met het wachten op het communicatie object om nieuwe berichten op te halen.

Let hierbij wel op dat het mogelijk is dat zowel de controller als het communicatie object zelf op hetzelfde moment de Queue van berichten proberen aan te passen. Maak dus handig gebruik van de lock module.

### Encryptie tussen controller en user interface

Om te voorkomen dat ongewenste gebruikers de camera's zomaar overnemen moet er encryptie worden toegepast op de communicatie. Daarnaast moet ook een authenticatie methode worden toegepast zodat alleen gebruikers met toegang de camera's kunnen besturen.

Daarnaast is het misschien nuttig om te experimenteren met hogere level communicatie dan sockets, bijvoorbeeld websockets of https.

### **Betere bedrading nulpunt schakelaars**

De huidige nulpuntschakelaars zijn bedraad met een solid core draad. Deze is nogal makkelijk te breken bij veel beweging het draaien van de camera. Voor toekomstige versies moeten de draden sowieso vervangen worden maar ik zou aanraden om de schakelaar op een pcb of klein project bord te solderen en daar ook een connector aan vast te maken. Op die manier kan bij kabelbreuk de kabel gemakkelijk vervangen worden.

### **Aansturen zoom**

In het proof-of-principle is de zoom niet uitgewerkt dit moet echter nog wel gedaan worden. Dit kan naar mijn idee op twee manieren. Door de schakelaar uit de camera te halen en deze via een relais of iets dergelijks aan te sturen via de Arduino. Dit is echter een lastige opgave aangezien niet precies bekend is hoe het er aan de binnenkant uit ziet en de kans nogal groot is dat de camera wordt beschadigd tijdens het openmaken.

Een beter alternatief is de zoom knop aansturen met een servo, hierbij moet wel opgelet worden dat de schakelaar analog is. Hij zoomt sneller in of uit op basis van de positie van de knop.

### **Controller bepaalt wat er op beeld te zien is**

Om ervoor te zorgen dat de user interface niet steeds ge-update moet worden als er nieuwe commando's worden toegevoegd kan dit in principe ook aangestuurd worden vanuit de controller. Zoals de controller nu de status meegeeft kan de controller ook aangeven welke knoppen er gebruikt kunnen worden door de gebruiker en wat ze moeten doen.

Een ander alternatief zou zijn om de user interface van python naar een web based format te verplaatsen. Hierdoor is de gebruiker altijd up to date met de laatste versie waar hij of zij dan ook is. Geen handmatige updates maar nodig. Ook dit zou een grote stap zijn maar zou prima samengaan met een grote aanpassing aan de communicatie.

## Referenties

- [1] Calculate distance, bearing and more between latitudelongitude points. <http://www.movable-type.co.uk/scripts/latlong.html/>.
- [2] whats the difference between servo and stepper motors. Accessed: 2020-02-28.
- [3] M. Arbabian Helen and R. Cogburn. "Choosing the Right Motor for Your Project – DC vs Stepper vs Servo Motors. Accessed: 2020-02-28.
- [4] P. Hut. "What's The Difference Between DC, Servo and Stepper Motors? Accessed: 2020-02-28.
- [5] KNRM. Wat is onderkoeling? <https://www.knrm.nl/images/downloads/Onderkoeling.pdf>.
- [6] Dr J. Nedwell, Mr J. Langworthy, and Mr D. Howell. Assessment of sub-sea acoustic noise and vibration from offshore wind turbines and its impact on marine wildlife; initial measurements of underwater noise during construction of offshore windfarms, and comparison with background noise. Technical report, COWRIE, May 2003.
- [7] Eric Rice. Stepper motor torque basics. <https://www.controleng.com/articles/stepper-motor-torque-basics/>, 2018.
- [8] Gabriel A. Rincón-Mora and Neeraj Keskar. Unscrambling the power losses in switching boost converters. <https://www.eetimes.com/unscrambling-the-power-losses-in-switching-boost-converters/>, 2006.
- [9] Akshay Upadhyay. Formula to find bearing or heading angle between two points: Latitude longitude. <https://www.igismap.com/formula-to-find-bearing-or-heading-angle-between-two-points-latitude-longitude/>.
- [10] J. Vrooman, G. Schild, A.G. Rodriguez, and F. van Hest. *Windparken op de Noordzee : kansen en risico's voor de natuur*. 2018.

## **Bijlagen**

Alle bijlagen zijn ook los in de opleverset te vinden.

### **19.1 Bijlage A: Test verslag**

# **Testomschrijving camerasytsem**

**Gerard van Walraven**  
0950584

**8 september 2021**

# **1 Simulatie**

Omdat het testen niet met operationele radar kan gebeuren is een simulatie geschreven om de radar na te bootsen. De simulatie bestaat uit een auto die met 30 km/h over de Fortunaweg in Schiedam rijdt zoals gezien vanuit het raam op de 1e verdieping bij Sens2Sea B.V.

## **2 Testplan**

### **2.1 Beweging**

**Het richtsysteem kan de camera naar een precieze hoek bewegen zoals doorgegeven over de seriële verbinding.**

Geef via de seriële verbinding het richtsysteem de opdracht op de camera naar twee willekeurige hoeken te bewegen. Test daarna zoals bij de stappen per graad calibratie door middel van een gradenboog, het x-as opzetstuk en een digitaal waterpas of de juiste hoek is bereikt.

**De camera volgt het object op basis van de simulatie.**

Plaats de camera in het juiste raam en activeer de simulatie. Als de camera de weg langzaam volgt en op de hoek van de straat en voor het gebouw keert word de simulatie juist gevolgd.

### **2.2 User interface**

**De controller reageert binnen 1 seconde op de input van de user interface**

Verbind de user interface met de camera en probeer elk mogelijke commando ik elk van de modes uit. Gebruik als het nodig is een stopwatch om de tijd tussen het indrukken van een knop en de update op het scherm bij te houden.

**De videoverbinding tussen de controller en de user interface loopt maximaal 1 keer per 30 seconde een frame vast**

Bekijk minimaal 5 minuten het beeld van de camera terwijl de simulatie word uitgevoerd. Maak notitie van elke keer dat het beeld vastloopt.

### **2.3 Landurig betrouwbaar**

**(Toegevoegd na de test in week 29)**

**De controller functioneert na 2 uur lang simulatie nog naar behoren en reageert op inputs van de user interface**

Laat het systeem minimaal 2 uur lang met de simulatie aan draaien. Test daarna of de simulatie nog goed word gevolgd en of de inputs van de user interface nog werken.

**De controller functioneert na 2 uur lang niks te doen nog naar behoren en reageert op inputs van de user interface**

Laat het systeem minimaal 2 uur lang zonder enige inputs draaien. Test daarna of de simulatie nog kan worden gevuld en of de inputs van de user interface nog werken.

## 3 Testresultaten week 27

### 3.1 Beweging

**Het richtsysteem kan de camera naar een precieze hoek bewegen zoals doorgegeven over de seriële verbinding.**

Op beide assen werden op de gradenboog en door de waterpas de juiste hoek aangegeven zoals de instructie aangaf.

**De camera volgt het object op basis van de simulatie.**

Het beeld van de camera bewoont juist over het wegdek zoals verwacht.

## 4 Testresultaten week 29

### 4.1 Beweging

**Het richtsysteem kan de camera naar een precieze hoek bewegen zoals doorgegeven over de seriële verbinding.**

Op beide assen werden op de gradenboog en door de waterpas de juiste hoek aangegeven zoals de instructie aangaf.

**De camera volgt het object op basis van de simulatie.**

De camera volgde de virtuele auto juist over het wegdek.

Een probleem in de volg code werd opgemerkt waardoor de x-as alleen in de positieve richting bewoont en dus niet terug draaide. Dit probleem is ter plekke opgelost waarna de test wel succesvol is afgerond.

### 4.2 User interface

**De controller reageert binnen 1 seconde op de input van de user interface**

De user interface reageerde in de meeste gevallen direct op de input, er was zelfs te weinig tijd om de stopwatch te starten.

In enkele gevallen liep de controller vast op de inputs van de user interface, dit kwam voor als een knop heel snel achter elkaar werd ingedrukt.

**De videoverbinding tussen de controller en de user interface loopt maximaal 1 keer per 30 seconde een frame vast**

In de 5 minuten tijdens de test is het beeld 3 keer waarneembaar vast gelopen. Dit gebeurde allemaal met meer dan 30 seconde tussen de momenten.

Wel is genoteerd dat bij het snel indrukken van knoppen op de user interface het beeld vast kan lopen, soms voor lange tijd.

### **4.3 Extra constatering**

Hoewel de tests allemaal goed zijn uitgevoerd is wel een nieuwe bug gevonden. Het systeem stop na ongeveer 10 minuten met het juist en op tijd reageren op de inputs van de radar en de user interface met betrekking tot beweging. Alle andere inputs werken nog wel prima.

## **5 Testresultaten week 31**

### **5.1 Beweging**

**Het richtsysteem kan de camera naar een precieze hoek bewegen zoals doorgegeven over de seriële verbinding.**

Het richten werkt naar behoren. Beide hoeken werden netjes door de motoren bereikt.

**De camera volgt het object op basis van de simulatie.**

De camera volgt het virtuele object netjes langs de Fortunaweg.

Opgemerkt is wel dat als de simulatie/radar net iets te laat is met het sturen van een nieuwe positie de motoren vrij abrupt stoppen. Hoewel dit wel de bedoeling is zou dit misschien iets gedempt kunnen worden door de maximale versnelling te verlagen.

### **5.2 User interface**

**De controller reageert binnen 1 seconde op de input van de user interface**

De controller reageert consistent sneller dan de stopwatch kan worden gestart. Daarnaast zijn de eerdere problemen met vastlopen verholpen.

**De videoverbinding tussen de controller en de user interface loopt maximaal 1 keer per 30 seconde een frame vast**

In de 5 minuten van de test is het beeld geen enkele keer waarneembaar vastgelopen.

### **5.3 Landurig betrouwbaar**

**(Toegevoegd na de test in week 29)**

**De controller functioneert na 2 uur lang simulatie nog naar behoren en reageert op inputs van de user interface**

De controller functioneert na 2 uur nog goed, wel zijn de heatsinks op de drivers erg warm geworden.

**De controller functioneert na 2 uur lang niks te doen nog naar behoren en reageert op inputs van de user interface**

De controller functioneert nog naar behoren en nieuwe inputs worden juist verwerkt.

## Change log

Versie	Datum	Wijzigingen
1	30/03/2021	Titelpagina, inhoudspagina en eerste versie probleemopstelling uitgewerkt.
2	31/03/2021	Bepaling positie en heading camera en Binnenhalen radar gegevens aan Uitwerking toegevoegd.
3	02/04/2021	Herinrichting uitwerking en hoek bepaling toegevoegd. Stage planning aan de bijlage toegevoegd.
4	05/04/2021	Keuze camera, benadering tilt en zoom toegevoegd. Keuze voeding toegevoegd.
5	22/04/2021	Inleiding uitgebreid met uitleg probleem
6	23/04/2021	Uitbreiding inleiding fundering windmolens
7	03/05/2021	Eisen camera toegevoegd
8	07/05/2021	Aanpassing conclusie camera
9	11/05/2021	Stukken tekst herschreven en afbeeldingen toegevoegd voor de duidelijkheid
10	13/05/2021	Tekst gecontroleerd op spelfouten en een paar stukken herschreven
11	07/06/2021	Nieuwe versie 95% presentatie
12	20/08/2021	Laatste versie herkansing 95% presentatie
13	07/09/2021	Definitieve versie