Nr. _____ av _____

Sjøkrigsskolen

Bacheloroppgave

Augmented Reality – Fremtidens verktøy for navigatører

av

Christer Algrøy, Julian Tobias Venstad og Markus Øvstedal

Levert som en del av kravet til graden:

BACHELOR I MILITÆRE STUDIER MED FORDYPNING I ELEKTRONIKK OG DATA

Innlevert: 4 desember 2019

Godkjent for offentlig publisering

I. Publiseringsavtale

En avtale om elektronisk publisering av bachelor/prosjektoppgave

Kadetten(ene) har opphavsrett til oppgaven, inkludert rettighetene til å publisere den.

Alle oppgaver som oppfyller kravene til publisering, vil bli registrert og publisert i Bibsys Brage når kadetten(ene) har godkjent publisering.

Oppgaver som er graderte eller begrenset av en inngått avtale vil ikke bli publisert.

Vi gir herved Sjøkrigsskolen rett til å gjøre denne oppgaven tilgjengelig elektronisk, gratis og uten kostnader	[X] Ja	[ ] Nei
Finnes det en avtale om forsinket eller kun intern publisering? (Utfyllende opplysninger må fylles ut) Hvis ja: kan oppgaven publiseres elektronisk når embargoperioden utløper?	[ ] Ja [ ] Ja	[X] Nei [ ] Nei

II. Plagiaterklæring

Vi erklærer herved at oppgaven er vårt eget arbeid og med bruk av riktig kildehenvisning. Vi har ikke nyttet annen hjelp enn det som er beskrevet i oppgaven.

Vi er klar over at brudd på dette vil føre til avvisning av oppgaven.

Dato: 03 - 12 - 2019

Kadett navn	Signatur
Kadett navn	Signatur
Kadett navn	Signatur

II. Forord og Takksigelser

Denne oppgaven tar for seg bruken av augmented reality (AR)-teknologi i navigasjon. Valget av tema kommer av en bestilling fra sjøforsvarets navigasjon og kompetansesenter, samt egen interesse for den fremtidsrettede teknologien. Leseren burde på forhånd ha kjennskap til integrerte maritime navigasjonssystemer, samt en grunnleggende forståelse av programmering og datakommunikasjon. Oppgaven kan være relevant for navigatører både sivilt og militært, samt teknologikyndige med interesse for augmented reality og datakommunikasjon.
Oppgaven hadde ikke blitt like omfattende uten god hjelp av flinke ansatte ved FHS Sjøkrigsskolen. Det finnes et kjent finsk ordtak som sier: "*Hjelp et menneske i motbakke, ikke når han er kommet opp*". Ordtaket gjenspeiler den hjelpen og veiledningen vi har fått gjennom bachelorprosjektet. Vi vil derfor benytte denne anledningen til å takke:

OK Petter Lunde, KVM Martin Frotvedt og Navigasjon og kompetansesenteret
For et godt samarbeid i forbindelse med bruk av simulatoranlegget gjennom hele prosjektperioden.

Avdelingsingeniør Frode Wikne
For profesjonell og inspirerende hjelp til utvikling av prototyper til prosjektet.

Førsteamanuensis Alexander Sauter
Takk for fruktbare diskusjoner og gode tilbakemeldinger. De sies at fire øyner er bedre enn to, og av og til trengs det en god veileder for å finne riktig kurs.

KL Odd Sveinung Hareide
For god hjelp når det kommer til design og navigasjonstekniske problemstillinger. Din kompetanse og svært brede nettverk har vært viktig for oppgaven.

Vi vil også takke alle dere som har tatt dere tid til å hjelpe oss. Om dette har vært fem minutter i gangen, formelle tester i simulator eller bare deling av tanker rundt oppgaven. Dere har alle vært en viktig ressurs for oppgaven. Avslutningsvis vil vi takke Fostech som har vært så hyggelig å låne oss Microsoft HoloLens gjennom hele prosjektet. Til dere som leser oppgaven håper vi dere får et innblikk i bruken av augmented reality i navigasjon, ser teknologiens muligheter - samtidig lærer noe nytt.

III. Sammendrag

Augmented reality er en spennende teknologi som for mange kan oppleves som futuristisk og prematur. Samtidig har teknologien hatt en enorm vekst de siste årene. Med store internasjonale firmaer på banen som Microsoft og Google, har vi de siste årene fått introdusert lette og brukervennlige hode-monterte brillesett. Slik satsing fører til at flere og flere har fått øynene opp for teknologien og dens muligheter. Denne oppgaven tar for seg hvordan man kan designe, implementere og teste et augmented reality system som kan brukes av navigatører under hurtigbåtnavigasjon.

Oppgaven tar utgangspunkt i å bruke Microsoft sine HoloLens Augmented Reality briller, som skal testes i Sjøforsvarets Navigasjons- og kompetansesenter sitt simulatoranlegg. Gjennom implementeringen gjør vi rede for hvordan vi ved hjelp av det flytbasert programmeringsverktøyet Node-RED, programmeringsspråket C# (uttales "C sharp") og spillmotoren Unity har utviklet programvare som henter navigasjonsdata fra simulatoranlegget, behandler dataen og viser den i det valgte brillesettet. En liten datamaskin leser av navigasjonsdata igjennom en kabel på broen. Dataen blir behandlet i Node-RED, og sendt via det lokale nettverket til brillesettet. Brukergrensesnittet er designet i Unity, og det er her farge, størrelse og plassering blir bestemt. Brukeren kan dog gjøre mindre personlige tilpasninger ved hjelp av en nettside. C# koden styrer hvordan dataen blir mottatt og presentert i brillene.

Resultatet av oppgaven er et brukergrensesnitt som fungerer og har blitt utviklet gjennom gjentatte tester i simulatoranlegget. Brukergrensesnittet gir navigatøren muligheten til å se sin egen fart og kurs, samt informasjon om inneværende og neste leg. Informasjonen som blir presentert er utviklet i samarbeid med navigasjonskyndig personell. Tilbakemeldingene fra de ulike testpersonene konvergerer i stor grad mot at systemet fungerer og kan gi mentalt overskudd. Til tross for dette er også testpersonene veldig samstemt i at sentrale faktorer som synsbredde, ergonomi og prosessorkraften til brillene må utbedres før teknologien kan implementeres.

IV. Nomenklatur

Definisjoner	
Automatic identification system	Automatisk identifikasjonssystem som fungerer ved at fartøy som har systemet transmitterer identitet, posisjon, fart og kurs samt annen informasjon
Brillesett	Blir brukt for å omtale Microsoft HoloLens.
Checksum	En metode for å sjekke om dataene dine har feil/er korrupte. Dersom checksum stemmer betyr det at dataene er valide.
Extensible Markup Language	Extensible Markup Language (XML) er et markeringsspråk som definerer et sett med regler for koding av dokumenter i et format som er lesbart for både mennesker og maskiner.
Field of view/synsbredde	Hvor stor andel av synsfeltet ditt som kan brukes til å presentere data.
Mixed Reality	Blanding av den virkelige verden og den virtuelle verden.
Mixed Reality Capture	En funksjonalitet i enhetsportalen til HoloLens som gjør at man kan ta bilde, videoopptak eller direktestrømning av hva man ser i HoloLens til en datamaskin. Nyttig for å feilsøke.
Navkomp	Brukes som referanse til sjøforsvarets navigasjon- og kompetansesenter.
Node	Ferdigprogrammerte funksjonsbrikker, som man kan legge i kjeder for å oppnå ønsket resultat.

Node-RED	Et visuelt programmeringsverktøy ment for å koble sammen forskjellig fastvare, programmeringsgrensesnitt og internett-tjenester. Verktøyet benytter seg av noder
Rutefil	En fil som inneholder data om en planlagt navigasjonsrute, for eksempel se *vedlegg L*.
Samba	Programvare for fildeling imellom Windows og Unix.
Stevn	Et objekt man skal seile mot eller ha rett bak seg på et legg. Typiske stevnobjekt er lykter og blinker.
Streng	En rekke med tegn.
Usability Engineering:	Fagfelt som omhandler HMI og å skape brukervennlige løsninger.
VNC Viewer	Verktøy for å fjernstyre datamaskiner.
Waypoint	Ett punkt i kartet man skal seile forbi for å følge ruten, blir brukt som neste punktet man skal seile til. Merk: Bruker veipunkt i oppgaven
Wheel over line	Linje i ECDIS som viser hvor man burde tørne for å komme rett inn på ny kurs. Merk: Bruker Tørnlinje i oppgaven
Wheel over Point	Punkt i ECDIS som viser hvor man burde tørne for å treffe rett inn på ny kurs. Merk: Bruker Tørnpunkt i oppgaven
Widget	En mindre modul i et grafisk brukergrensesnitt. Den presenterer enten informasjon, og kan i noen tilfeller la brukeren interagere med det.

Forkortelser	
AIS	Automatic identification system
API	Application Programming Interface Programmeringsgrensesnitt på norsk.
AR	Augmented Reality. Utvidet virkelighet på Norsk
DGPS	Differential Global Positioning System.
ECDIS	Electronic Chart Display and Information System.
HMD	Head Mounted Display.
HMI	Human Machine Interface.
HPU	Holografisk Prosesserings Enhet.
HUD	Head Up Display.
IMU	Inertial measurement unit.
IP-Adresse	Internet Protocol-adresse.
NMEA	National Marine Electronics Association.
XML	Extensible Markup Language (Se Definisjoner)

V. Innholdsfortegnelse

VI. Bildetekstliste

VII. Bildetekstliste Vedlegg

Vedlegg D.

Vedlegg E.

Vedlegg O.

Vedlegg Q.

1 Introduksjon

1.1 Bakgrunn

"Any sufficiently advanced technology is indistinguishable from magic."
Arthur C. Clarke (1984)

Moderne teknologi har de siste årene hatt en enorm vekst med gjennombrudd som sensorteknologi basert på kvantefysikk, kunstig intelligens og målrettede energivåpen. Et annet viktig, men kanskje undervurdert gjennombrudd er utviklingen av augmented reality teknologi. Den anerkjente teknologigründeren Tim Sweeney, mener at augmented reality (AR) vil være den største teknologiske revolusjonen i vår levetid (Straw, 2015). Til tross for å være en futuristisk teknologi, som blant mange blir oppfattet som prematur, har teknologien hatt en enorm vekst de senere årene. Med globale teknologigiganter som Microsoft og Google på banen har AR-teknologi blitt et viktig satsningsområde mot fremtidens teknologi, med bruksområder innenfor olje og gass, finans og shipping. Etter lanseringen av lette og brukervennlige briller som *Magic Leap* og *Microsoft HoloLens* har også den maritime verden fått økt interesse. Sjøforsvaret har også fått opp øynene for den revolusjonerende teknologien og har involvert seg i en rekke prosjekter som tar for seg denne nyskapende teknologien. Med dette som utgangspunkt kan det se ut som augmented reality vil kunne bli et viktig verktøy for fremtidens navigatører.

Bakgrunnen for oppgaven tar utgangspunkt i at det er skrevet adekvate mengder litteratur på bruken av augmented reality, både lokalt i form av bacheloroppgaver, men også en rekke publikasjoner fra sjøforsvarets navigasjons og kompetansesenter. Ser man dette i lys av annen litteratur som er produsert i forbindelse med augmented reality kan det argumenters for at det er et tilstrekkelig teoretisk rammeverk innenfor temaet. Til tross for dette er det blitt designet og utviklet svært lite program- og maskinvare som benytter seg av augmented reality i en maritim kontekst. Dette resulterer i at det finnes stort potensiale for utviklingen av teknologien i fremtiden. Dette fører til at vi anser oppgaven som relevant i den grad at vi har som mål å sette teorien i praksis og få testet ut konseptet som det har blitt skrevet så mye om.

1.2 Oppgaveformulering

Vi skal i denne oppgaven designe, implementere og teste et system som gjør det mulig å fremvise navigasjonsdata i et sett med augmented reality briller. Brillene skal kunne brukes i navigasjonssimulatoren ved sjøforsvaret navigasjons og kompetansesenter. Brillesettet som skal brukes i oppgaven er Microsoft HoloLens sine førstegenerasjons briller. Vi har satt følgende krav til prototypen:

Prototypen må minimum kunne:

- Vise navigasjonsdata i brillene.
- Være enkel i bruk.
- Vise hensiktsmessig informasjon.

Prototypen bør kunne:

- Motta data trådløst.
- Bruke stemmestyring/håndkontroller til å interagere med systemet.
- Vise alarmer/meldinger som er relevant for en navigatør.
- Markere og interagere med objekter i rommet.

Dette impliserer at systemet skal:

- Håndtere overføringen av en rekke verdier mellom server og klient.
- Formatere og presentere verdiene for brukeren.
- Håndtere brukerinteraksjoner.
- Kjøre over lengre tid uten avbrudd eller forsinkelser.

1.3 Avgrensing

Oppgaven avgrenses til å omhandle selve designet av programvare for fremvisning av navigasjonsdata i AR-briller. Herunder vil oppgaven spesielt sette lys på de tekniske løsningene som gjelder design av brukergrensesnitt, samt datakommunikasjon mellom simulator og AR-brillesettet. Videre vil oppgaven ta for seg testing av prototypen utviklet i oppgaven. Dette resulterer i at det ikke blir gjort egne, dypere studier på valg av relevant navigasjonsteknisk informasjon, da dette faller utenfor det tekniske fagfeltet. Relevant litteratur på dette vil bli vedlagt for videre lesning. En annen viktig presisering er at det ikke vil bli presentert et fullstendig teoretisk rammeverk på alle konsepter innenfor datakommunikasjon og programmering, da det forventes at leseren har grunnkunnskaper på temaet. Det vil bli lagt med forslag til litteratur som kan leses for å få en bedre forståelse av oppgaven. Andre begrensninger er at design, utvikling og testing vil foregå i forbindelse til eller ved simulatoranlegget på FHS Sjøkrigsskolen og ikke på et fysisk fartøy.

På grunn av sikkerhetsmessige hensyn fra *Kongsberg Digital* vil vi ikke koble oss på deres ECDIS systemer. Forbudet kommer av at *Kongsberg Digital* ikke ønsker at vi skulle tukle for mye med deres systemer. Dette resulterer i at vi ikke får tilgang til ECDIS sine data fra rutemonitoreringsvinduet.

1.4 Struktur

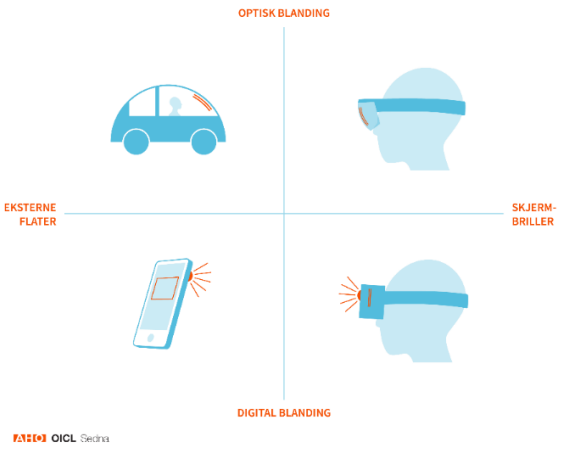
Oppgaven består av åtte kapitler, hvor hvert kapittel inneholder ulike delkapitler. Først gjør vi rede for teori som vil hjelpe leseren forstå oppgaven videre. Deretter gjøres det rede for hvilke metoder vi har brukt for å komme frem til vår prototype og for testene i etterkant. Under kapittelet implementering vil leseren få et innblikk i hvordan de valgte løsningene har blitt til. Etter dette vil våre resultater og funn gjøres rede for, før vi til slutt vil dele våre tanker og erfaringer gjort under og i etterkant av utviklingen. Vedleggene er nummerert alfabetisk. Referanser vil følge APA 6th sitt referansesystem. Det vil bli lagt referanser i oppgaven, etterfulgt av en utfyllende referanseliste i *kapittel 9.*

2 Teori og maskinvare

2.1 Augmented Reality

Hvorvidt man vil bruke det engelske ordet augmented reality, den norske versjonen utvidet virkelighet eller nynorsken sin auka røynd, beskriver alle teknologien som gjør at vi kan se virtuelle objekter i den fysiske virkeligheten (Azuma, 1997). Til tross for dette finnes det en rekke definisjoner på hva augmented reality er. En enkel, men fremdeles presis definisjon kan være at AR handler om å plassere digitale objekter på relevante steder i den fysiske virkelighet (Urke, 2018, 20). En annen mer omfattende definisjon kan være Schmalstieg og Höllerer sin definisjon fra 2016, som sier at AR ønsker å presentere informasjon som er tilknyttet den virkelige verden (Schmalstieg og Höllerer, 2016, s. 3). Schmalstieg og Höllerer legger til at augmented reality går forbi vanlig menneske-maskin interaksjon, i den grad at det bringer sammen den virtuelle og den reelle verden (Schmalstieg og Höllerer, 2016, s. 3). Videre i augmented reality miljøet blir ofte definisjonen til Ronald T. Azuma betegnet som den mest aksepterte. Han la frem i forskningsartikkelen sin fra 1997 at AR må oppfylle følgende kriterier: Kombinere det reelle og virtuelle, interagere i sanntid og operere i tre dimensjoner (Azuma 1997, s. 335). Det er viktig å legge til at denne definisjonen ikke krever spesifikt utstyr som et head-mounted display (HMD) (Schmalstieg og Höllerer, 2016, s. 3).

I tillegg til disse tradisjonelle tekstbok definisjonene kan artikkelen *Teknikkar for Auka Røynd (AR)* skrevet av John Olav H.Eikenes være relevant for virkelig å forstå konseptet. Her skiller han mellom eksterne flater og skjermbriller for å beskrive hvor utvidingen av virkelighet foregår (Eikenes, 2018). Videre skiller han mellom optisk og digital blanding for å beskrive hvordan den fysiske og virtuelle verden blir kombinert (Eikenes, 2018). Kombinert blir dette satt sammen i et informativt diagram som vist nedenfor. Dette blir vist på *Figur 1*.



Figur 1 Eksterne flater og skjermbriller i kombinasjon med optisk og digital blanding.

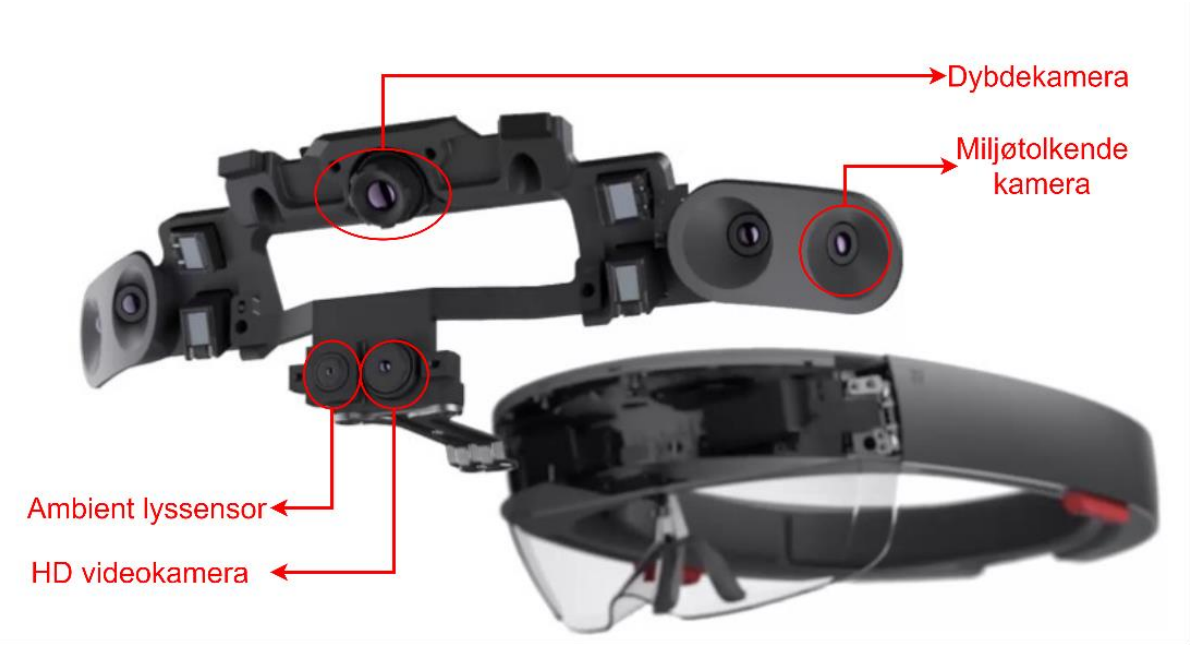
2.2 HoloLens

Figur 2 Microsoft HoloLens.

Microsoft HoloLens var verdens første Head Mounted Display (HMD) med frittstående holografisk-datamaskin (Microsoft, 2018). Brillene bruker optikk-teknologi og sensorer for å levere tredimensjonale hologrammer som tilsynelatende er plassert i verden rundt seg (Microsoft, 2018). Brillene støtter Wi-Fi, Bluetooth, har egne innebygde stereohøyttalere og har en batteritid på opptil 3 timer ved bruk (Microsoft, 2018). Ved menneske-maskin interaksjon reagerer brillene på lyd og forhåndsprogrammerte gester (Microsoft, 2018).

Figur 3 HoloLens i bruk

HoloLens brillene bruker en gjennomsiktig holografisk linse som bruker diffraktiv bølgeledningsteknologi for å kunne prosjektere hologrammer (Microsoft, 2018). Videre inneholder brillene en inertial measurement unit (IMU) bestående av akselerometer, gyroskop og et magnetometer som brukes til å spore posisjon. HoloLens bruker sine fire grå-skala miljøforsåtende kameraer sammen med det innebygde dybdekameraet til å orientere seg i verden rundt seg, samt for å forstå gester fra brukeren (Pollefeys, 2018)[1]. De ulike kameraene kan ses på Figur 4. Dybdekameraet har 120 x 120 graders field of view (FOV) og er veldig likt det man finner i Microsoft Kinect. Brillesettet har også et 2-megapixel videokamera som kan brukes til å ta bilder og film fra brukerens synspunkt. Når det kommer til prosessor, bruker HoloLens Intel sitt 32 bit brikkesett. Brillene har ikke en egen grafikkprosessor, men har en skreddersydd holografisk grafikkprosessor (HPU) designet av Microsoft. Videre har brillene 64 GB med flashminne og 2 GB RAM (Microsoft, 2018).



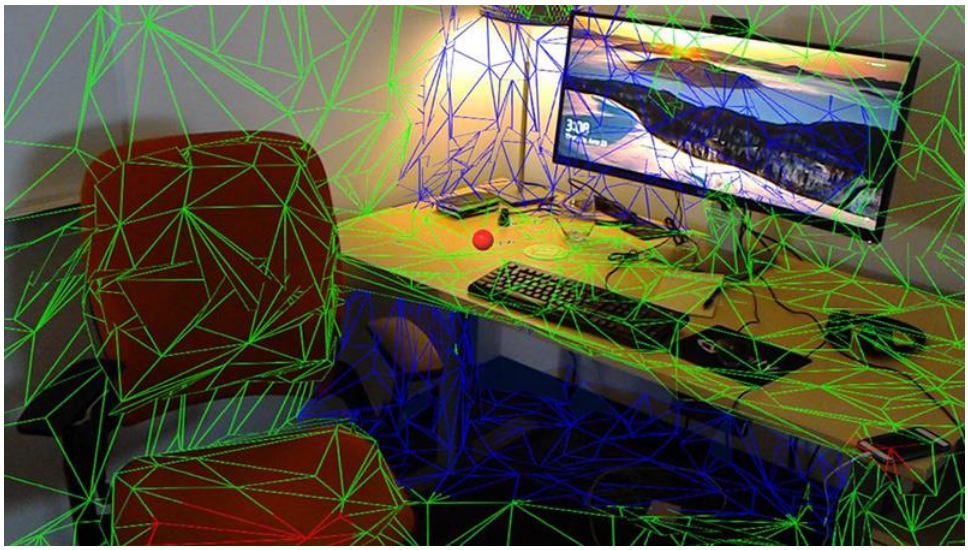
Figur 4 Kameraer og sensorer i Microsoft HoloLens.

[1] For videre lesning om hvordan HoloLens bruker sensorpakken sin til å kartlegge verden rundt seg Marc Pollefeys sin artikkel på microsoft.com. Pollefyes er for øvrig partner director of science hos Microsoft.

2.3 Mixed Reality Toolkit

Mixed reality Toolkit (MRTK), tidligere kalt HoloToolKit, består av ett sett komponenter og funksjoner som skal øke tilgjengeligheten og muligheten til å utvikle Mixed Reality apper i Unity. Kort forklart tilbyr MRTK grunnleggende byggeklosser i form av skript, modeller og teksturerer i et samlet bibliotek. Dette biblioteket kan lett importeres og implementeres i et Unity-prosjekt. Pakken har et stort rammeverk av nødvendige dokumentasjon i form av guider, bruksanvisninger og gjennomganger på hvordan man skal benytte de inkluderte ressursene. Den gjør det også enklere å feilsøke apper ved å tilby en HoloLens-emulator i unity (Microsoft, 2019). Denne emulatoren tillater utviklere å teste produktene sine uten å ha et fysisk brillesett for hånden.

2.4 Spatial Mapping



Figur 5 Spatial mapping.

Spatial mapping, eller romkartlegging, er en metode Microsoft HoloLens bruker til å danne en detaljert representasjon av verden i området rundt seg (Microsoft, 2018). Brillesettet måler avstanden mellom enheten og omverdenen med et dybdekamera, og memorerer dette. Etter hvert som omgivelsene blir kartlagt vil brillene kunne konstruere en struktur som illustrerer overflatene rundt seg, som vist i *Figur 5* (Egkarchos, 2019). Dette gjør at applikasjoner naturlig kan justeres til brukerens naturlige forventninger om den virkelige verden (Microsoft, 2018). Teknologien blir blant annet benyttet i HoloLens av dens sensorpakke for å kunne rekonstruere verden rundt seg.

2.5 Head Mounted Display

Figur 6 Head Mounted Display.

Skjermbriller (HMD) er brillesett man bruker til å fremvise informasjon i synsfeltet. Herunder skiller man mellom briller man kan se igjennom, og briller man ikke kan se gjennom. Innenfor augmented reality brukes briller som man kan se gjennom, og herunder skiller man mellom briller som bruker optisk eller digital blanding for å utvide virkeligheten. De optiske skjermbrillene bruker ulike projeksjonsteknikker for å reflektere kunstige bilder, samtidig som man kan se gjennom displayet. Innenfor fagmiljøet blir ofte «*wearable, immersive, augmented and virtual reality systems*» (WIAR) brukt for å skildre HMD-ene (Grabowski, 2015). Slike systemer lar brukeren oppleve den fysiske virkeligheten gjennom visuelle display som viser sensor og beslutningstakingsinformasjon sammen med virkeligheten (Grabowski, 2015).

2.6 Generelt om datakommunikasjon

Datakommunikasjon handler om fjernoverføring av informasjon i digital form mellom datamaskiner og brukerutstyr (Bothner-By, 2018). For at kommunikasjonen mellom ulike digitale systemer skal kunne fungere benyttes protokoller (Bothner-By, 2018). Når et felles medium skal deles, trengs det regler på hvem som skal bruke ressursen til enhver tid (Assev og Mikalsen, 2012). En protokoll defineres derfor som formater og fremgangsmåter som kreves for å få datautstyr til å kommunisere (Bothner-By, 2018). Herunder finner man regler for dataformat, sending og mottak, timing, feilsjekking med mer (Bothner-By, 2018).

Datakommunikasjon bruker ikke kun én protokoll. De ulike protokollene settes sammen i det som omtales som lag. Innenfor datakommunikasjon heter modellen som representerer de ulike protokollene og lagene for OSI-modellen (Bothner-By, 2018). Modellen er delt opp i syv lag,

hvor hver av de syv lagene har ulike kommunikasjonsoppgaver. På denne måten kan man altså si at OSI-modellen er en beskrivelse av hvordan datakommunikasjonen kan foregå.

2.7 TCP/IP kommunikasjon

TCP/IP er en gruppe kommunikasjonsprotokoller, som benyttes for å koble sammen datamaskiner i et nettverk (Johnsen og Liseter, 2017). TCP/IP kommunikasjonsprotokoll-gruppen er et resultat av advanced reasearch project agency (ARPA) sitt prosjekt på 1960-tallet, som tok for seg hvordan koble sammen datamaskiner av ulik fabrikant (Assev og Mikalsen, 2012). Prosjektet resulterte i forskningsnettverket ARPANET, og har i ettertid utviklet seg til dagens internett (Johnsen og Liseter, 2017). Kort fortalt består TCP/IP av flere ulike protokoller, der de to viktigste er TCP og IP.

2.8 Internettprotokoll

IP er en nettverksprotokoll som brukes for å formidle data mellom datamaskiner som er knyttet til samme nettverk (Dvergsdal og Ulseth, 2018). Dvergsdal og Ulseth skriver at hovedhensikten med IP er å få små nettverk til å fremstå som et enhetlig nettverk, uavhengig av hvordan de er knyttet sammen. Videre skriver de også at IP gjerne kalles for et abstraksjonslag fordi den definerer et enkelt begrepsapparat og enkle mekanismer som fritar annen programvare fra å forholde seg til en mer kompleks underliggende virkelighet (Dvergsdal og Ulseth, 2018). Oppsummert på en enkel måte kan man si at internettprotokollen bruker nettverksadresser for å bestemme hvilken vei dataene skal ta seg gjennom et nettverk for å nå mottaksadressen (Assev og Mikalsen, 2012). For videre lesning se Sigurd M. Assev og Arne B. Mikalsen sin bok Drift av lokalnettverk s. 46-57.

2.9 TCP – Transport Control Protocol

TCP overfører data mellom en tjener og en klient ved hjelp av protokolldataenheter eller blokker som kalles segmenter (Assev og Mikalsen, 2012). TCP blir ofte betegnet som en pålitelig og robust transporttjeneste på grunn av at den sørger for feilretting og at segmentene kommer i riktig rekkefølge (Assev og Mikalsen, 2012). Vi kan også si at TCP er forbindelsesorientert, noe som betyr at den sender data i en tretrinns prosess (Assev og Mikalsen, 2012). Først etableres en transportforbindelse, deretter blir dataen utvekslet før forbindelsen avsluttes (Assev og Mikalsen 2012, s. 64). Det er også naturlig å legge ved at TCP er strømming-orientert, som betyr at de to kommuniserende tjenestene (klient og tjener) skriver og leser strømmer til/fra hverandre (Assev og Mikalsen, 2012). Noe som resulterer i at en mottaker ikke vil kunne skille ut data fra to ulike segmenter (Assev og Mikalsen, 2012). Til

slutt kan man si at TCP operer i full dupleks, noe som vil si at TCP kan sende data i begge retninger samtidig (Assev og Mikalsen, 2012). For videre lesning se Sigurd. Assev og Arne B. Mikalsen sin bok Drift av lokalnettverk s. 61-64.

2.10 Node-RED og flytbasert programmering

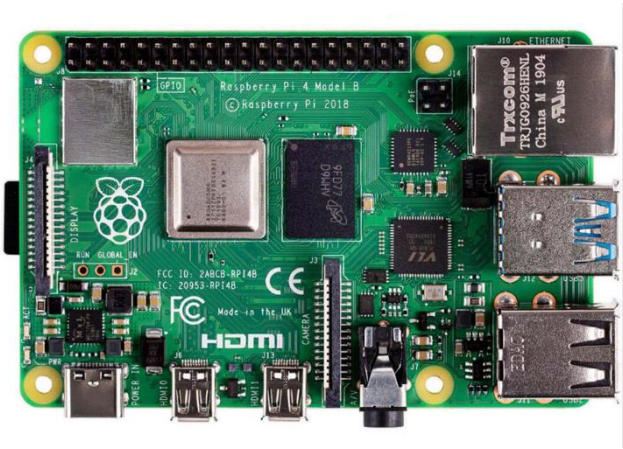
Flytbasert programmering ble utviklet av J. Paul Morrison på 1970 tallet og var et paradigmeskifte innenfor programmering (Nodered, 2019). Teknologien tar utgangspunkt i at den beskriver applikasjoner som nettverk av noder (Nodered, 2019). Nodene får data, prosesserer dataen og sender den videre. Nettverket har ansvar for flyten av data mellom nodene (Nodered, 2019). Den flytbaserte programmeringsmodellen danner grunnlaget for visuell programmering og gjør det lettere og mer tilgjengelig for flere brukere (Nodered, 2019).

Node-RED er et flytbasert utviklingsverktøy for grafisk programmering som knytter sammen maskinvare, APIs og nettverksbaserte tjenester (Lewis, 2016). Innenfor Node-RED verden er en flyt (flow) et viktig begrep. En flyt er representert som en fane inne i programmeringsverktøyet og brukes for å organisere ulike noder (Nodered, 2019). Begrepet flyt kan også bli brukt for å beskrive et enkelt sett med sammenkoblete noder (Nodered, 2019). Det neste begrepet som er viktig å definere er en node. Noder er byggeklossene til en flyt, litt som funksjoner. Noder reagerer på meldinger fra andre noder, eller ved å få input fra en ekstern hendelse, som for eksempel en HTTP-forespørsel eller en timer (Nodered, 2019).

2.11 Unity

Unity er en kryss-plattform-spillmotor utviklet av Unity Technologies og ble introdusert for første gang i 2005. Motoren gir i dag brukeren mulighet til å skape spill og andre opplevelser i både 2D, 3D, virtual reality og augmented reality. Motoren bruker programmeringsgrensesnittet (API) C# som primært programmeringsspråk for utvikling av ulike skript. I dag er Unity blitt en sentral plattform innenfor spill, film og animasjon og støtter en rekke plattformer som Windows, Android, Playstation og Linux for å nevne noen.

2.12 Raspberry Pi



Figur 7 Raspberry Pi Model 4.

Raspberry Pi er en billig, enkelbrett-pc på størrelse med et kredittkort (Raspberrypi, 2019). PC-en ble utviklet med den hensikt å spre datakunnskap til utdanningsinstitusjoner og utviklingsland (Raspberrypi, 2019). I tillegg til utdanning har Raspberry Pi store bruksområder innenfor automasjon, robotteknikk og som mediesenter. De nyeste modellene har svært kapable systemer med opptil 4gb ram, støtte for 4k oppløsning og fire kjerners prosessor med opp til 1.5 GHz klokkefrekvens (Raspberrypi, 2019).

2.13 Baudrate, databits, paritet og stop bit

For å kunne forstå NMEA-strenger kan det være relevant å gå gjennom noen sentrale begreper innen dataoverføring. Baud er en enhet for modulasjonshastighet som brukes ved overføring av digitale signaler, og angir antall sendte symboler per sekund (Myren, 2017). Baudrate må ikke misforstås med bitrate, der baudrate handler om hvor mange ganger et signal endrer per sekund, handler bit rate om hvor mange bits som kan bli sendt per tidsenhet (Myren, 2017). Databits, paritet og stop bits forklares ofte sammen ved å bruke 8-N-1 forkortelsen. 8-N-1 standarden brukes ved seriell port kommunikasjon og forteller at det er en start bit, 8 data bits, ingen paritet og en stop bit. Paritet bit eller check bit brukes til feilavkoding under dataoverføring. Stop biten symboliserer at dataoverføringen er ferdig.

2.14 NMEA-strenger

The National Marine Electronics Association (NMEA) er en ideell organisasjon bestående av produsenter, distributører, utdanningsinstitusjoner og andre interessenter i marin elektronikk (Tronico og NMEA, 2019). NMEA-standarden definerer et elektronisk grensesnitt og dataprotokoller for kommunikasjon mellom marine instrumenter (Tronico, 2019). NMEA-formatet inneholder et bredt spekter av telegrammer som benyttes basert på instrumentet (Kjerstad, 2019, 1-186). Vi skiller mellom ulike typer versjoner av NMEA-formatet, der NMEA 0183 er den eldste, men mest vanlige standarden og brukes på de fleste eldre systemer i dag (Kjerstad, 2019). NMEA 0183 standarden gir muligheten for en enkel sender, men flere lyttere på en krets (Tronico, 2019). Dette resulterer i at utstyret som støtter denne standarden som regel er produsert enten som lytter eller sender (Tronico, 2019). Videre bygger det asynkrone kommunikasjonsgrensesnittet på RS-232 standarden med følgende parametere: baudtrate:4800, 8 databits, en stopbit og ingen paritet og handshake (Kjerstad, 2019).

Den nye standarden heter NMEA 2000 og er et toveis, multi-sendende og multi-mottakende seriell-data nettverk (NMEA, 2019). Det viktigste fortrinnet sammenlignet med NMEA 0183 er muligheten til å bygge et nettverk av enheter som snakker sammen via samme kabel, samt økt kapasitet og hastighet på dataoverføring (Seatronic, 2019).

2.15 Oppbygging av NMEA-0183 datastrenger

Felles for alle NMEA-strengene er at de er bygget opp av en rekke lesbare ASCII tegn med følgende generelle struktur: $TTSSS, D1, D2,…....,Dn*CS (Kjerstad, 2019).

$ - Symboliserer starten på en setning

TT–viserinstrumenttype.

SSS – datatype.

“,” - Skille mellom datafelt.

Dn – Datafelt.

“*” - Sjekksum identifikasjon.

CS= Sjekksum.

Under vil det bli forklart hvordan de ulike NMEA-telegrammene som er blitt brukt er bygget opp.

$GPGGA

Vi har brukt GPGGA datastrenger i dette prosjektet. Viktig bemerkninger er at GP-en indikerer instrumenttypen og GP-står for GPS mottaker (GPSinformation, 2019).

GGA Global Positioning System Fix Data. Time, Position and fix related data for a GPS receiver

```
                                                  11
       1         2       3 4        5 6 7  8   9  10 |  12 13  14   15
       |         |       | |        | | |  |   |   | |   | |   |    |
$--GGA,hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh
```

Figur 8 Oppbygningen av en $GPGGA streng.

1)Tid (UTC).
2) Breddegrad.
3) Nord eller Sør.
4) Lengdegrad.
5) Øst eller Vest.
6) GPS kvalitetsindikator.
7) Antall satellitter brukt (00-12).
8) Satelittgeometri (HDOP).
9) Høyden på antennene.
10) Enhet på antennehøyde [m].
11) Geoid separasjon
12) Enhet på geoid separasjon [m]
13) Tid i sekunder siden siste SC104 type 1 eller 9 oppdatering. Null-felt dersom DGPS ikke er i bruk.
14) Referanse ID
15) Checksum

$GPVTG

VTG Track Made Good and Ground Speed

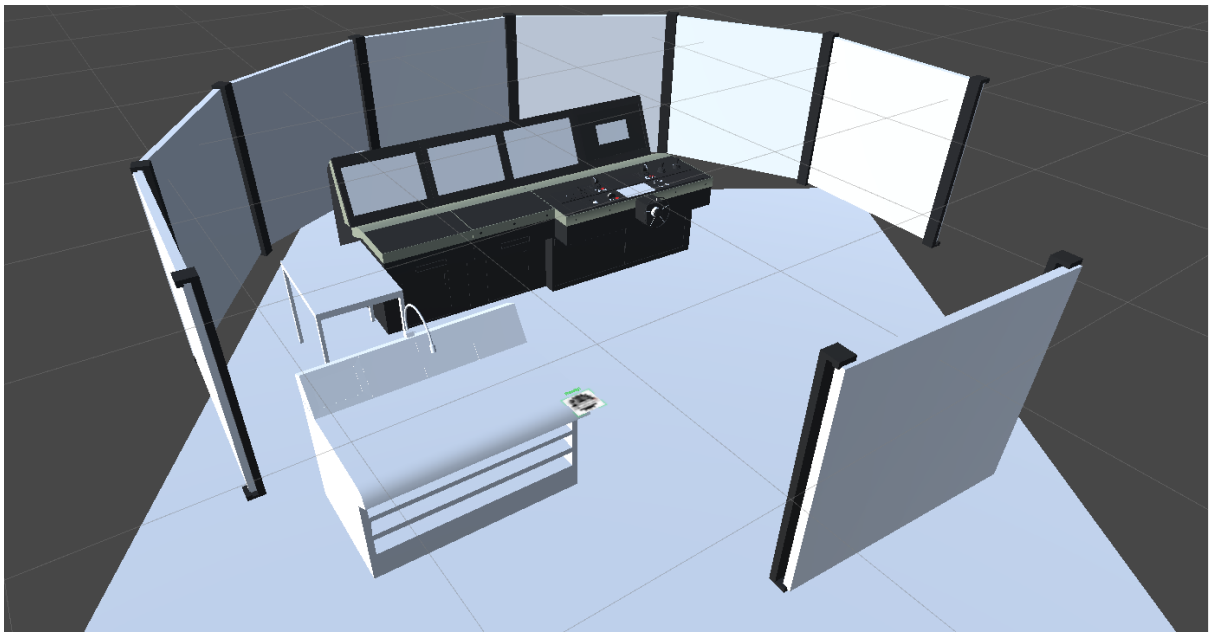
```
       1   2 3   4 5  6 7   8 9
       |   | |   | |  | |   | |
$--VTG,x.x,T,x.x,M,x.x,N,x.x,K*hh
```

Figur 9 Oppbygningen av en $GPVTG streng.

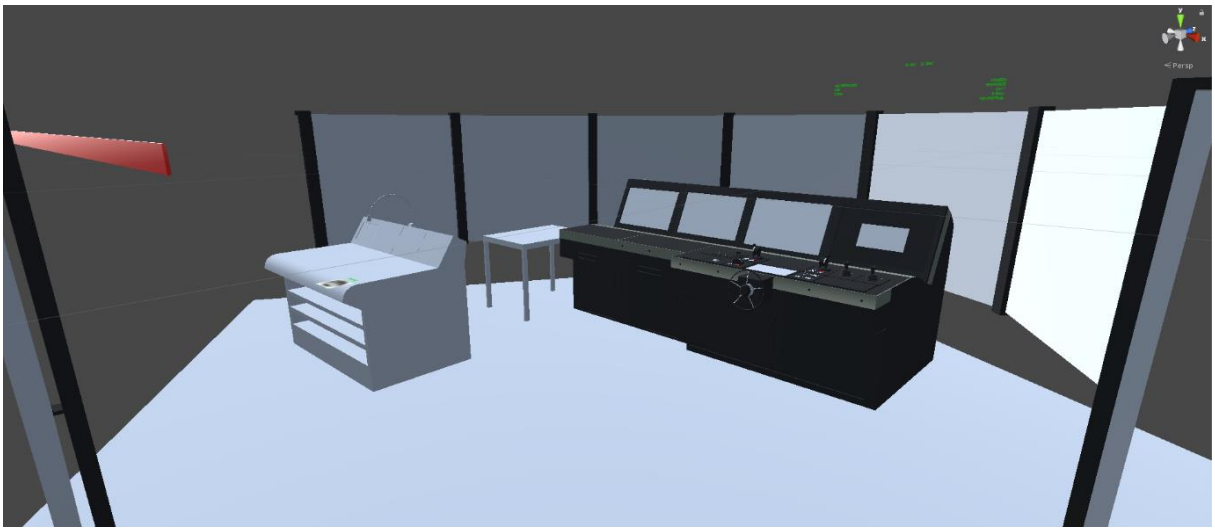
1) Course over ground
2) T = True
3) Magnetisk kurs
4) M = Magnetisk
5) Fart [Kn]
6) N = Knop
7) Fart i km/h
8) K = Kilometer per time
9) Checksum

2.16 Småbåtsimulator

Småbåtsimulatorene ved Sjøforsvarets navigasjon og kompetansesenter er levert av *Kongsberg Digital* og brukes innenfor undervisning, trening og undersøkelser. Under kan man se datamodeller og ekte bilder av småbåtsimulatorene. Datamodellene er laget i Blender, og fremvist i Unity. Simulatoren har åtte projektorer som sammen danner den virtuelle virkeligheten som man trener i. *Figur 10* og *Figur 11* viser 3D-modell av simulatoren.

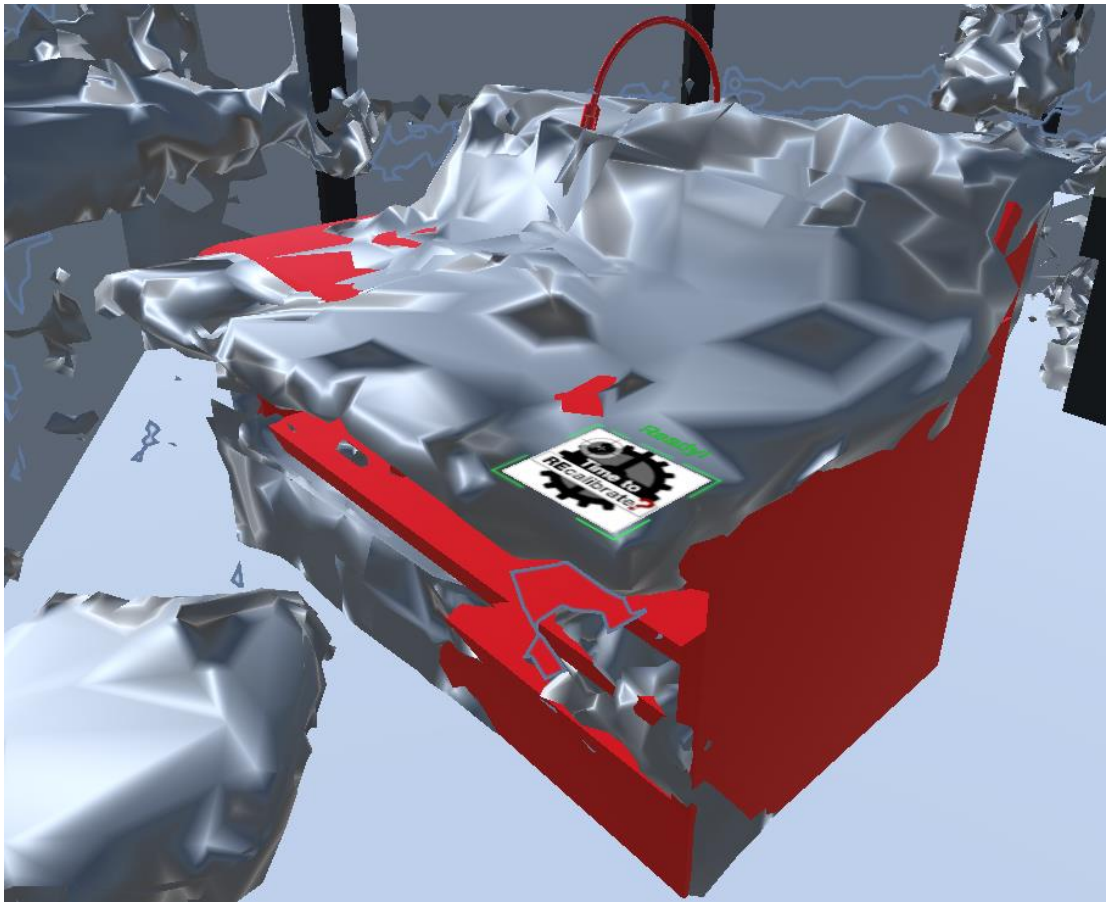


Figur 10 3D-modell av simulatorbro. Modell laget av KVM Martin Frotvedt



Figur 11 3D-modell av simulatorbro, inn dør.

Figur 12 illustrerer hvordan spatial mapping modellen samsvarer med 3D modellen. *Figur 13* kan brukes til å sammenligne modellen med virkeligheten.



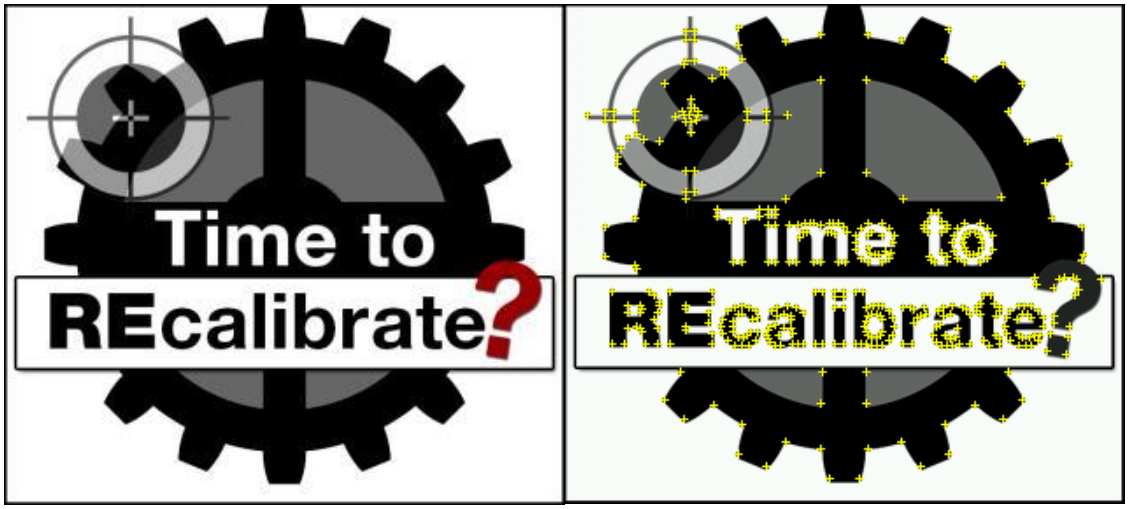
Figur 12 Spatial mapping i grått lagt over 3D-modell i rødt.

Figur 13 Bro Echo i simulatoranlegget med peilesøylen øverst i senter av bildet.

2.17 Vuforia og bildegjenkjenning

Vuforia er et Software Development Kit (SDK) spisset mot AR (libary.vuforia, 2019). Pakken inneholder en rekke ressurser, og gir brukeren mulighet for å gjenkjenne og spore både 2D bilder og 3D objekter i sanntid. Vuforia-plattformen støtter utvikling av AR apper til både IOS, Android, og universal windows platform (libary.vuforia, 2019).

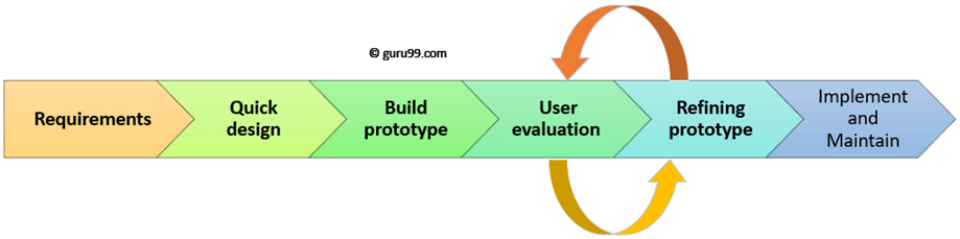
Bildemål (*engelsk: image targets)* er bilder som Vuforia-motoren kan oppdage og detektere (libary.vuforia, 2019). I kontrast med QR-koder og data matriser, trenger ikke bildemål spesielle kombinasjoner av svart og hvite områder for at koden skal bli gjenkjent (libary.vuforia, 2019). Vuforia-motoren detekterer og følger kjennetegnene som finnes naturlig i bildet (libary.vuforia, 2019). Dette gjøres ved at motoren sammenligner disse naturlige kjennetegnene opp mot en database som inneholder bildemålets kjennetegn. Når bildemålet er oppdaget, vil Vuforia motoren registrere bildets posisjon så lenge det er delvis i kameraet sitt synsbredde (libary.vuforia, 2019). *Figur 14* illustrerer hvordan Vuforia gjenkjenner bilde ved hjelp av kjennetegn.



Figur 14 Bildemål for kalibrering. (Tannhjul). Viser hvordan Vuforia gjenkjenner bildet.

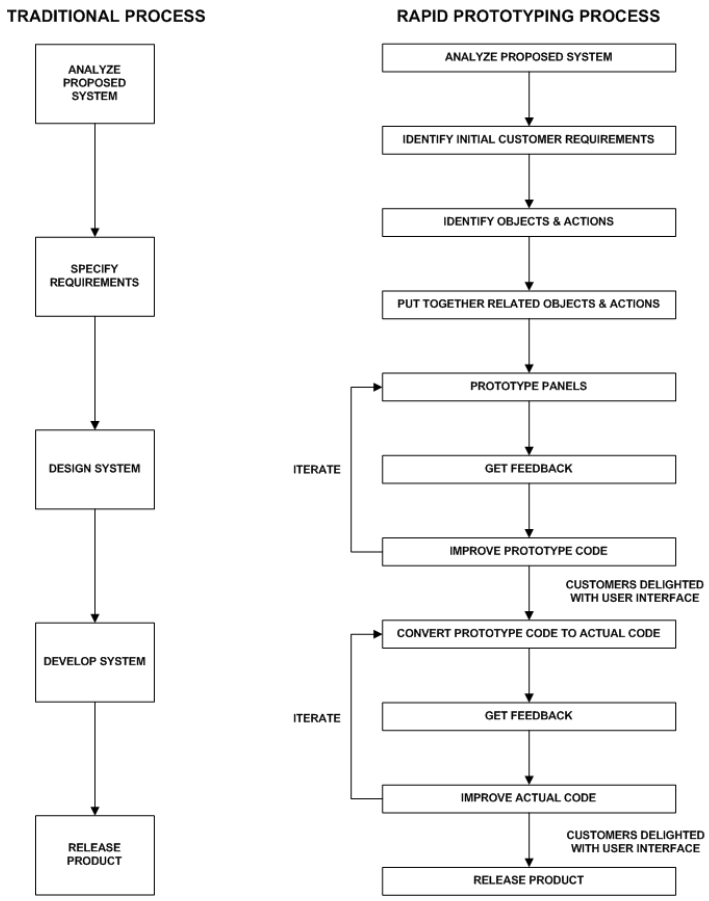
3 Metode

3.1 Den videreutviklende prototypemodellen



Figur 15 Prototype under utvikling.

For å utvikle prototypen i dette prosjektet har vi brukt en prototypemodell for utviklingen av systemet. Felles for alle prototypemodeller er at de bygger på mange av de samme prinsippene. Det starter med en krav-funksjonsanalyse, som spesifiserer de ulike funksjonene og kravene som prototypen må ha. Her er det naturlig å inkludere de som skal bruke systemet, slik at det endelige produktet samsvarer med deres ønsker og forventninger. I noen utviklingsprosesser kan det være gunstig å utvikle en grovskisse eller modell før man starter utviklingen av prototypen. Dette gir brukeren og utvikleren felles mentale modeller før selve utviklingen iverksettes. Det neste er selve utviklingen av prototypen. Herunder vil det være ulike strategier basert på om man utvikler software, jobber med elektronikk eller driver med metallarbeid. Deretter kommer brukerevalueringen. Her kan en relevant målgruppe teste produktet i den hensikt å avdekke eventuelle feil, mangler og andre ting som må fikses før en eventuell lansering eller implementering.



Figur 16 Forskjellen på den tradisjonelle- og den rapide prototypeutviklingsprosessen.

Den spesifikke modellen vi har valgt å bruke heter videreutviklende rapide prototypemodell. På engelsk heter den "evolutionary rapid prototyping model", men i mangel på gode oversettelser, velger vi å omtale den som videreutviklende rapide prototypingsmodell. Videre skiller den seg fra den tradisjonelle prototypeprosessen ved at den:

1. Involverer brukeren/kunden i en mye større grad (Sadabadi, 2009). Dette gjennom inkludering av brukeren både i startfasen, men også gjennom brukertester og kontinuerlige tilbakemeldinger (Sadabadi, 2009).
2. Bygger på tanken om at man bygger prototypen på en strukturert måte og kontinuerlig ombygger og forbedrer den gjennom prosjektet (Sadabadi, 2009). På den måten kan man si modellen erkjenner det faktum at vi ikke forstår alle kravene, funksjonene og utfordringene med en gang og gir utviklergruppen mulighet til å endre på kravene og funksjonene gjennom utviklingsprosessen (Sadabadi, 2009).

3.2 Heuristisk evaluering av brukergrensesnitt

Gjennom dette delkapittelet vil ordene testperson og observatør bli brukt gjentatte ganger. En testperson er den som tester et brukergrensesnitt og er valgt ut på bakgrunn av sin kompetanse. En observatør er en som er med på å holde/arrangere testen.

Fra forrige delkapittel ble det forklart at en evaluering fra brukermassen er en sentral del av utviklingen av et bra produkt. Heuristisk evaluering er en metode innenfor *usability engineering*[2] for å finne brukerproblemer i design av brukergrensesnitt, slik at de kan bli fikset i utviklingsprosessen (Nielsen, 1994). Den heuristisk evaluering handler om la en liten gruppe med evaluatorer bedømme brukergrensesnittet opp mot etablerte brukervennlighetsprinsipper (Nielsen, 1994). Et viktig prinsipp innenfor heuristisk evaluering er at det er vanskelig for en enkel person å identifisere alle brukervennlighetsproblemene til et brukergrensesnitt alene (Nielsen, 1994). Derfor benyttes flere testpersoner for å forbedre effektiviteten og troverdigheten til metoden (Nielsen, 1994). Testene gjennomføres ved at hver testperson tester brukergrensesnittet alene. Etter evalueringen har blitt gjennomført, kan testpersonen kommunisere med observatørene (Nielsen, 1994). Det er viktig å gjennomføre denne prosessen på en riktig måte for å tilstrebe uavhengige og objektive evalueringer fra hver testperson (Nielsen, 1994). Til tross for dette kan observatøren assistere dersom testpersonene har begrenset teknisk kompetanse og trenger å få visse deler av brukergrensesnittet forklart (Nielsen, 1994). Dette resulterer i at testpersoner uten stor kjennskap til brukergrensesnittet kan bidra til testingen, noe om igjen vil kunne styrke troverdigheten til testen (Nielsen 1994).

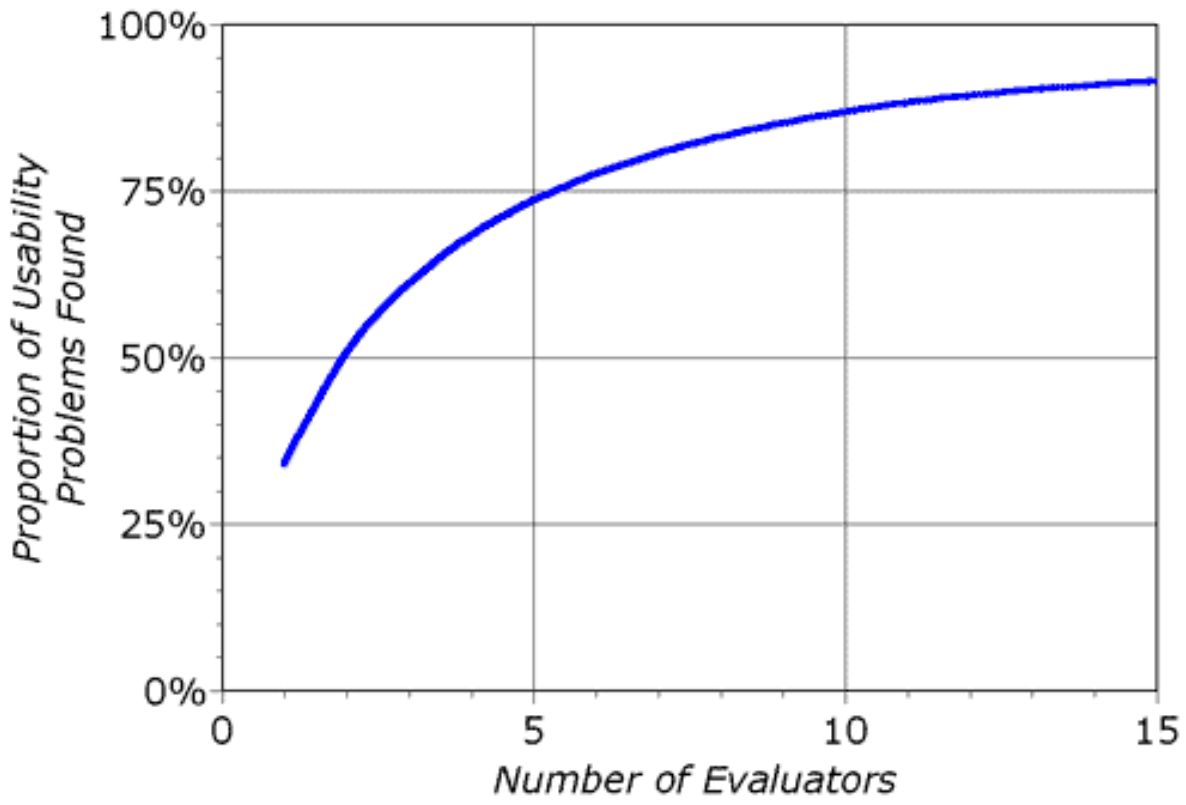
Når det kommer til hvordan besvarelsen av evalueringen skal utformes, skal det være rom for utfyllende svar med referanser til brukervennlighetsprinsippene (Nielsen, 1994). Det er på denne måten viktig at besvarelsen fra testpersonen er så spesifikk som mulig, samt at vedkommende adresserer brukervennlighetsproblemene separat (Nielsen, 1994). Dette er det to grunner til. Den første grunnen er at det kan skape uklarhet dersom det er en rekke problemer som blir presentert samtidig (Nielsen, 1994). Den andre grunnen er at det ikke er sikkert at alle brukervennlighetsproblemene som blir adressert kan fikes. På den måten er det mer oversiktlig om problemene blir presentert separat.

[2] Usability engineering er et fagfelt som spesialiserer seg innenfor menneske-datamaskin interaksjon, og har stort fokus på brukervennlighet.

Et annet viktig aspekt vedrørende en heuristisk evaluering er antall testobservatorer. I utgangspunktet kan en enkel testperson gjennomføre en heuristisk evaluering alene, men tester har vist at dette som regel resulterer i dårlige resultater (Nielsen, 1994). Nielsen og Landauer presenterte i 1993 en modell basert på følgende formel:

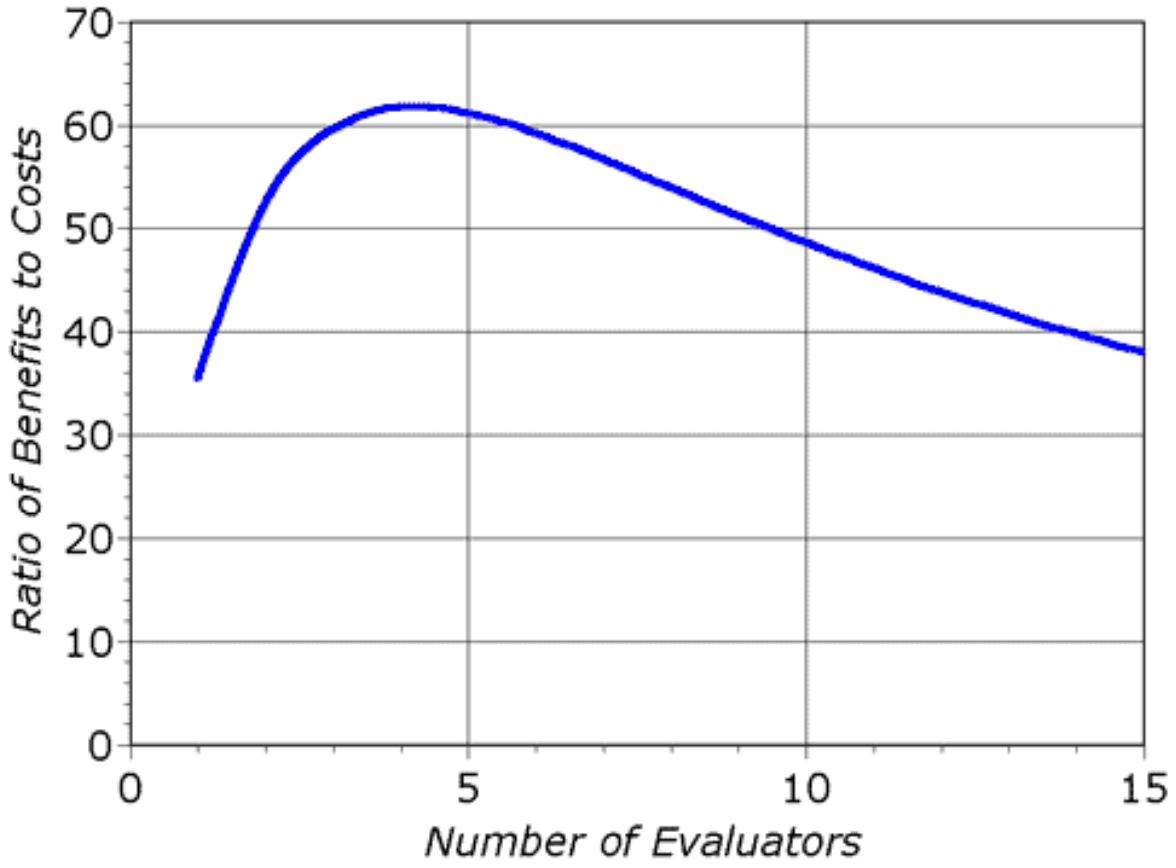
$$\text{ProblemerFunnet}(n) = N\,(1 - (1-L)^{n})$$

Hvor n representerer antallet uavhengige testpersoner, N representerer det totale antallet brukervennlighetsproblemer som brukergrensesnittet har og L representerer prosentandelen av feil en enkel testperson kan finne under testen (Nielsen og Landauer, 1993). Verdien L varierte fra 19-51 prosent, med et gjennomsnitt på 35 prosent (Nielsen, 1994). Verdien for N varierte fra 16-50, med et gjennomsnitt på 33 (Nielsen, 1994). Resultat fra prediksjonsformelen til Nielsen og Landauer ble følgende:



Figur 17 Korrelasjon mellom antall testpersoner og andel feil funnet.

Det er ikke nok å bare bruke en slik graf for å beregne det optimale antallet av testpersoner. Man trenger også en kost-/nytte-modell for den heuristiske evalueringen. Nielsen og Landauer lagde også en kost-/nytte-modell basert på et prøveprosjekt (Nielsen, 1994). Modellen tok for seg både faste og variable utgifter. Utfra dette kom de frem til en modell som viste kost-/nytte-fordelene basert på antall testpersoner (Nielsen, 1994). Modell så slik ut:



Figur 18 Korrelasjon mellom antall testpersoner og kost/nytte.

Basert på resultatene til Nielsen og Landauer i *Figur 18* kan man se det optimale antallet testpersoner, som i dette tilfellet er fire dersom man vurderer etter kosteffektivitet. Slår man sammen resultatene fra *Figur 17* og *Figur 18* kan man konkludere med at man kan være tilstrekkelig fornøyd med tre-fem testpersoner i en heuristisk evaluering.

3.3 Utvikling ved hjelp a prototypemodell

Gjennom dette prosjektet har vi brukt den utviklende rapide prototypingsmodellen for å utvikle vår prototype. Vi startet med å analysere bestillingen som ble gitt fra nav-komp. Her ble funksjoner, design og generelle ønsker fra den aktuelle brukeren undersøkt. Deretter undersøkte vi hva som var mulig med tanke på ressurser, kompetanse og tid, før vi kom fram til hvilke funksjoner og krav som var realiserbare, med tanke på brukerens ønsker og våre begrensinger.

Det neste steget var i utgangspunktet at man skulle lage en grov modell av den aktuelle prototypen. Deretter å innhente tilbakemeldinger fra brukermassen, før man starter på selve utviklingen av prototypen. På bakgrunn av tidsrammen og omfanget av bacheloroppgaven har vi valgt å hoppe over dette steget, dog det kan være svært relevant for større ressurskrevende prosjekter. Det ble naturlig for oss å finne ut hvilke verktøy vi hadde tilgjengelig før vi introduserte brukernes tilbakemeldinger. Etter å ha presentert data hentet fra simulatoren i brillesettet, startet selve utviklingen av prototypen. Utviklingen var preget av den prototypemodellen, og det ble kontinuerlig utviklet nye versjoner av den daværende prototypen. Samtidig var funksjons- og kravanalysen kontinuerlig den overordnete retningslinjen for vår design og utvikling.

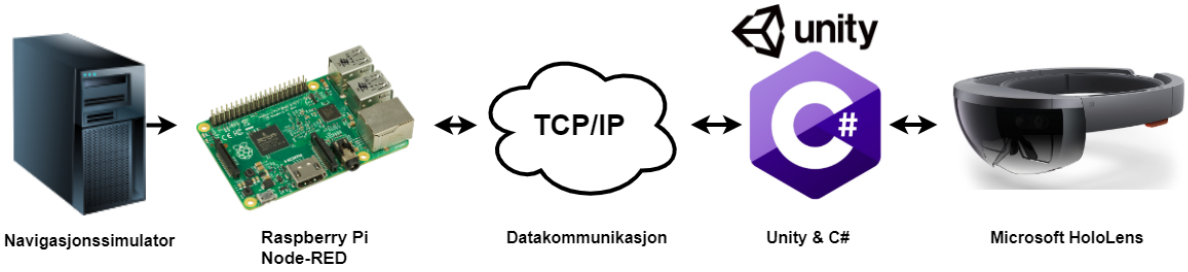
3.4 Gjennomføring av heuristisk evaluering

Evalueringene startet med at kandidatene fylte ut et samtykkeskjema på forhånd. Her fikk testpersonene forklart hva som var målene med testen, hvordan testen skulle gjennomføres og hvilke rettigheter de hadde med tanke på personvern og deres deltakelse. Deretter må testpersonene fylle ut hva de samtykker til, før de signerer skjemaet. For utfyllende informasjon av skjemaet se *vedlegg A.*

Kandidatene ble deretter tatt med inn navigasjonssimulatoren ved nav-komp. De ble så forklart de ulike funksjonene og kommandoene som systemet har, før sentrale tekniske detaljer ble forklart. Deretter fikk testpersonene prøve systemet samtidig som de navigerte en og samme forhåndsplanlagt rute. Denne hadde vi på forhånd hadde valgt ut på bakgrunn av sin variasjon. Dette gjorde at navigatøren fikk testet brillene i ulike situasjoner. Dersom det oppstod noen tekniske utfordringer som det ikke var forlangt at testpersonene skulle kunne løse selv, fikk testpersonen hjelp. Utenom dette ble testpersonene i stor grad uforstyrret under testen. Samtidig som testpersonene holdt på ble det tatt bilder og notert observasjoner for videre bruk i rapporten. Etter selve uttestingen av systemet var ferdig, ble kandidatene spurt om å fullføre spørreskjemaet som er vedlagt som *vedlegg B*. Etter at spørreskjemaene var utfylt ble det holdt et utviklingsmøte med bachelorgruppen for å diskutere testen og de resultatene som ble levert. Resultatene ble deretter lagret ført digitalt og lagret for videre bruk.

4 Implementering og realisering av AR-systemet

I dette kapittelet vil det bli dokumentert hvordan systemet er satt sammen. Kapittelet tar fokus på det overordnede systemet og hvordan ting er satt sammen, fremfor å være veldig teknisk detaljert. Dette resulterer i at kapittelet ikke vil gå i dybden på tekniske detaljer. For en mer teknisk innføring anbefales det å se *vedlegg O* for utfyllende kommentarer.



Figur 19 Dataens vei fra simulator til brillesettet.

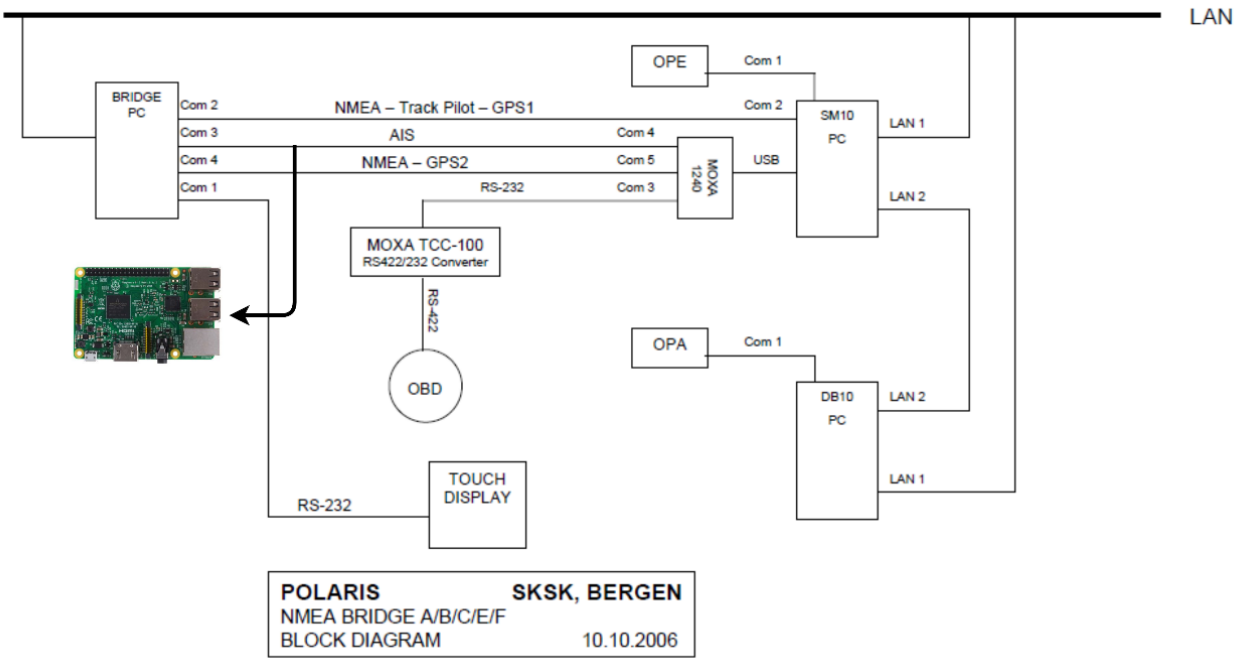
Figur 19 gir et grovt oversiktsbilde av hvordan systemet er satt opp. Dersom man leser fra venstre, kan man se at det hele starter ved navigasjonssimulatoren. Dataen blir lest av en Raspberry pi, og behandlet i Node-RED. Den blir deretter sendt ved hjelp av socket-kommunikasjon til brillesettet. Mottak og visning av data blir håndtert av de ulike C#-skriptene, mens spillmotoren Unity styrer hvordan dataen blir vist i brillesettet.

4.1 Datauthenting

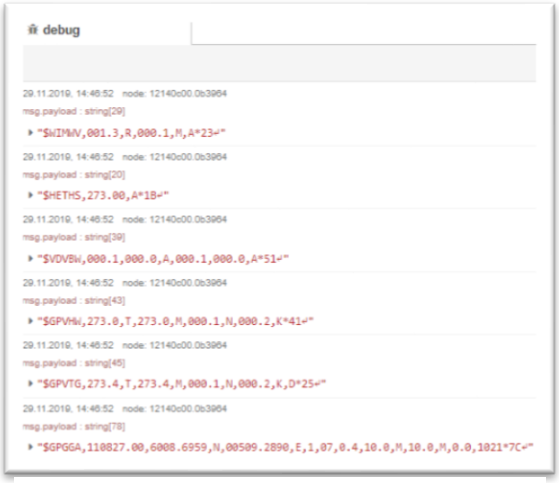
Datauthenting blir gjort ved å koble seg på simulatorens COM port 3 som vi kan se på og *Figur 20.* I utgangspunktet blir denne porten brukt til å sende automatic identification system (AIS) datastrenger til SeaCross-systemet. Ved hjelp fra *Kongsberg Digital* fikk vi rekonfigureret[3] utgangen slik at den sender relevant informasjon i henhold til vår kravspesifikasjon, med en høyre frekvens enn det var i utgangspunktet. viser et flyt-diagram av simulatoranlegget. Det er viktig å merke seg at Raspberry Pien er tilkoblet Com 3 kabelen som kommer fra BRIDGE PC, samt at det trengs en RS232-adapter for å koble kabelen til USB-porten på Pien. illustrerer dette. Dataene som kommer ut kommer i tekststreng format, med NMEA-standard oppbygging. *Figur 21* viser hvordan rå-dataen kommer ut fra simulatoren i Node-RED.

.

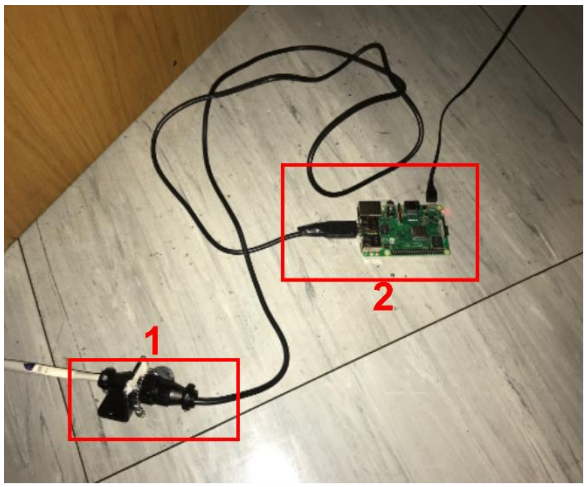
[3] Fra AIS-strenger til NMEA-strenger, se *vedlegg Q for konfigurasjon.*



Figur 22 Flytdiagram av Polaris. Raspberry Pi koblet til Com 3.

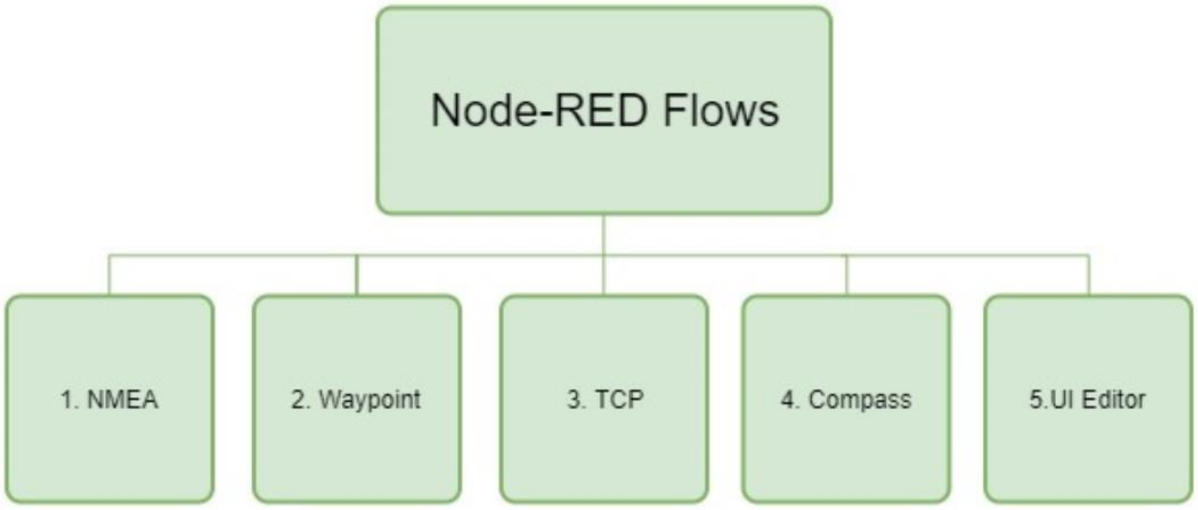


Figur 21 avlesning av NMEA-strenger i Node-RED.



Figur 20. 1: overgang fra NMEA-bus. 2: Raspberry Pi leser av data.

4.2 Node-RED

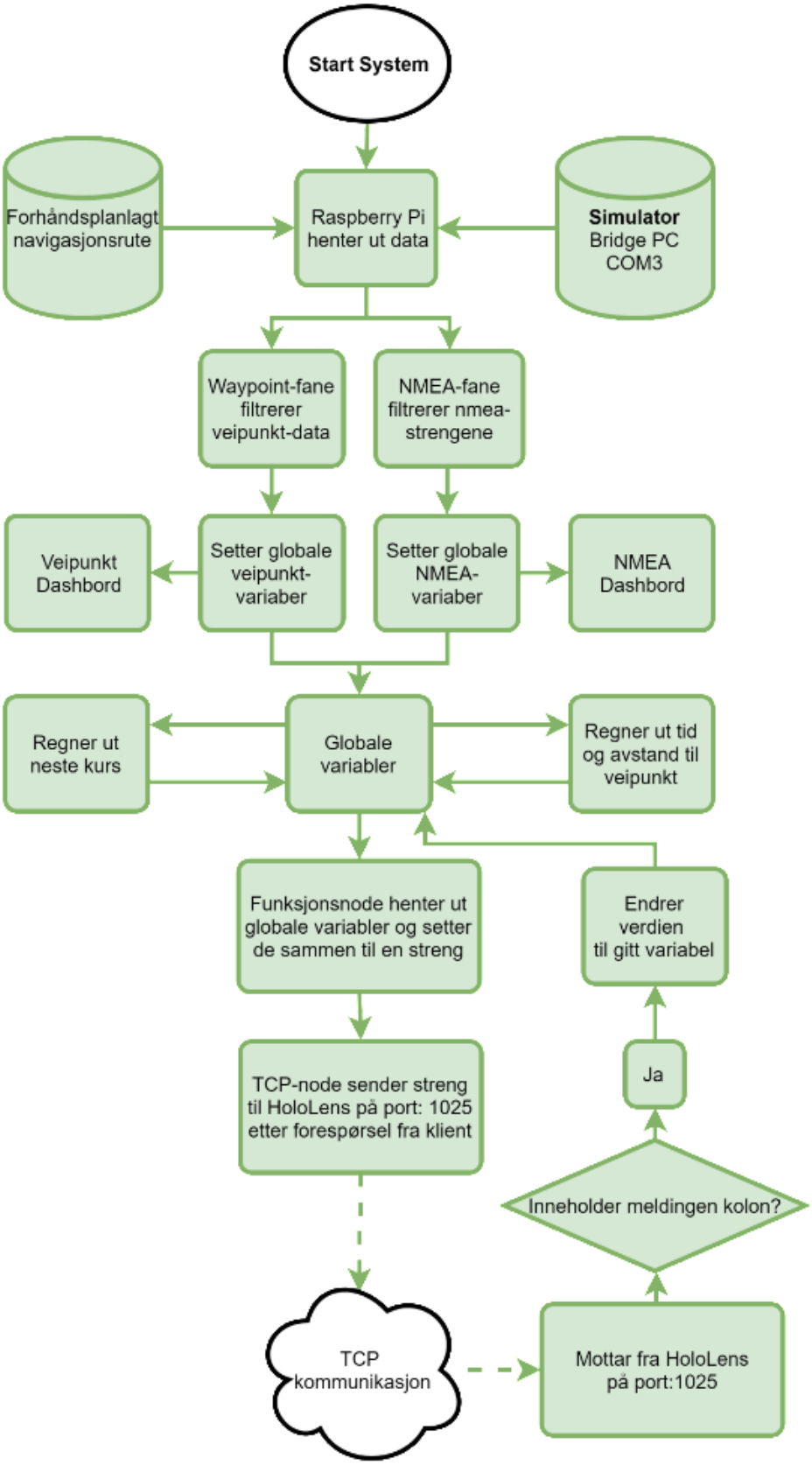


Figur 23 Node-RED struktur.

Figur 23 viser de ulike fanene til Node-RED. Utfra dette kan vi se at Node-RED i hovedsak har fem funksjoner.

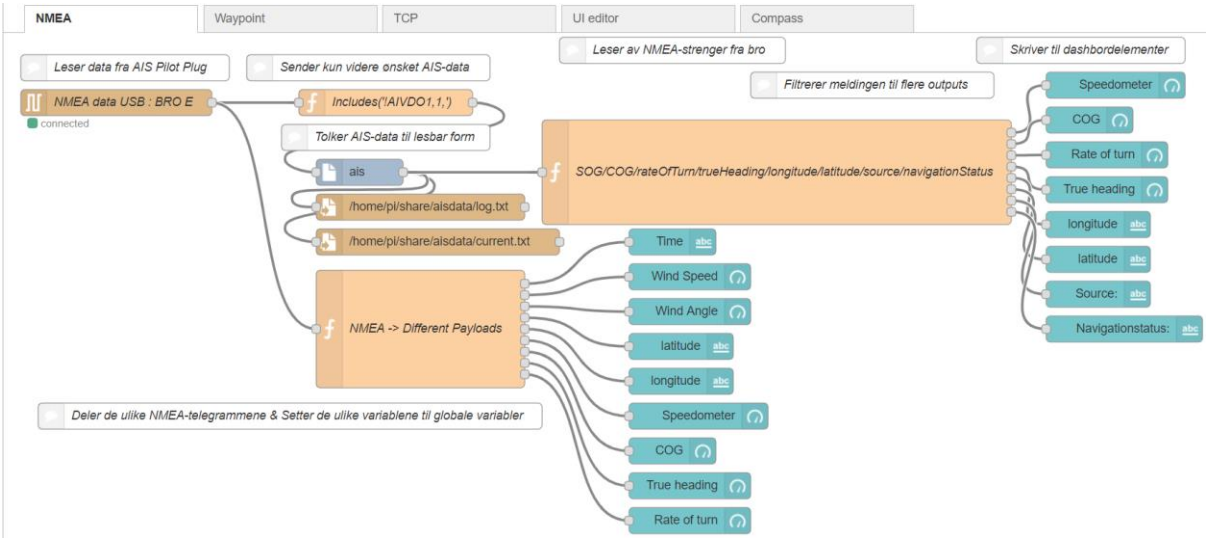
1. NMEA-fanen er ansvarlig for å lese av data-strenger fra simulatoranlegget. Deretter blir dataen filtrert, overført til globale variabler, og vist i Node-RED sitt dashbord (som webside).
2. Waypoint-fanen er ansvarlig for å lese av navigasjonsrutene. Fanen filtrerer og lagrer rutens informasjon i globale variabler. Til slutt blir disse dataene vist i Node-RED-dashbordet. Fanen regner også ut avstand og estimere tid til neste veipunkt, samt kursen som skal holdes. Her styres også valgt veipunkt.
3. TCP-fanen er i praksis systemets server, og står for all ekstern kommunikasjon. Den har ansvaret for å sende data til brillesettet (les: klienten) etter forespørsel.
4. Compass-fanen er ansvarlig for avlesning av data fra kompassbrikken. Den gir retningen til brillesettet. Den viser også retning i Node-RED-dashbordet.
5. UI editor-fanen er ansvarlig for å utføre eventuelle brukerstyrte endringer på brukergrensesnittet. Det er mulig å tilpasse farge, tekststørrelse og plassering i rommet.

I *Figur 24* kan man se et flyt-diagram av Node-RED delen av systemet. Diagrammet vil bli forklart i detalj i påfølgende avsnitt.



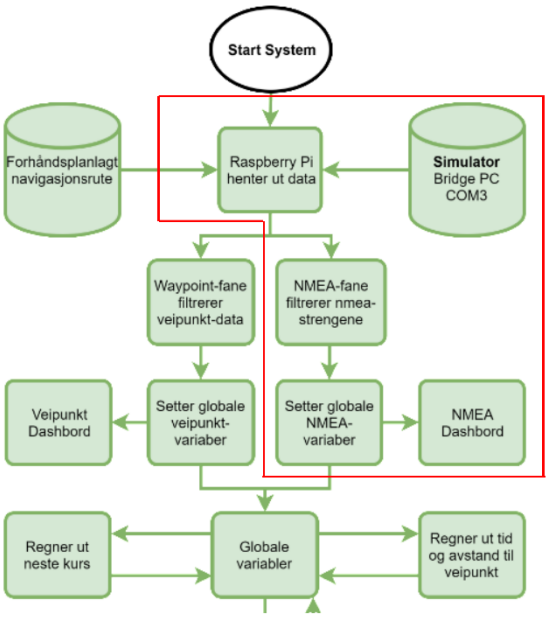
Figur 24 Flytdiagram over Node-Red delen.

4.2.1 NMEA-fane



Figur 25 NMEA-fanen. NMEA-informasjon leses av som seriell input på venstre siden. Forskjellige verdier vises grafisk til brukeren på en webside via de turkise nodene. Noen av NMEA-verdien lagres i globale variabler i funksjonsnodene i midten.

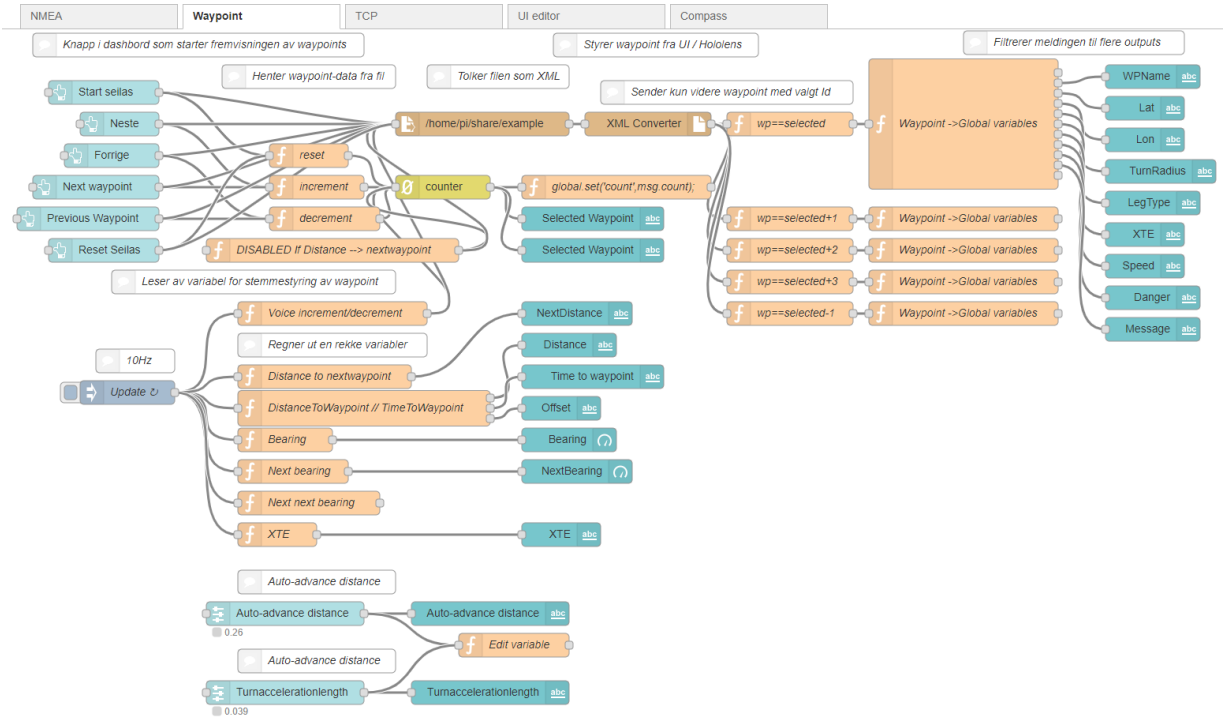
Figur 25 er et skjermbilde av fanen NMEA i Node-RED. Figuren leses venstre mot høyre, og man ser at dataen blir behandlet på en systematisk måte. Dataen blir først lest av NMEA-data fra USB noden og deretter sendt til to funksjoner. NMEA til forskjellige «*payload*»-funksjonen filtrerer relevant data fra NMEA-strengene. Disse verdiene blir lagret i globale variabler som kan brukes senere i andre sub-flows. Funksjonsnoden har flere outputs slik at man kan koble på ønsket dashbord-element for å presentere dataen. NMEA-fanen samsvarer med den markerte delen av flytdiagrammet som vist på Figur 26.



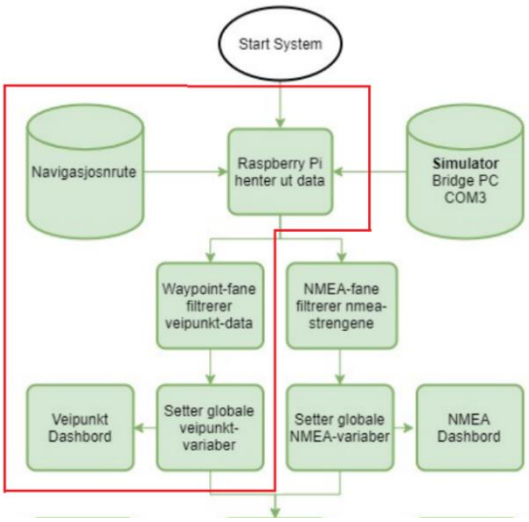
Figur 26 Flytdiagram av NMEA-fane.

Prosjektet startet med avlesning av AIS-strenger. Vi gikk vekk fra disse dataene, senere beskrevet i *kapittel 6.6*. AIS-avlesningen ligger fortsatt som en mulighet, men dens data vil ikke bli brukt av HoloLens. AIS-delen av NMEA-fanen er ikke i bruk nå. Den kan dog brukes dersom man ønsker å benytte AIS-strenger som datakilde. Kun ønskede AIS-strenger blir sendt videre. AIS-delen krevde dog en forhåndsprogrammert node for å tolke de noe mer kryptiske AIS-strengene.

4.2.2 Waypoint



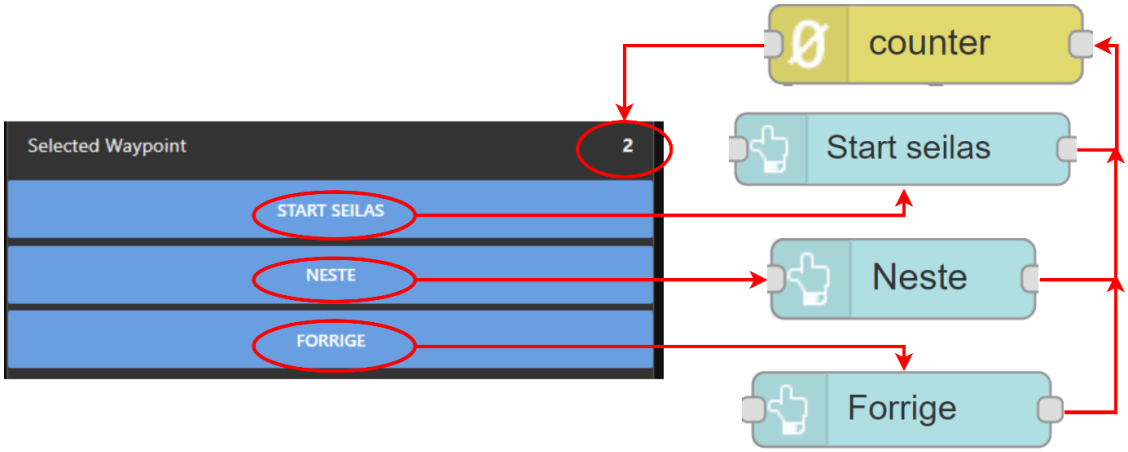
Figur 27 Waypoint-fanen.



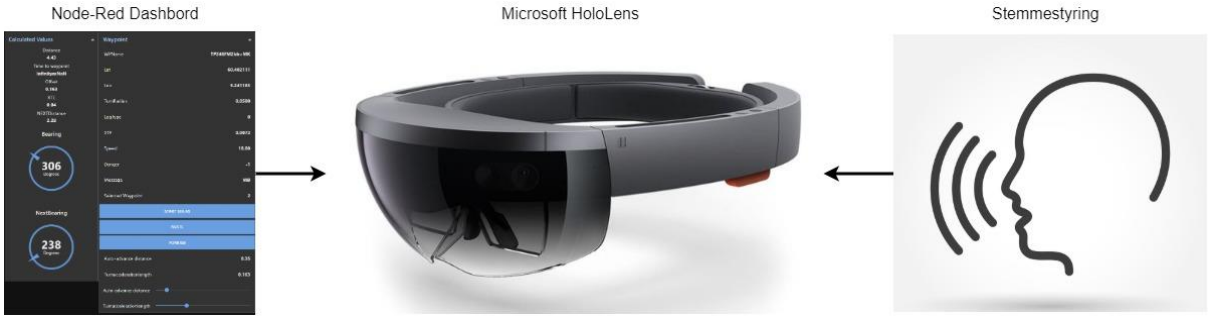
Figur 28 sammenheng mellom WP-fanen og markert del av flytdiagrammet

Midtpunktet av fanen som er vist i *Figur 27* er filnoden som i dette tilfelle heter /home/pi/share/example og er en referanse til filbanen hvor man finner rutefilen. Filen må legges inn manuelt enten ved bruk av minnepenn, VNC Viewer eller Samba. XML-konverteringsnoden tolker et XML-formatert dokument, og setter verdiene til ulike payloads. Funksjonsnoden skriver disse verdiene til globale variabler. Datapunkter fra inneværende veipunkt blir vist i dashbordet (websiden). Den gule noden er en telle-node som holder tellingen på valgt veipunkt. Det valgte veipunktet blir sendt videre av en rekke andre veipunkter. Eksempelvis dersom man har trykket på knappen «*neste veipunkt*» tre ganger vil telleren vise tre, og gi ut veipunkt nummer tre. *Figur 29* illustrerer sammenhengen mellom nodene og dashbordet. Denne telleren styres av Node-RED dashbord knapper eller ved hjelp av stemmestyring i HoloLens som vi kommer tilbake til

senere. *Figur 30* viser hvordan man kan interagere med systemet ved hjelp av både stemmestyring og Node-Red dashbord.



Figur 29 Sammenhengen mellom nodene og dashbordet for å skifte veipunkt.



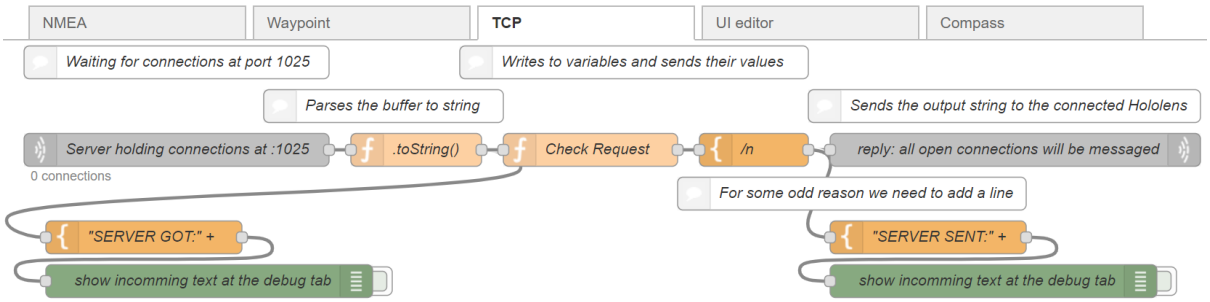
Figur 30 Hvordan man kan interagere med Microsoft HoloLens.

De andre viktige funksjonene som kan nevens: avstand/tid til veipunkt og inneværende, så vel som neste kurs.

- Avstand/tid funksjonen regner ut avstanden til neste veipunkt ut ifra posisjonen til fartøyet. Den gjør dette ved å sammenligne posisjonen til fartøyet med neste veipunkt. Den bruker deretter en vei-fart-tid-formel til å gi et tidsestimat. Den vil her bare benytte seg av fartøyets fart, og tar ikke hensyn til retning.
- Kursene regnes ved bruk av trigonometri. Vi bruker posisjonene til to veipunkter som skal seiles imellom.

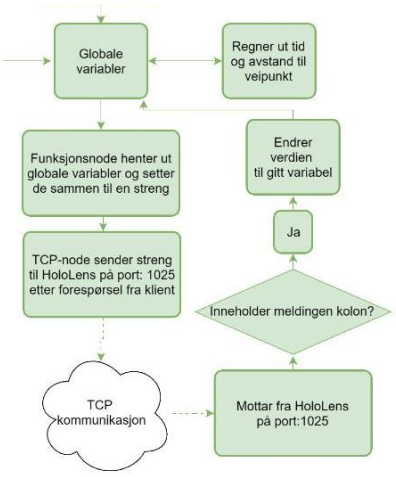
Veipunkt-fanen samsvarer dermed med den markerte delen i *Figur 28.*

4.2.3 TCP



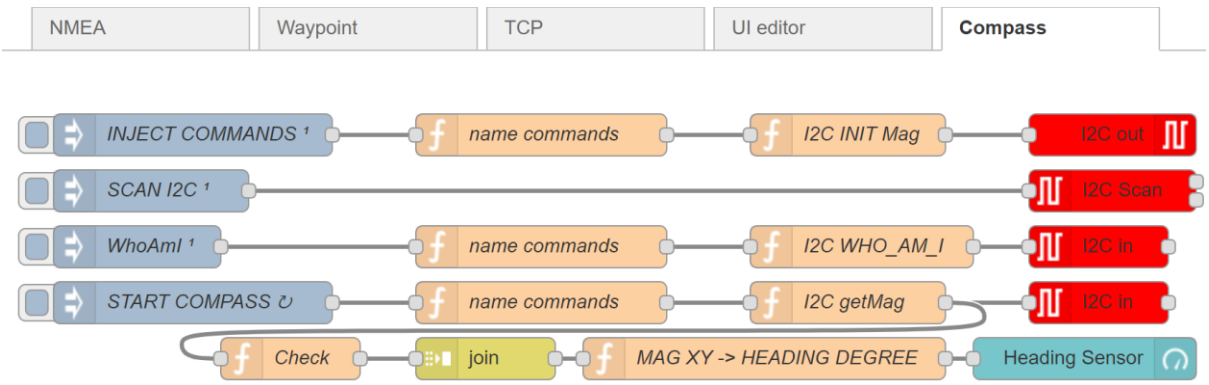
Figur 31 TCP-fanen.

TCP-fanen brukes som server for systemet. Den består hovedsakelig av to TCP-noder, hvor den til venstre lytter etter innkommende meldinger fra brillesettet, og den til høyre sender data på forespørsel fra klient. Innkommende melding blir oversatt fra buffer til tekststreng, ser over meldingen som har blitt mottatt og sender en oppdatert variabelliste i strengformat tilbake til brillesettet. All socket-kommunikasjon skjer på port 1025 hos serveren.



Figur 32 Flytdiagram av TCP-fanen.

4.2.4 Compass

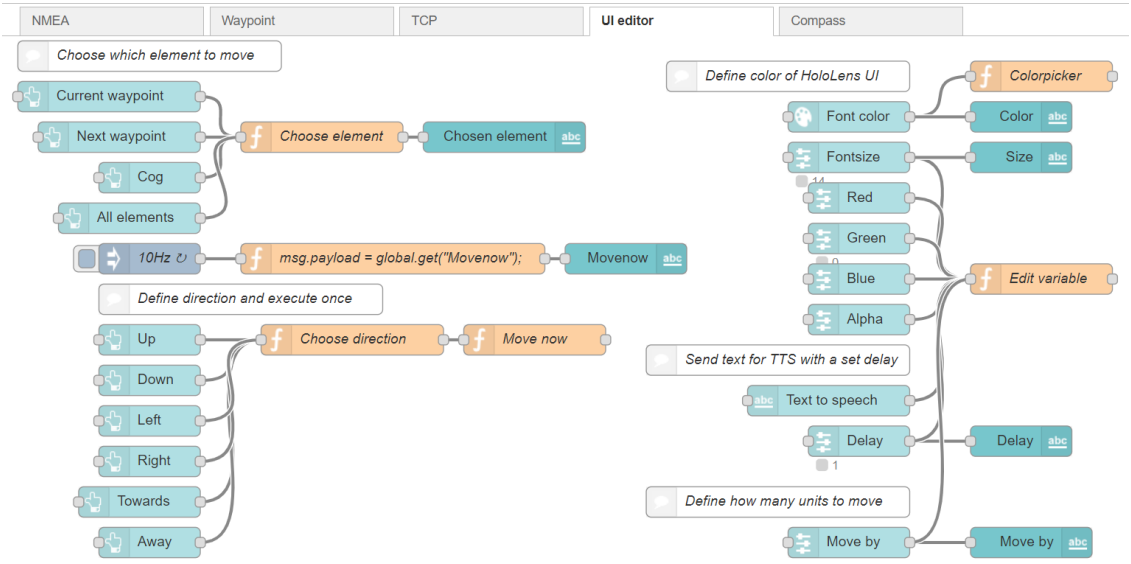


Figur 33 Compass-fanen.

Compass-fanen består av fire linjer. De tre første linjene sørger for at kompass blir stilt inn med rett magnetisk skala og output frekvens. Selve innhenting av sensor-data skjer i den siste linjen. I2C-in noden henter sensor-data fra sensoren. Deretter blir x-, y- og z-komponenten av den magnetiske fluksen satt sammen i en felles melding. Til slutt vil funksjonsnoden regne ut headingen til sensoren ved bruk av trigonometri og presentere den i brukergrensesnittet.

4.2.5 UI Editor

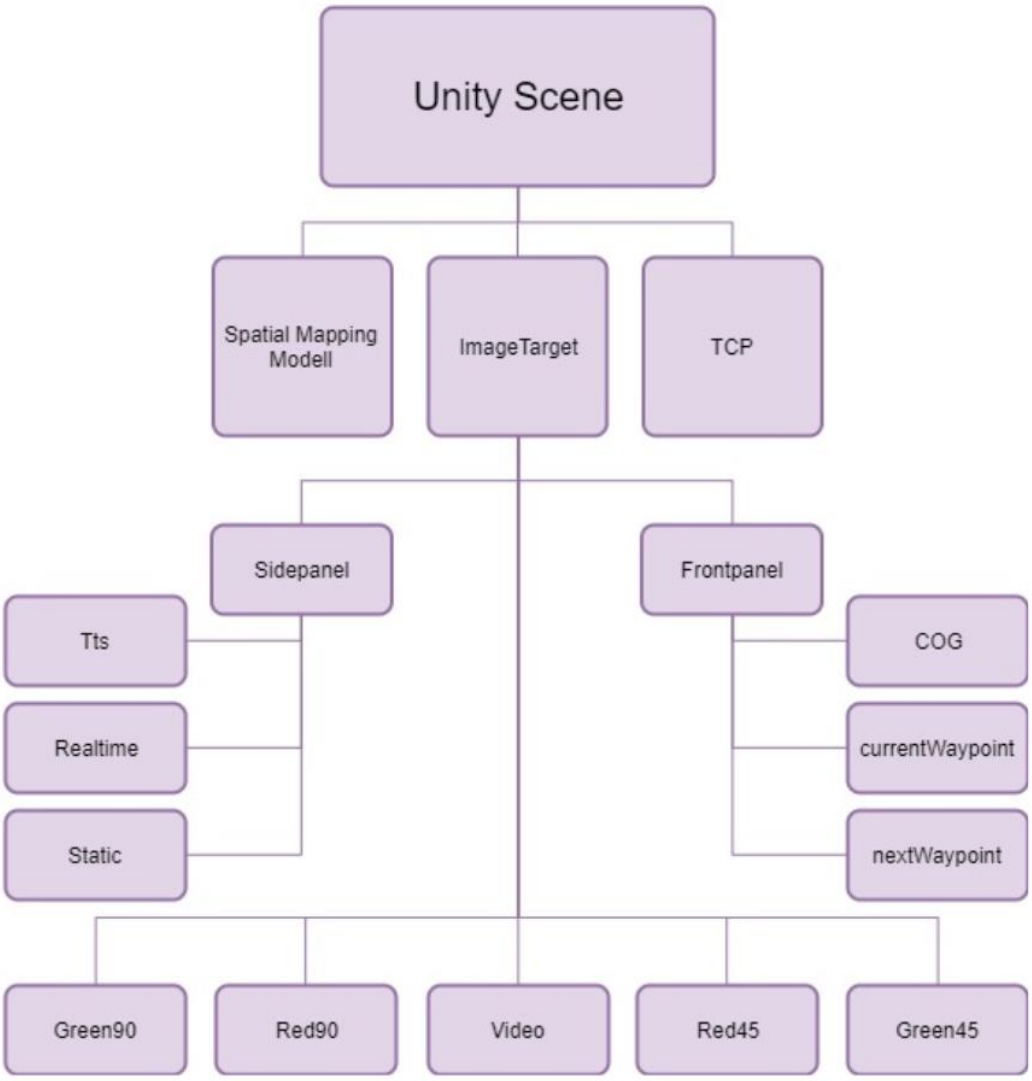
Denne fanen har totalt tre funksjoner. I *Figur 34* vil den venstre delen sammen med «*Move by*» nederst til høyre har til oppgave å flytte på de ulike modulene i scenen. Høyre del har til oppgave å endre på farge og skriftstørrelse. I tillegg kan man sende tekst som vil bli lest opp av HoloLens etter en gitt forsinkelse. Elementer flyttes ved å angi hvor mange meter elementet skal flyttes, hvilket element som skal flyttes og til slutt ønsket retning. Når retning angis, vil elementet flyttes med de gitte parameterne. For å endre farge på skriften kan man enten skrive inn verdier for rød, grønn, blå og alpha eller velge farge med Node-RED-dashbord sin fargevelger.



Figur 34 UI Editor-fanen.

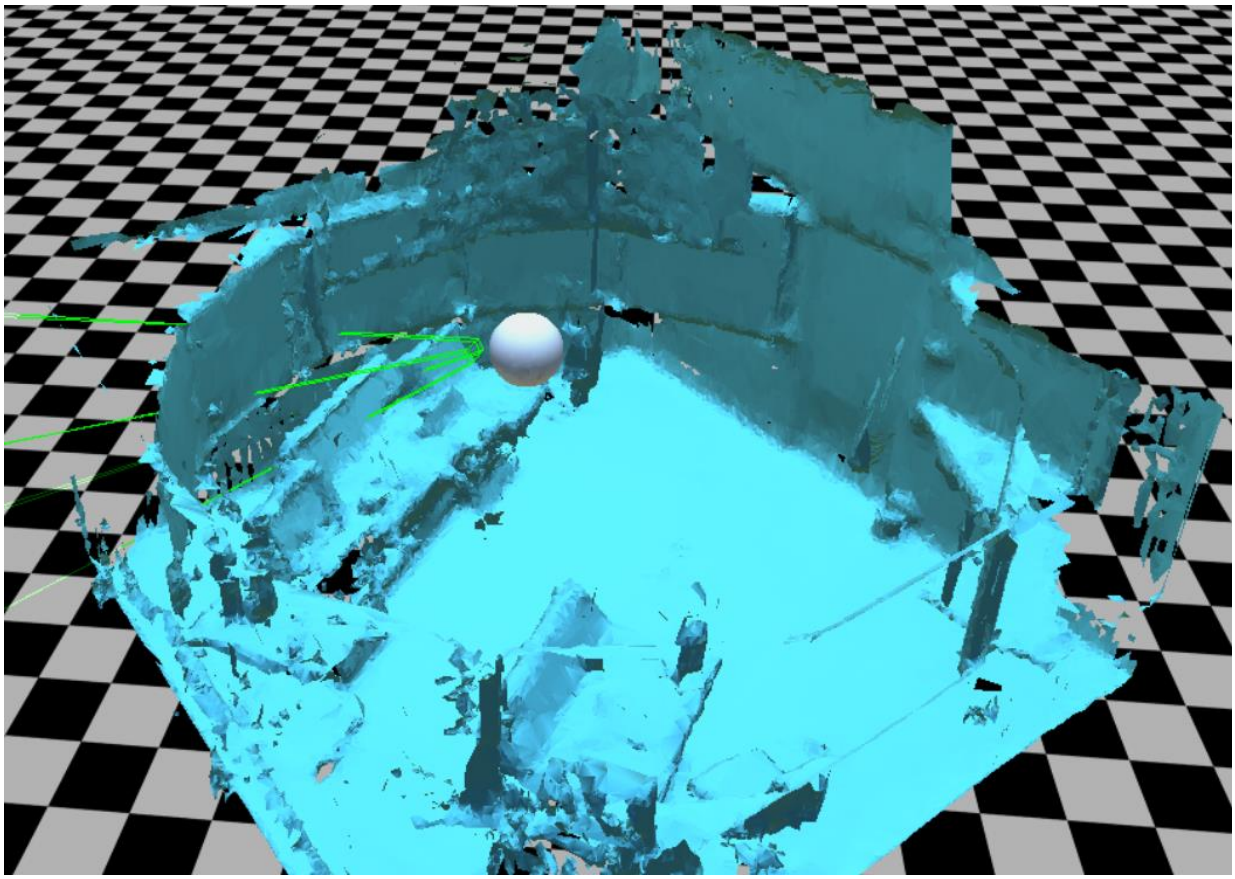
4.3 Unity

Unity er ikke et programmeringsspråk sammenlignet med JavaScript og C#, men det er fremdeles naturlig å dokumentere hvordan vi har implementert det på samme måte. Strukturen er bygget opp på følgende måte som illustrert i *Figur 35*.

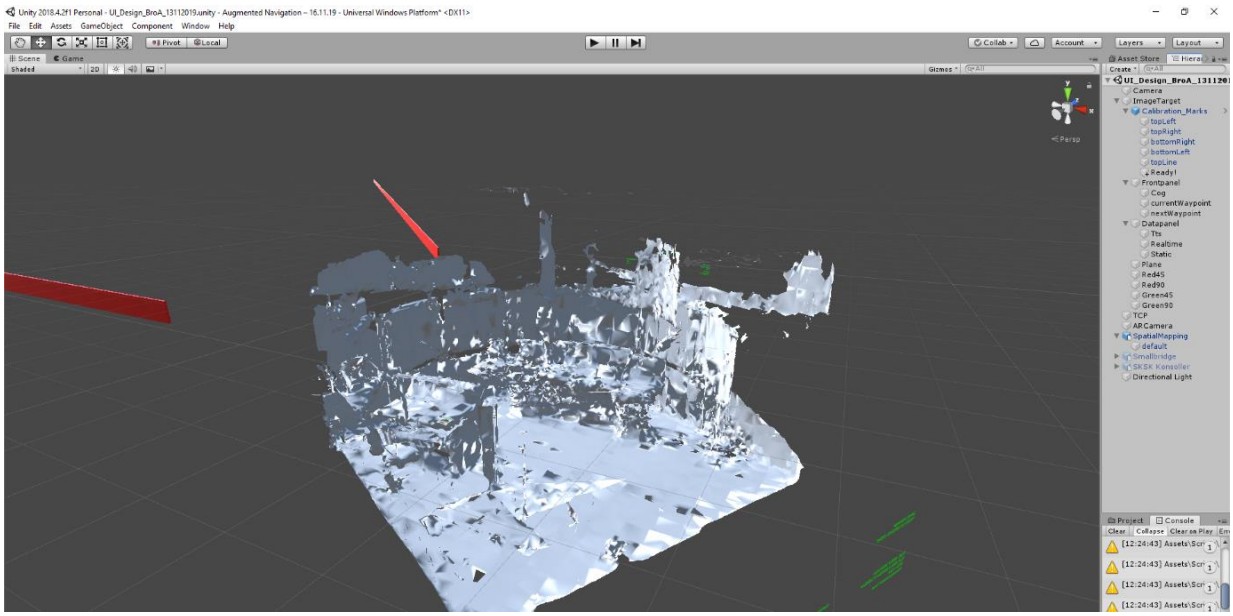


Figur 35 Hierarkiet til Unity-scenen.

Unity-scenen består av tre under objekter, som underobjekter under seg. Det første objektet er en spatial mapping modell. Vi brukte Unity sin spatial mapping teknologi for å kartlegge rommet. Resultatet ble et tredimensjonalt romkart som vi bruker til å plassere de ulike brukergrensesnittmodulene i. Dette reduserte tiden vi brukte på å plassere modulene riktig i brukergrensesnittet. I tillegg var det mye lettere å få et forhold til lengder og retninger ved å ha denne modellen å støtte seg på.

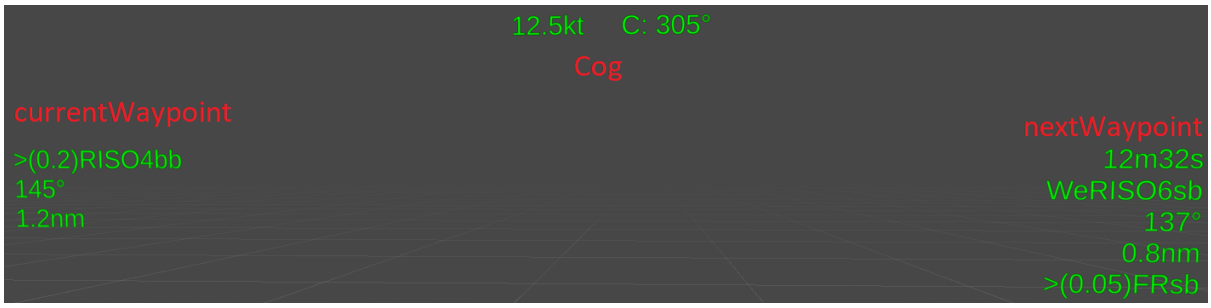


Figur 36 Spatial mapping i HoloLens.



Figur 37 Romkartlegging lagt inn i Unity.

Det neste objektet er imagetarget. Dette er det mest omfattende objektet i scenen og har flere objekter under seg. Vi går gjennom det systematisk: under imagetarget finner vi først frontpanelet. Dette er det panelet man ser dersom man står i simulatoren og ser rett forut. Herunder finner man igjen underobjekter. Frontpanelet er illustrert i *Figur 38*.



Figur 38 Frontpanel med tre tekstfelt.

Den røde teksten markerer de ulike tekstfeltene i frontpanelet. Ut ifra denne figuren kan man se hvordan de ulike objektene ligger i synsfeltet. Sidepanelet er bygget opp på samme måtte og vil ikke bli videre utdypet. De andre underobjektene hører ikke til noe plan, men er spredt rundt om i scenen. Herunder finner man de ulike retningsindikatorene som skal vise grønn og rød 45/90 grader.

4.3.1 Vuforia

Bildemålet som blir brukt har en egen database med bildets kjennetegn. Disse kjennetegnene bruker Vuforia Engine til å feste den virtuelle verden til den reelle verden. Dette gjør at programmet blir regelmessig og likt. Et alternativ ville vært å starte brillene i samme posisjon hver gang, men dette gir ikke mulighet til å kalibrere på nytt.

Vuforia er inkludert i Unity versjon 2018.4.2f1, og det krever derfor svært lite for å komme i gang. Vi la til et «*AR Camera*» og et «*ImageTarget*». Alle objektene ligger som et barn av vårt målbilde. Alt som vises i vårt program, har en relativ posisjon i forhold til dette bildet. Det er så AR-kameraet som sørger for at alle elementene blir gjengitt på rett plass i forhold til bildet.

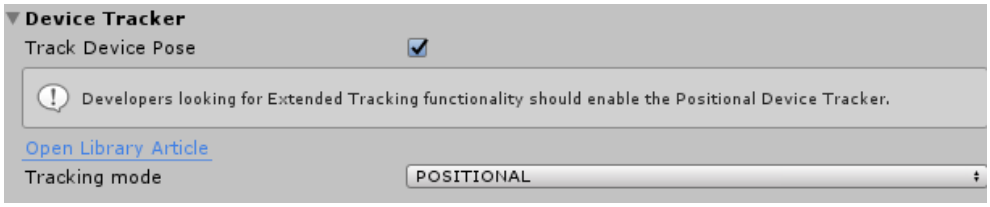
4.3.1.1 Bildevalg

Det er viktig å velge et bilde med tydelig retning, og vi prøvde oss derfor frem med flere ulike bilder. En kort periode ble logoen til kadettforeningen «*Valkyrien*» brukt. Konsekvensen av dette var at det virtuelle overlegget sin rotasjon ble uregelmessig. Etter noen mindre, uformelle tester[4] kom vi frem til at bildet av tannhjulet ga langt mer stabile resultater. Det var tydelig at Vuforia var avhengig av kjennemerkene samlet på teksten under Valkyrien-logoen. Sannsynligvis kom den uregelmessige rotasjonen av at bildet var symmetrisk.

[4] Testen bestod av å dekke til deler av bildet og rotere det.

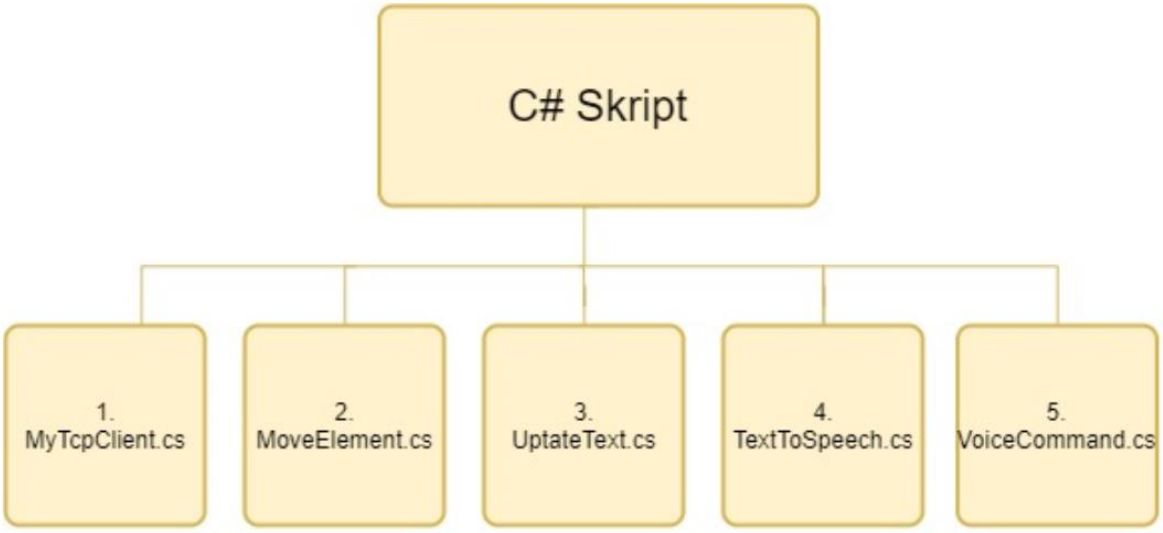
Figur 39 Valkyrien bildemål.

Under konfigurasjonen av Vuforia er Device Tracker er skrudd på. Dette gjør at Vuforia husker hvor bildet ble sist sett, og holder alle gjengivelser aktive. Eksempelvis kan man se på kalibreringsbildet, se vekk og fjerne bildet for deretter å se tilbake. Da vil man se Ready!-firkanten i posisjonen hvor bildet tidligere lå. Dersom HoloLens oppdager bildet på en ny plassering vil tidligere gjengivelse fjernes og alt blir vist i nye, kalibrerte posisjoner.



Figur 40 Tillater permanent Vuforia-fremvisning.

4.4 C# Skripts



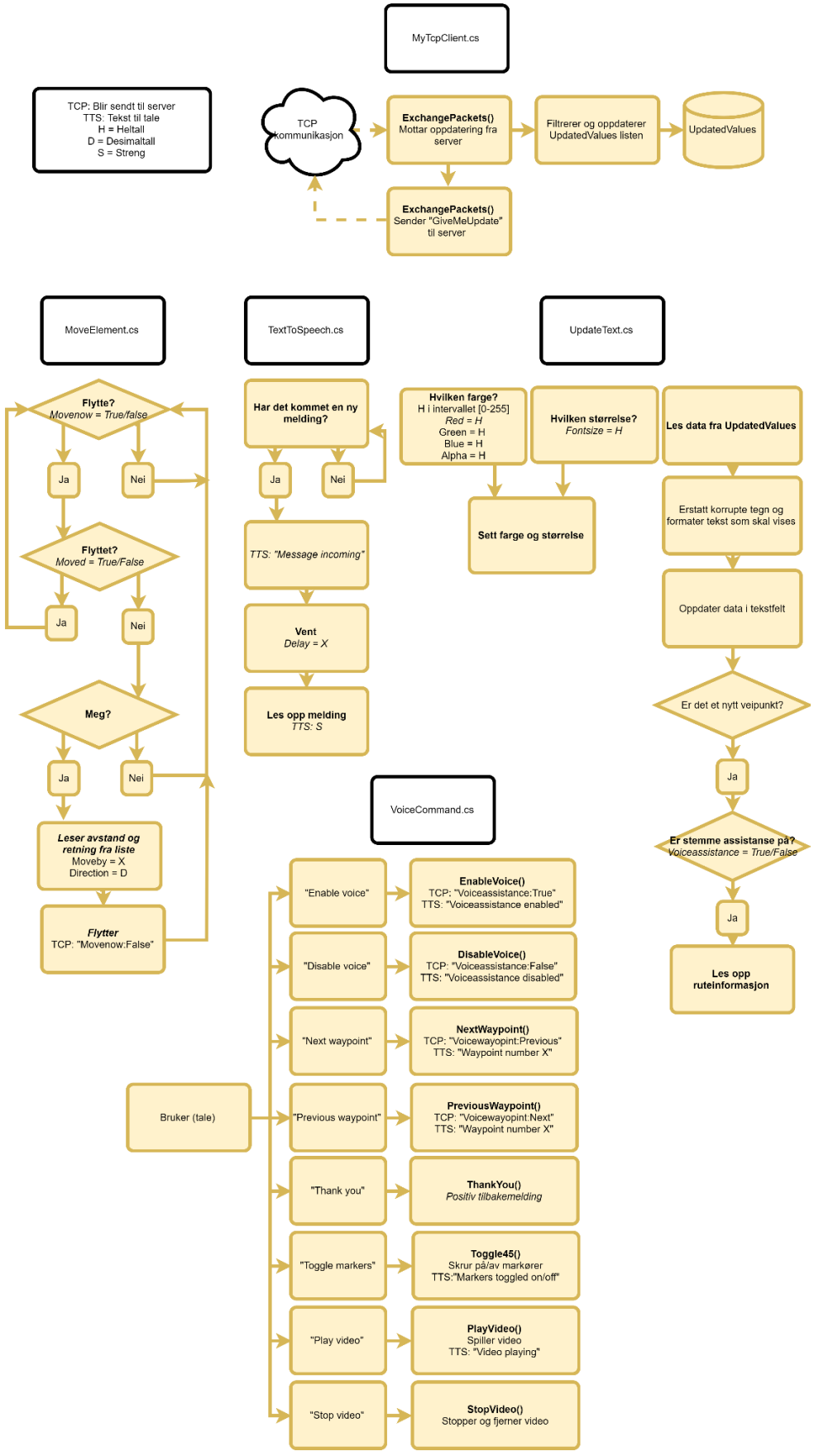
Figur 41 De ulike skriptene brukt i Unity.

Figur 41 er en oversikt over de ulike skriptene brukt i dette prosjektet. Som vist kan man se at det er fem skript.

- MyTcpClient.cs styrer overføringen av pakker mellom brillesettet[5]og serveren.
- MoveElement.cs kontrollerer flyttingen av de ulike elementene i brukergrensesnittet.
- UpdateText.cs styrer oppdateringen av tekstfeltene i scenen. Skriptet vil også styre tekstens farge og størrelse.
- TextToSpeech.cs styrer funksjonene som gjør det mulig å få tekst til tale.
- VoiceCommand.cs håndterer all stemmestyring.

På *Figur 42* ser man et flytdiagram over hvordan de ulike skriptene fungerer i grove trekk.

[5] Klienten



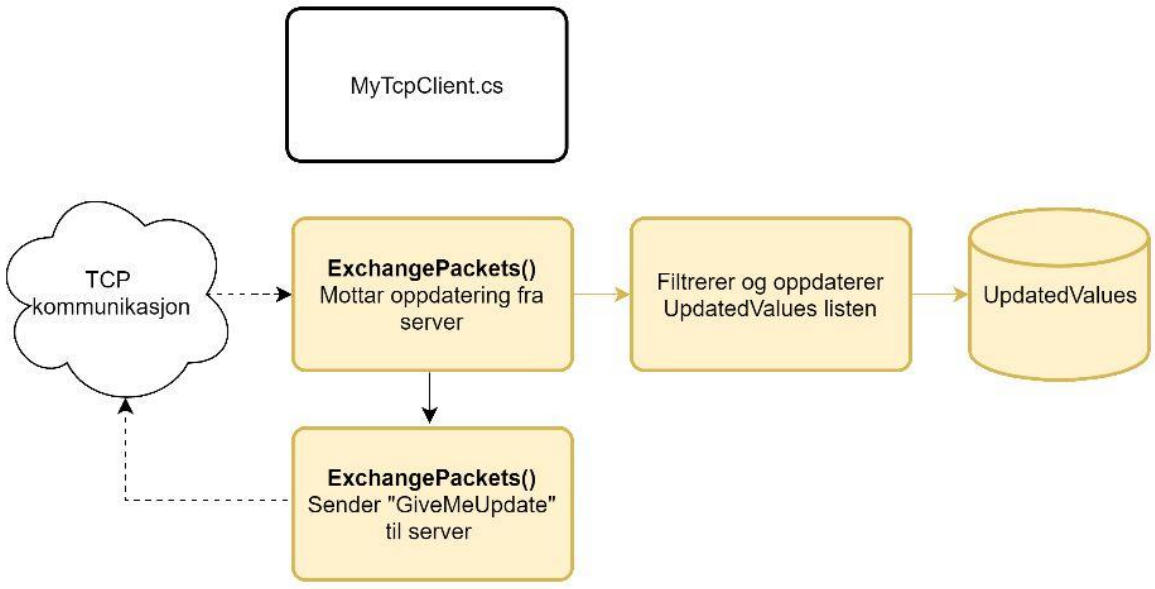
Figur 42 Flytdiagram av C# skripts.

4.4.1 MyTcpClient.cs

Skriptet håndterer kommunikasjonen med TCP-serveren som beskrevet i *kapittel 4.2.3*. Den første delen av skriptet tar for seg de rent nettverkstekniske funksjoner som gjør at man kan etablere en TCP-forbindelse med serveren. Av viktige funksjoner som burde nevnes er *ExchangePackets()*. På *linje 178* i *Figur 43* kan man se at ved hjelp av funksjonen *write()* sender klienten *«GiveMeUpdate»* til serveren. Det er denne forespørselen som gjør at klienten får oppdaterte variabler tilbake. På *linje 181* blir data mottatt av funksjonen *ReadLineAsync()*, og lagret i variabelen *PreviousResponse* som blir håndtert videre og aktuelle verdier blir gjort tilgjengelig.

```
172     public async void ExchangePackets()
173     {
174
175         try
176         {
177             // Sends "GiveMeUpdate" to the server
178             writer.Write("GiveMeUpdate");
179
180             // Writes the recieved message to PreviousResponse
181             PreviousResponse = await reader.ReadLineAsync();
182
183             //Writes response to console in Visual Studio
184             Debug.Log(PreviousResponse);
185
186             //Splits the string twice and updates the variables with their new values
187             string[] elements = PreviousResponse.Split('|');
188             int index = 0;
189             foreach (var elem in elements)
190             {
191                 if (elem == "")
192                     continue;
193                 string[] split = elem.Split(':');
194                 index = findIndex(AvailableVariables, split[0]);
195                 UpdatedValues[index] = split[1];
196                 index++;
197             }
198         }
199
200         catch (Exception e)
201         {
202             Debug.Log(e.ToString());
203         }
204     }
```

Figur 43 ExchangePackets() fra MyTcpClient.cs.



Figur 44 MyTcpClient.cs flytdiagram.

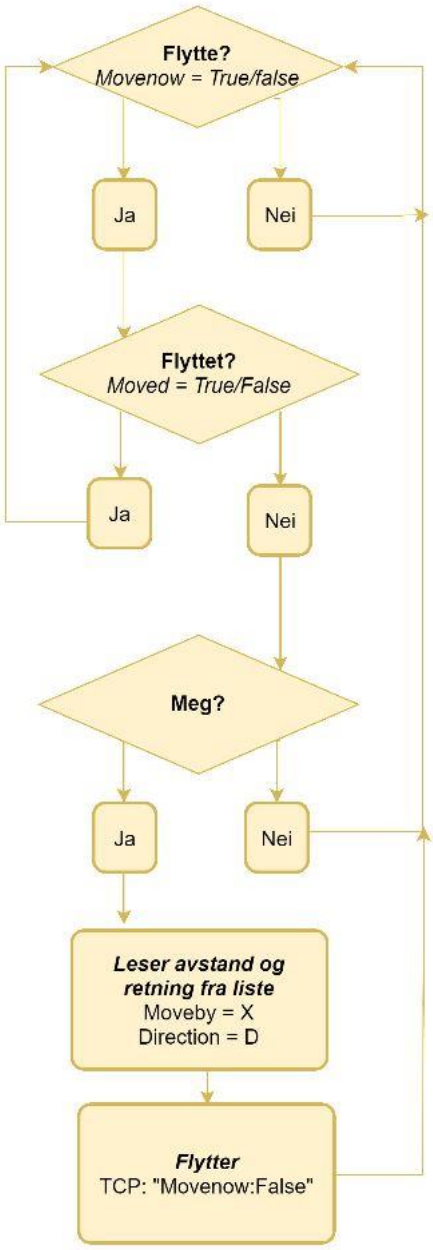
4.4.2 MoveElement.cs

Skriptet håndterer alle flytt av elementer i scenen. Nedenfor er logikken som koden bruker for å gå gjennom ett flytte, sammen med ett flytdiagram som illustrerert i *Figur 46.*

Flytte?: HoloLens får melding av serveren om å flytte.

Flyttet?: Sjekker om elementet allerede har flyttet[6].

Meg?: Sjekker hvilket element som skal flyttes[7].



Figur 46 flytdiagram av MoveElement.cs.

```
33  void  MoveIt()
34   {
35       // Get variable
36       string Movenow = MyTcpClient.giveMe("Movenow");
37       if (Movenow != "True")
38           moved = false;
39
40       // If the variable is set to "True" the function will continue
41       if (Movenow == "True" && !moved)
42       {
43           //Checks which element should move
44           bool Match = false;
45           string Element = MyTcpClient.giveMe("Element");
46           if ((Element == "Cog" && cog) || (Element == "Currentwaypoint" &&
               currentWaypoint) ||
47               (Element == "Nextwaypoint" && nextWaypoint) ||
                   (Element=="Allelements"))
48               Match = true;
49
50
51           if (Match)
52           {
```

Figur 45 Kriteria for flytt.

Dersom kriteriene for å gjennomføre flytt er oppfylt, blir elementet flyttet med parametere for avstand og retning gitt fra server. Merk at bildet under bare er et utdrag, og det er definert flere retninger med samme fremgangsmåte.

```
56                // Moveby is how far
57                string Movebystring = MyTcpClient.giveMe("Moveby");
58                float Moveby = float.Parse(Movebystring);
59
60                // Direction is one of six directions (think of a dice)
61                string Direction = MyTcpClient.giveMe("Direction");
62
63                // Defines which component we wish to move (in this case it's
                    the object which the script is attached to)
64                RectTransform myRectTransform = GetComponent<RectTransform>();
65
66                // Towards user
67                if (Direction == "Towards")
68                    myRectTransform.localPosition += Vector3.up * Moveby;
69
70                // Away from user
71                else if (Direction == "Away")
72                    myRectTransform.localPosition += Vector3.down * Moveby;
73
74                // down
75                else if (Direction == "Down")
76                    myRectTransform.localPosition += Vector3.left * Moveby;
77
```

Figur 47 lengde og retning på flytt

[6] Det tar litt tid før serveren har blitt informert om et utført flytt. Dette unngår uregelmessige flytt.
[7] Denne sørger for at hvert enkelt element sjekker om det er «meg» som skal flytte.

4.4.3 UpdateText.cs

Hovedfunksjonen til dette skriptet er å hente ut dataen som har blitt overført fra serveren og passe på at den blir fremvist i riktig modul i brukergrensesnittet. Dette skriptet formaterer og presenterer dataen som har blitt sendt fra serveren. Samtidig har skriptet ansvaret for at fremvisning skjer i rett tekstfelt til rett tid – synlig for brukeren med brillene. Fremgangsmåten har likhetstrekk på kryss av tekstfeltene, vi kommer til å benytte oss av en av de mer kompliserte for å vise prinsippene.

```
96      else if (currentWaypoint)
97      {
98          string currentwaypointstring = MyTcpClient.giveMe("Prevwaypointname");
99          // Fixes symbols from TCP-communication
100         // In the TCP-communication some symbols gets interpeted wrong and needs to be corrected.
101         // These are the symbols that we found needed to be replaced
102         currentwaypointstring = currentwaypointstring.Replace("&amp;gt;", ">");
103         currentwaypointstring = currentwaypointstring.Replace("&amp;lt;", "<");
104         currentwaypointstring = currentwaypointstring.Replace("&#x2f;", "/");
105
106         // Reads the string after '>' or '<'
107         if (currentwaypointstring.Contains(">"))
108             currentwaypointstring = ">" + currentwaypointstring.Substring(currentwaypointstring.LastIndexOf('>') + 1);
109         else if (currentwaypointstring.Contains("<"))
110             currentwaypointstring = "<" + currentwaypointstring.Substring(currentwaypointstring.LastIndexOf('<') + 1);
111         // Formats text to display
112         textToScreen = currentwaypointstring + "\n" +
113         MyTcpClient.giveMe("Bearing") + "°\n" +
114         MyTcpClient.giveMe("Distancetowaypoint") + "nm\n";
115     }
```

Figur 48 Formatering av tekstfelt.

For å beskrive bruker vi kodesnutten i *Figur 48*. Variabelen *textToSceen* representerer det som til slutt vises i brilleglasset. For å endre teksten som vises i brillene, setter man *textToScreen* til ønsket tekst. Vi benytter metoden *MyTcpClient.giveMe()* til å hente verdiene som er lagret lokalt av *MyTcpClient.cs* som beskrevet i *kapittel 4.4.1*.

Figur 50 kan man se hvordan de ulike variablene vil manifestere seg i brukergrensesnittet.


```
textToScreen = currentwaypointstring + "\n" +
MyTcpClient.giveMe("Bearing") + "°\n" +
MyTcpClient.giveMe("Distancetowaypoint") + "nm\n";
```

Figur 50 Tekstfelt i Unity.

4.4.4 TextToSpeech.cs

Dette skriptet gjør tekst til tale. I *Figur 51* ser vi lite utdrag for å illustrere hvordan skriptet kan brukes. *Linje 86 og 87* setter betingelser som må være oppfylt. Dersom betingelsene blir oppfylt, vil funksjonen Say(String) kalles på, og lese opp den gitte teksten. I dette eksempelet vil brillesettet for eksempel lese opp «*Waypoint reached, advancing to waypoint number five*».

```
83      //Gives the user feedback if the waypoint was automatically switched.
84      public async void AutomaticNext()
85      {
86          if (MyTcpClient.giveMe("Automaticwaypoint") == "True" &&
87              (Waypointid != int.Parse(MyTcpClient.giveMe("Waypointid"))) && !isTalking)
88          {
89              isTalking = true;
90              Waypointid = int.Parse(MyTcpClient.giveMe("Waypointid"));
91              MyTcpClient.forceSend("Automaticwaypoint:False");
92              Say("Waypoint reached, advancing to waypoint number" + Waypointid);
93              isTalking = false;
```

Figur 51 Bruk av Say(string).

4.4.5 VoiceCommands.cs

Dette skriptet sørger for gjenkjenningen av stemmekommandoer som vist på *Figur 52*. For en dypere forklaring av hvordan dette blir tilrettelagt, se *vedlegg K*.

```
//Avaiable voicecommands
keyActs.Add("enable voice", EnableVoice);
keyActs.Add("disable voice", DisableVoice);
keyActs.Add("next waypoint", NextWaypoint);
keyActs.Add("previous waypoint", PreviousWaypoint);
keyActs.Add("thank you", ThankYou);
keyActs.Add("toggle markers", Toggle45);
keyActs.Add("play video", PlayVideo);
keyActs.Add("stop video", StopVideo);
```

Figur 52 Tilgjengelige kommandoer.

HoloLens lytter kontinuerlig etter kommandoer fra brukeren. Dersom en strofe blir gjenkjent vil programmet respondere med rett handling. Dersom brukeren sier «*next waypoint*» vil følgende kode, som vist på *Figur 53*, kjøres. Klienten ber serveren bytte veipunkt, bekrefter for brukeren at veipunktet er byttet samt hvilket veipunkt som nå vises.

```
void NextWaypoint()
{
    MyTcpClient.forceSend("Voicewaypoint:Next");
    int nextWaypoint = int.Parse(MyTcpClient.giveMe("Waypointid")) + 1;
    TextToSpeech.Say("Waypoint number " + nextWaypoint);
}
```

Figur 53 NextWaypoint() fra VoiceCommands.cs.

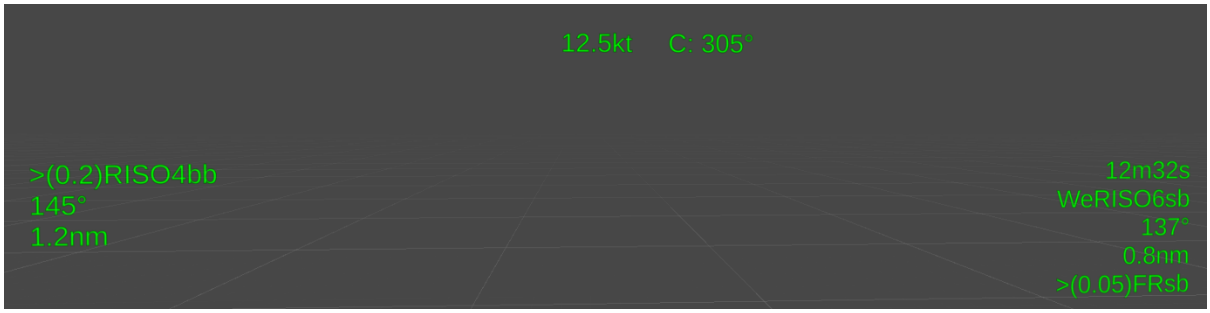
5 Resultater

I dette kapittelet vi alle resultatene bli lagt frem. Vi har definert resultater som de ulike elementene i AR-brukergrensesnittet, det digitale dashbordet i Node-RED, funnene fra de heuristiske evalueringene samt evaluering av måloppnåelse

5.1 AR-Brukergrensesnitt

Brukergrensesnittet i brillesettet er delt opp i tre moduler. Frontpanelet, sidepanelet og de fire retningsindikatorene. I tillegg har vi et bildemål for å kalibrere dette brukergrensesnittet til simulatoren. Videre kommer vi til å presentere det digitale brukergrensesnittet i Node-RED. Vi kommer til å gå gjennom hver del i hvert sitt delkapittel.

5.1.1 Frontpanelet



Figur 54 Frontpanelet i Unity.

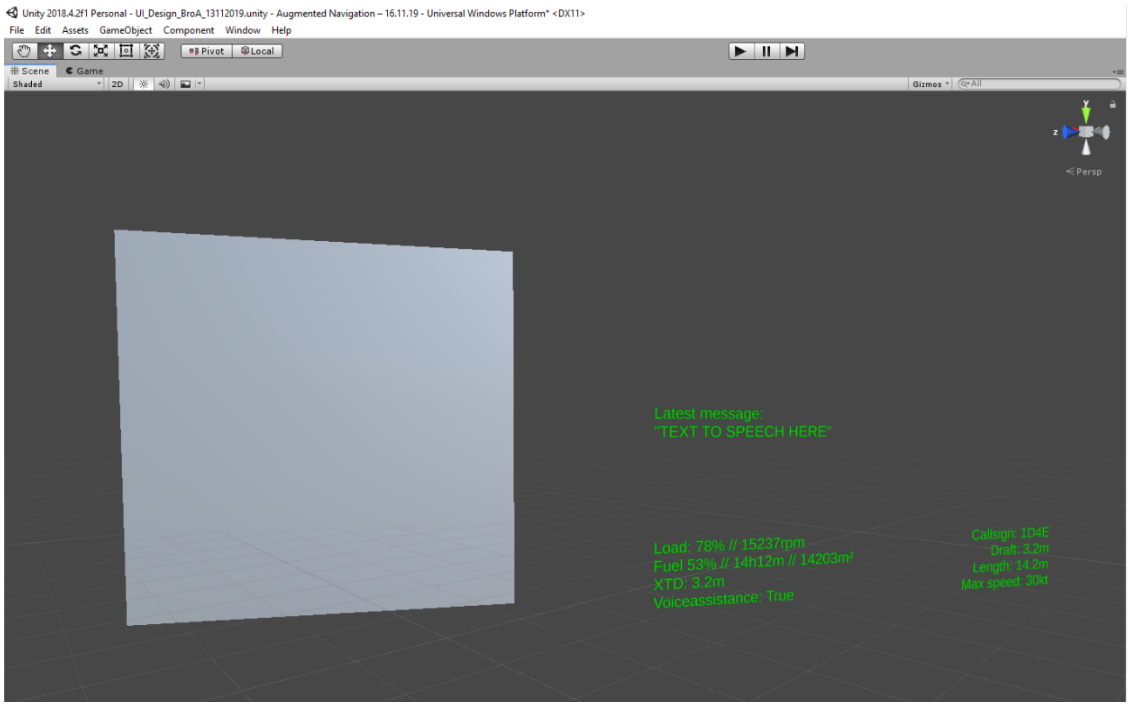
Frontpanelet er hvor den mest sentrale navigasjonsdataen blir presentert. Man kan ut ifra brukergrensesnittet i *Figur 54* se at panelet består av tre soner. Den øverste sonen inneholder informasjon om fart og kursen. Farten og kursen kommer fra NMEA-strengen $GPVTG og har en oppdateringsfrekvens på 10 hertz. Modulen til venstre gir informasjon gjeldende den inneværende kurs. Fra øverste til nederste linje er: inneværende stevn. kursen man skal holde og gjenværende distanse til neste veipunkt. Tekstfeltet til høyre inneholder informasjon om neste leg. Øverste til nederste linje: tid til neste turn i minutter og sekunder, tørnindikator, etterfulgt av kursen og distansen på legget. Til slutt kommer kursnotasjonen som gir informasjon om neste stevn. En viktig presisering her er at informasjonen til høyre, utenom tidsparameteren, ikke vil vises før det er mindre enn 3 minutter til tørn[8].

[8] Beskrevet i detalj i *Vedlegg O* under *UpdateText.cs*

Figur 55 Frontpanel under bruk, skjermbilde fra HoloLens.

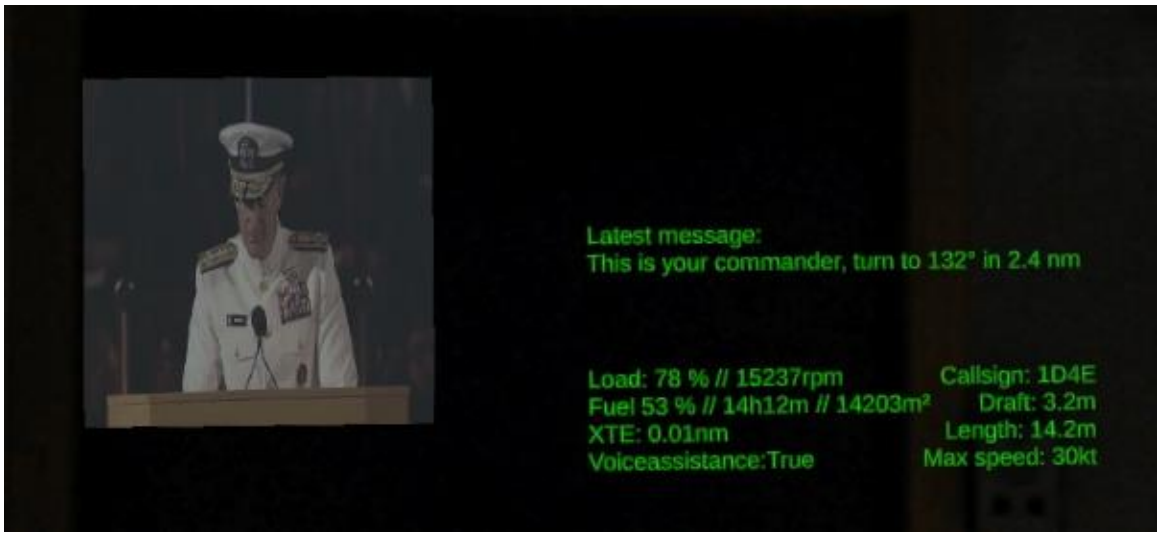
Til sammenligning med *Figur 54*, hvor brukergrensesnittet i Unity vises, viser *Figur 55* brukergrensesnittet *i* simulatoren under seilas. Bilde er tatt med Microsoft HoloLens sitt Mixed Reality Capture verktøy og viser eksakt hvordan brukeren ser det.

5.1.2 Sidepanel



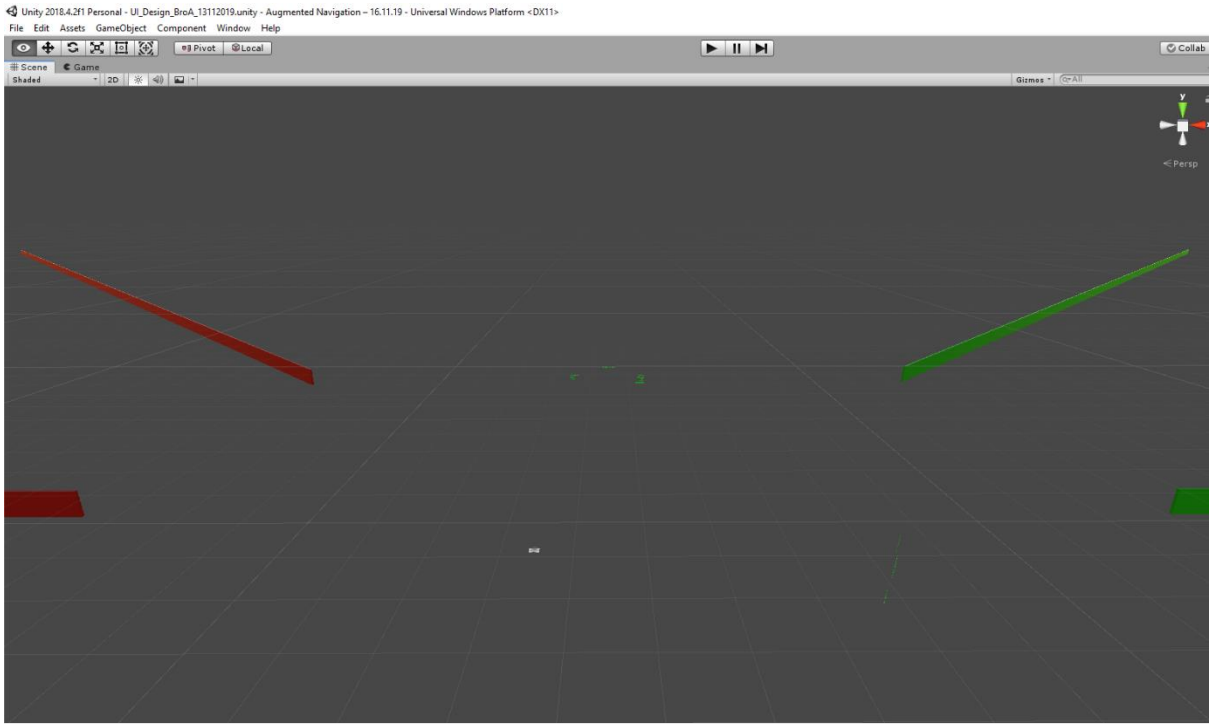
Figur 56 Sidepanelet i Unity.

Vi har også lagt til et datafelt som viser annen informasjon, som ikke trenger være i frontpanelet, men fortsatt tilgjengelig. Det være seg informasjon om fartøyet, drivstoffnivå, callsign og siste melding mottatt. Den grå firkanten til venstre på *Figur 56* representerer et lerret hvor video kan bli spilt. Viktig presisering er at det er kun videospilling, siste melding og stemmeassistance er fungerende dynamiske variabler.



Figur 57 Sidepanel under bruk med video aktiv.

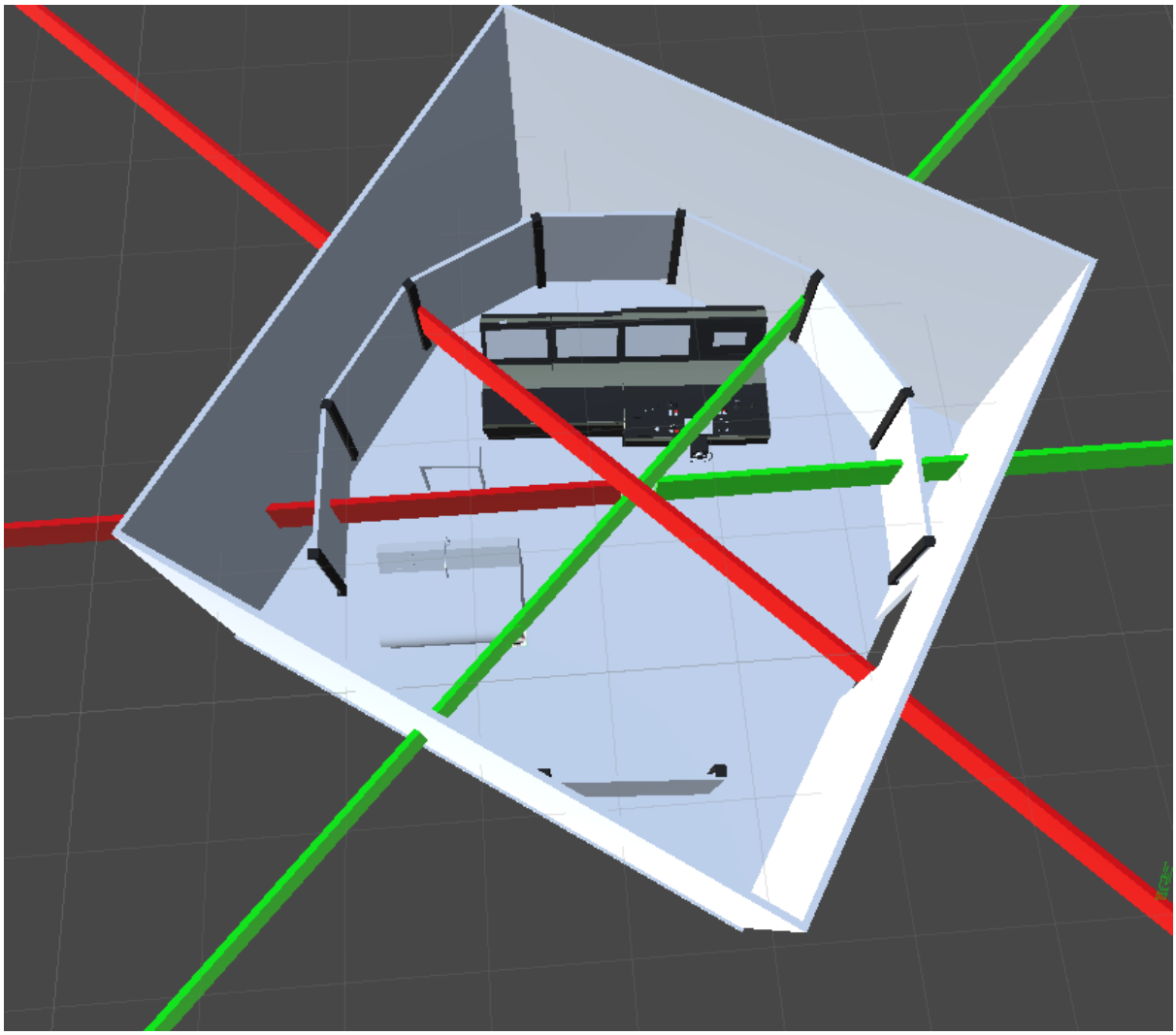
5.1.3 Retningsindikatorer



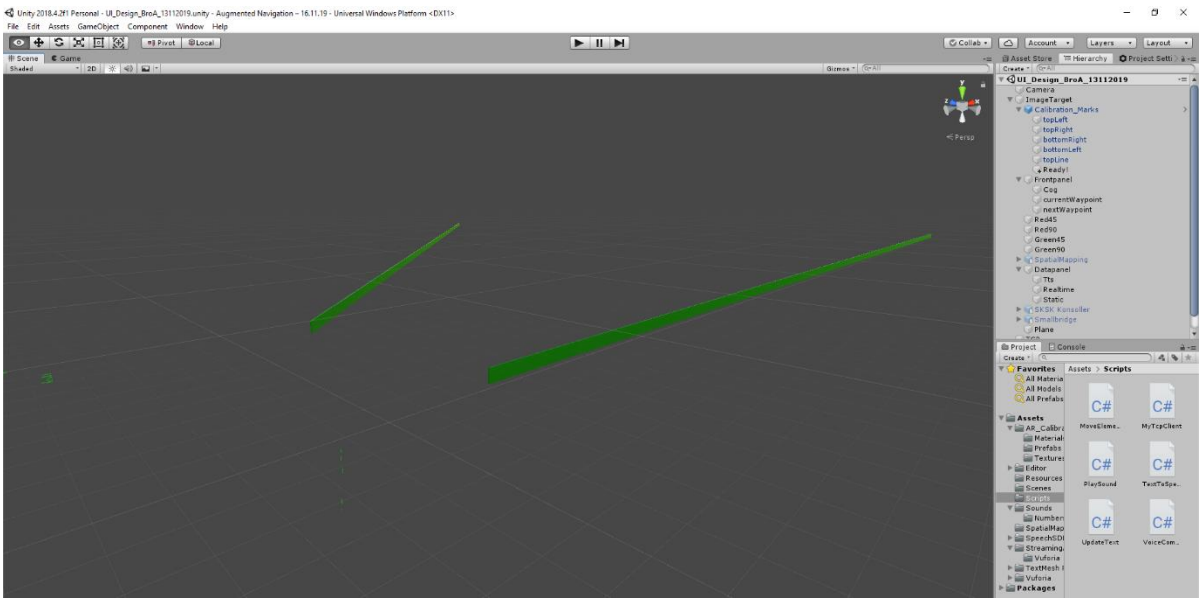
Figur 58 Retningsmarkører i Unity. Viser hvor 45˚ og 90 ˚ er i forhold til fronten på skipet.

Vi har etter ønske fra testpersoner[9] laget fire retningsindikatorer ved hjelp av fire kuboide-objekter i Unity. Disse vil gjøre navigatørene mer fristilt fra peilesøylen, ettersom disse hjelper å bedømme relativ peiling til gjenstander og fartøy. I *Figur 58* kan vi se disse, det første paret som er de to nærmest midten skal indikere 45° i forhold til rett frem. Kuboide-objektene står 90° på hverandre slik at de danner 45° gradere fra senter. Det andre paret skal indikere 90° i forhold til 0° rett frem. Alle kuboide-objektene krysser peilesøylen[10] i simulatoren som vist under *Figur 59*, og vil dermed kunne brukes til navigasjon. Ved forlengelse av objektene vil alle krysse dette punktet.

[9] Man tørner gjerne med lykter o.l. ved 45 eller 90 grader i forhold til fartøyet.
[10] Brudeseth OBD. Peilesøylen kan sees øverst, på midten av *Figur 13.*

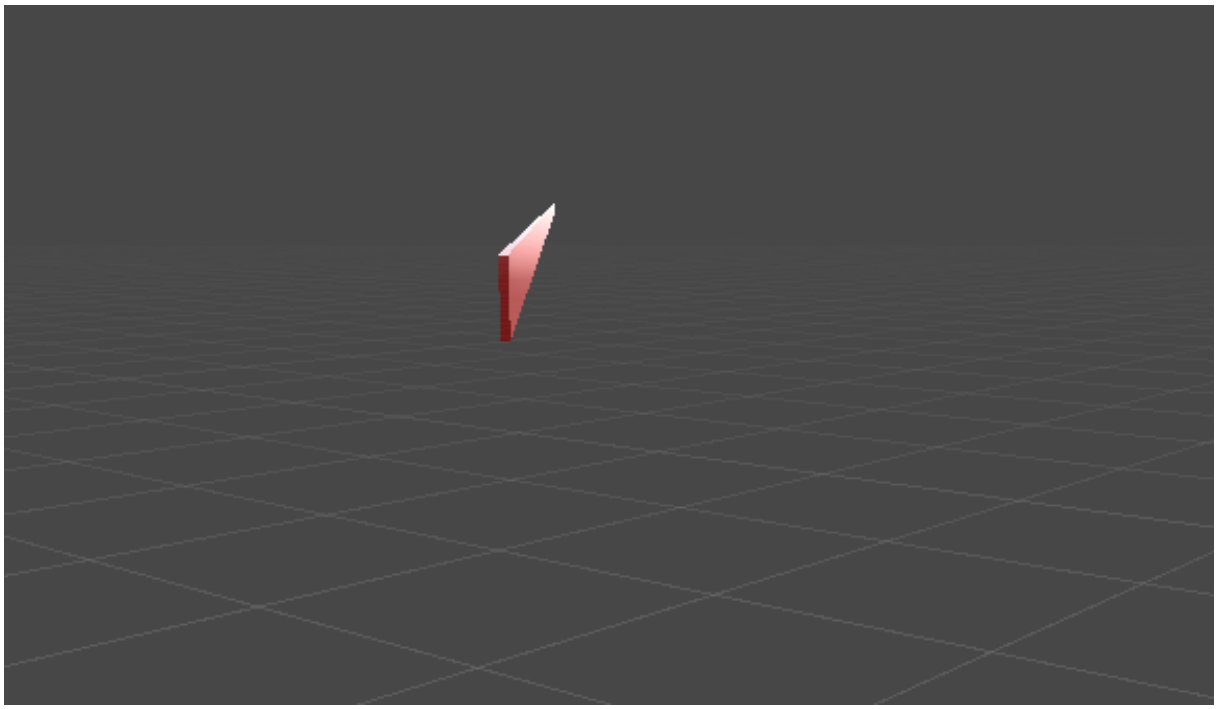


Figur 59 Alle markører krysser på peilesøylen.



Figur 60 Lengden til et 45° og 90° par.

Figur 61 Rød 45° på babord i HoloLens.



Figur 62 Rød 45° på babord i Unity.

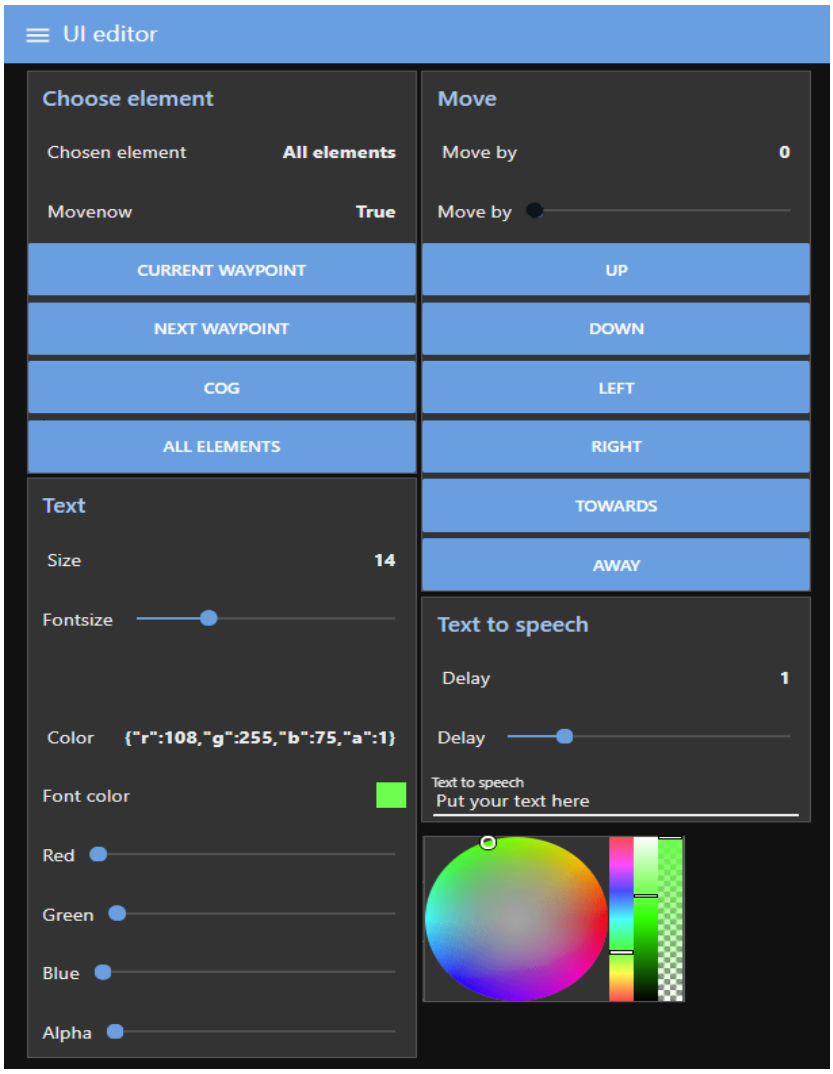
5.1.4 Aktivering og kalibrering av brukergrensesnittet

Figur 63 Venstre: bildemålet i Unity. Høyre: bildemål gjenkjent av Vuforia.

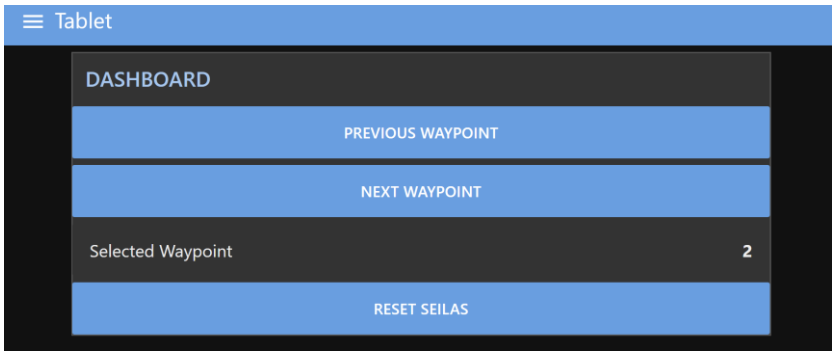
Den endelige prototypen bruker "*Time to Recalibrate*?"-bildet for å starte og replassere objektene i rommet relativt til denne. Når man skal få frem de virtuelle objektene i brillene, må man se på bildet og vente til den grønne stiplede firkanten og «*Ready!*» vises som på *Figur 63.* Når den grønne firkanten kommer til synet, er brukergrensesnittet kalibrert og klart til bruk.

5.2 Digitalt dashbord i Node-RED Dashbord

Det digitale brukergrensesnittet i Node-RED består av seks faner hvorav to er tiltenkt å brukes av navigatøren. Den første fanen skal kunne flytte på de ulike modulene, samt endre farge og skriftstørrelse. Den andre skal kunne bytte mellom de ulike veipunktene som et alternativ til stemmekommandoer. Disse kan du henholdsvis se i henholdsvis *Figur 64* og *Figur 65.* Under «*Choose element*» velger man hvilket tekstfelt som skal flyttes. Til høyre velger man hvor langt og hvilken retning tekstfeltet skal flyttes. Det blir flyttet hver gang man trykker på en retning. Under «*Text*» velges størrelse på font. Videre ned finner man mulighet for å endre farge på teksten. Dette kan gjøres enten ved å stille på hver enkelt komponent eller ved å bruke fargevelgeren som vist nederst til høyre. Under «*Text to speech*» kan man velge forsinkelse mellom varsel og melding, samt sende en melding til HoloLens som vil bli lest opp for brukeren. Dette gjøres ved å fylle inn ønsket melding under «*Text to speech*» inndatafeltet og trykke enter. Fanen for å endre veipunkt er tiltenkt å åpne på et nettbrett ellet tilsvarende, her er det kun mulig å hoppe mellom veipunkt, samt starte seilaset og begynne på første veipunkt. Dersom man skulle ønske å lese om de resterende fanene og derav få en dypere forståelse av programmeringen kan man lese i *vedlegg D.*



Figur 64 UI Editor-fane I Node-RED dashboard.



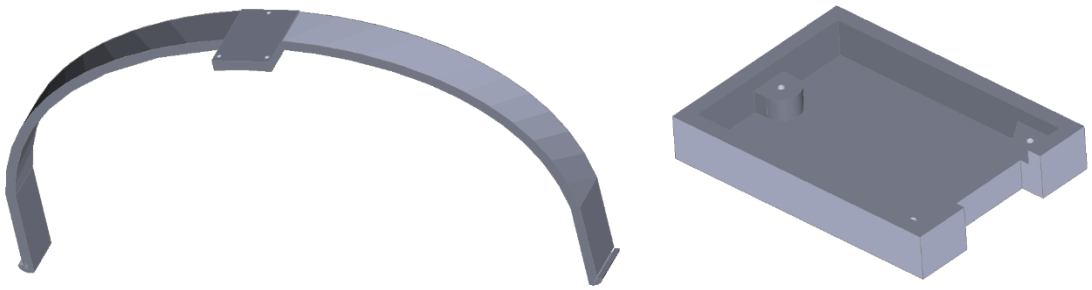
Figur 65 Tablet-fane, Node-RED dashbord..

5.3 3D-printing av HoloBøyle

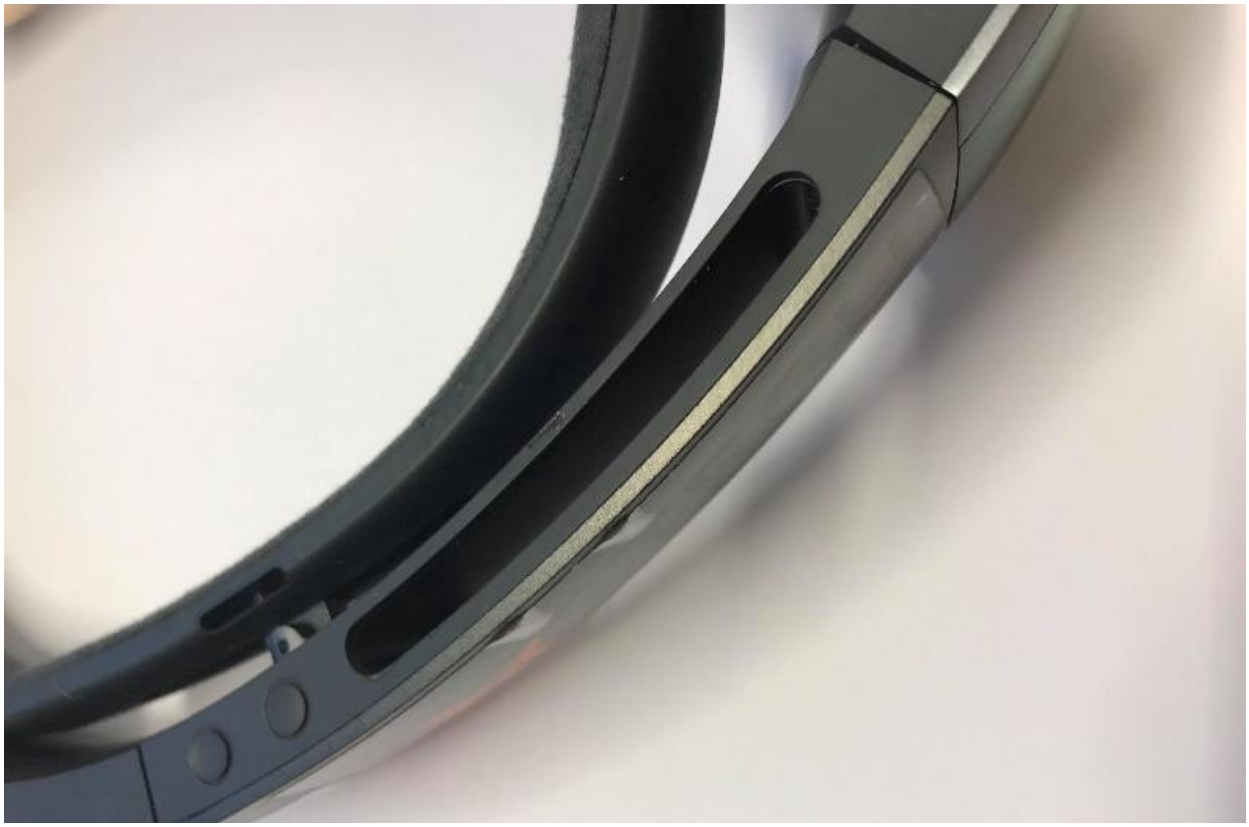
Vi har i dette prosjektet utviklet og designet en bøyle som kan festes til Microsoft HoloLens. Bøylen kan holde en sensorbrikke som gir headingen til brillene relativ til nord. Modellen er designet i *SolidWorks* og printet i en *Arkforged MK2* printer. Se *vedlegg O* for detaljerte tegninger.

Bøylen består av to deler. Selve bøylen går over hodet i en halvsirkel og vises i *Figur 66.* Sensorhuset er festet til "*HoloBøylen*" ved hjelp av skruer, og er vist i *Figur 67*. Kabelruting blir gjort ved bruk av strips på innsiden av bøylen.

Figur 66 Bøyle ferdig montert på brillene.



Figur 67 3D-modell av HoloBøylen og sensorhuset.



Figur 68 Viser ventilasjonskanalen som bøylen blir plassert i.

Innfestingsmekanismen drar nytte av ventilasjonskanalen til brillesettet. Enden av bøylene har en tapp som kan presses nede i ventilasjonskanalen. Ventilasjonskanalen blir smalere jo lengre ned man kommer i ventilasjonskanalen. Når bøylen er ført igjennom kanalen vil tappen holde igjen som en krok. Vi har ikke hatt noen problemer med høy temperatur til tross for denne modifikasjonen. Temperaturen leser vi av enhetens nettportal.

5.4 Resultater fra heuristisk evaluering

Figur 69 Stemningsbilde fra en av de heuristiske evalueringene.

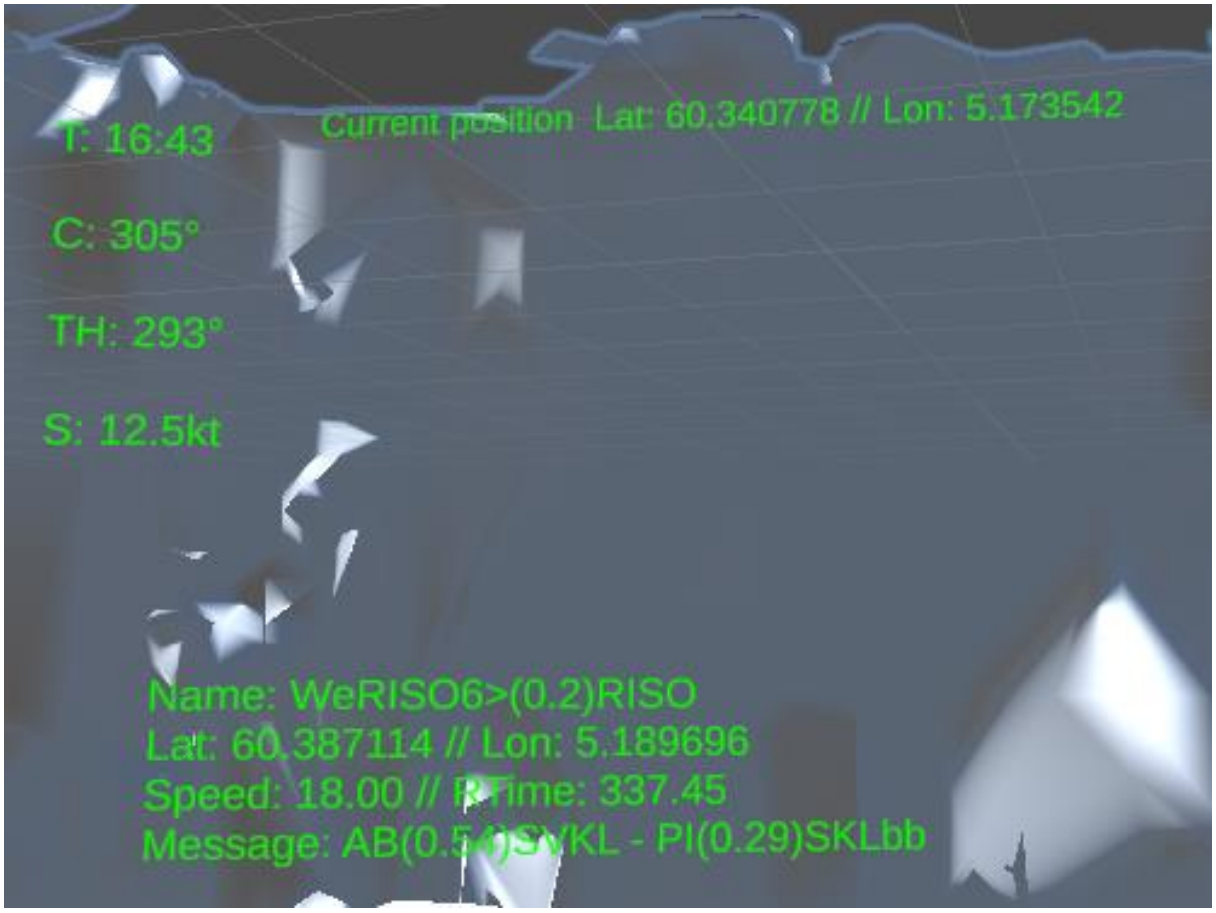
5.4.1 Praktiske tester: Del I

Den første heuristiske evalueringen ble gjennomført med to personer. Etter denne testen ble det samlet inn tilbakemeldinger fra testpersonene. Disse ble også brukt som sparringspartnere når det kom til implementering av nye funksjoner og tilpasninger av programmet.

Tilbakemeldinger og observasjoner kan oppsummeres med:

- Informasjonen som prosjekteres oppleves som for stor.
- Lysintensiteten oppleves som for sterk.
- Det er ikke optimalt med bruk av grønnfarge, da dette kan gjøre at staker forsvinner.
- Deler av informasjonen som vises er unødvendig, samt sentral informasjon mangler.
- Stemmefunksjonen oppleves som hakkete, en knapp hadde vært bedre.
- De virtuelle objektene ligger for nært. Man må endre øyets fokus.
- Brukergrensesnittet bør gjenspeile hvordan kursene blir lest opp av assistenten på bro.
- Kursnotasjonen bør deles opp.
- Ønske om knapper som supplement til stemmestyringen

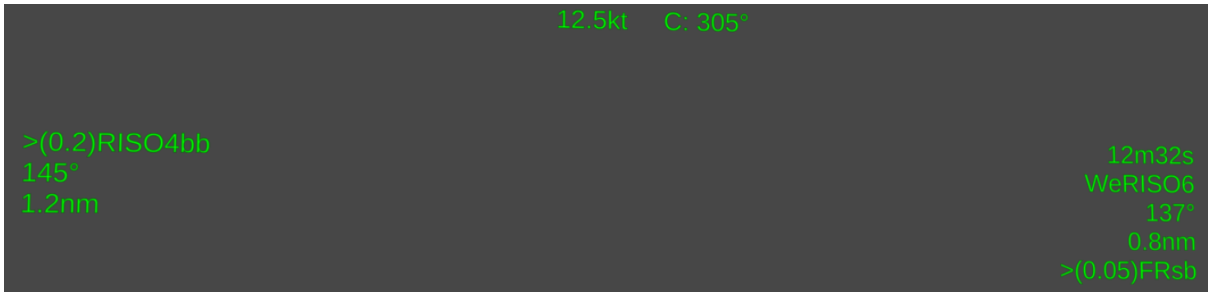
Frontpanelet så ut som *Figur 70* under Del I. Merk at dette var også de eneste elementene i scenen.



Figur 70 Frontpanelet under Del I (30.09.2019).

5.4.2 Praktiske tester: Del II

Den andre heuristiske evalueringen ble gjennomført etter å ha utbedret feil og mangler som vi oppdaget fra den første gjennomføringen. Frontpanelet ble endret, som vist på *Figur 71*, før Del II. Merk at alle elementene som beskrevet i *kapittel 5.1* var nå også en del av brukergrensesnittet.



Figur 71 Frontpanelet under Del 2.

Denne evalueringen ble gjennomført med seks testpersoner. De ble organisert i to grupper á tre testpersoner. Alle de utvalgte var navigasjonskyndige og ble valgt på bakgrunn av sin kompetanse.

Figur 72 Del II av heuristisk evaluering.

Tilbakemeldinger og observasjoner kan oppsummeres med:

- Systemet oppleves som lett å forstå, samt lett å implementere i etablerte rutiner.
- Brillene sin ergonomi er en utfordring, og det kreves erfaring for å justere brillene optimalt. Noen av testpersonene opplever også at må bevege nakken for å se informasjonen.
- Informasjonen som vises i brillesettet oppleves som gjennomtenkt og flyter godt med etablerte navigasjons prosedyrer. Fargen og lysstyrke oppleves forskjellig fra person til person, men testpersonene er fornøyd med at farge og lysstyrke kan tilpasses hver enkelt person. Videre oppleves dybden på informasjonen som vises som god.
- Noe av informasjonen i brillene oppleves som forsinket i forhold fra informasjonen som vises i navigasjonssystemene.
- Etter at testpersonene har blitt komfortabel med brillesettet, opplever de at ved å ha informasjonen tilgjengelig et økt mentalt overskudd. Dette som et resultat av at de ikke trenger memorere informasjon og stille oppfølgingsspørsmål. De kan lese kursnotasjoner samtidig som de blir lest opp av assistent.

Spesielt interessant observerte testholderne at:

- Til tross for at vi har implementert trådløse knapper, som tidligere testpersoner hadde forespurt, ble bare stemme-kommandoer benyttet under testen.
- Stor variasjon på hvor bra testpersonene klarer benytte systemet fra teststart

5.5 Måloppnåelse

Vi har i denne oppgaven valgt å dele opp måloppnåelse i fire nivåer. Der vi definerer høy måloppnåelse som at vi har oppnådd det vi satt som målsetting, middels som at vi delvis har fått til det vi satte som målsetting og lav som at vi har prøvd og fått det til med begrensninger. Ingen betyr at vi har forsøkt, men ikke fått til å implementere i programmet.

Høy måloppnåelse
Middels måloppnåelse
Lav måloppnåelse
Ingen måloppnåelse

Basert på en egen evaluering av prosjektet har vi kommet fram til følgende resultat:

Minimumskrav:

Mål	**Evaluering av oppnåelse**	**Kommentar**
Vise navigasjonsdata i brillene	Høy	Viser kursnotasjon, tid, avstand, kurs samt fartøyets fart og kurs
Være enkel i bruk	Middels	Krever noe opplæring, personlig tilpasning og erfaring.
Vise hensiktsmessig informasjon	Høy	All informasjon som blir presentert i frontpanelet blir benyttet. 45 og 90 markører blir benyttet

Bør ha krav:

Mål	**Evaluering av oppnåelse**	**Kommentar**
Motta data trådløst	Høy	All dataoverføring skjer trådløst via lokalnettet mellom server og klient. God rekkevidde og stabilt.
Bruke stemmestyring/ håndkontroller til å interagere med systemet	Middels	Stemmestyringen oppleves for noen kronglete. Har kun virtuelle knapper, ingen fysiske. Bruker ingen håndgester. Programmet kan betjenes problemfritt med stemme.
Vise alarmer/meldinger som er relevant for en navigatør	Lav	Kan kun vise og lese opp melding fra serveren. Har ingen meldingslogg. Forsøkt å lage en XTE-alarm. Får ingen alarmer fra broen
Markere og interagere med objekter i rommet	Ingen	Dette er realisert, men kun ved selvstendige tester. Mixed Reality Toolkit og Vuforia hadde kompabilitetsproblemer som må jobbes mer med for å implementere i ett fullverdig system.

6 Drøfting

6.1 Valg av brillesett

I dette prosjektet ble det brukt Microsoft sine HoloLens briller. Begrunnelsen for dette er todelt. Første argumentet tar for seg at augmented reality briller er svært dyre, og med et begrenset bachelorbudsjett er det vanskelig å få plass til briller i budsjettet. I forlengelsen av dette er det også utfordrende å få tak i de ulike brillesettene, og med rammeavtaler i Forsvaret er det ikke alle leverandører av elektronikk som tilbyr AR-briller. Dette resulterte i at når vi fikk muligheten å låne briller fra Fostech, ble det en svært god løsning for oss.

Vi mener uavhengig av dette at Microsoft HoloLens sine briller var det optimale brillenesettet til vårt bruk. Vi vurderte også andre brillesett som Magic Leap og Vuzix Blade. Magic Leap behøvde en ekstern datamaskin sammenlignbart med en gammel Discman og Vuzix Blade hadde ingen 3D-funksjonalitet. Ingen av de har håndgester. På den andre siden finnes det både lettere briller og brillesett med videre synsvidden som begge er funksjonaliteter som hadde vært satt pris på. I fremtiden vil nok Microsoft HoloLens II være et mer optimalt brillesett ettersom dette forbedrer mange av svakhetene til HoloLens, den bedrer ergonomien, reduserer vekten og utvider synsvidden i tillegg til å øke prosessorkraften. Til slutt mener vi at å utvikle en UWP-applikasjon, mot HoloLens, er hensiktsmessig ettersom konseptet vårt muligens vil være enklere å overføre til det som ser ut til å bli det beste produktet tilgjengelig på markedet, HoloLens 2. Ved prosjektstart var ikke disse lansert og er per desember 2019 ikke tilgjengelig i Norge. Dette resulterte i at vi valgte Microsoft HoloLens til dette prosjektet.

6.1.1 Microsoft HoloLens

Av kritikk til brillesettet er det i hovedsak tre ting vi mener burde kommenteres. Det første er at brillene bare har et synsfelt på 30° x 17.5°, noe som kan oppleves som begrensende for mange. Dette gjør at dersom man har informasjon som spenner seg utover dette synsfeltet, må man bevege på hodet for å kunne se all informasjonen. Spesielt dersom man står for nærme. Det kan sammenlignes med å lese en tavle igjennom et nøkkelhull. Det kan argumenteres for at det ikke er nødvendig med større synsfelt, da den viktigste informasjonen får plass i synsfeltet. Argumentet er valid i den grad at det funker greit for dette prosjektet, men med en liten endring av brukergrensesnittet vil det bli kunne bli en begrensende faktor. Det er også viktig å tenke på at brukergrensesnittet er laget for nettopp denne synsbredden. Ved en større synsbredde ville vi hatt flere muligheter, og færre begrensninger. Vi ønsker også å kommentere batteritiden til brillene. Ved bruk holder ikke brillene lengre enn høyst to timer med høy

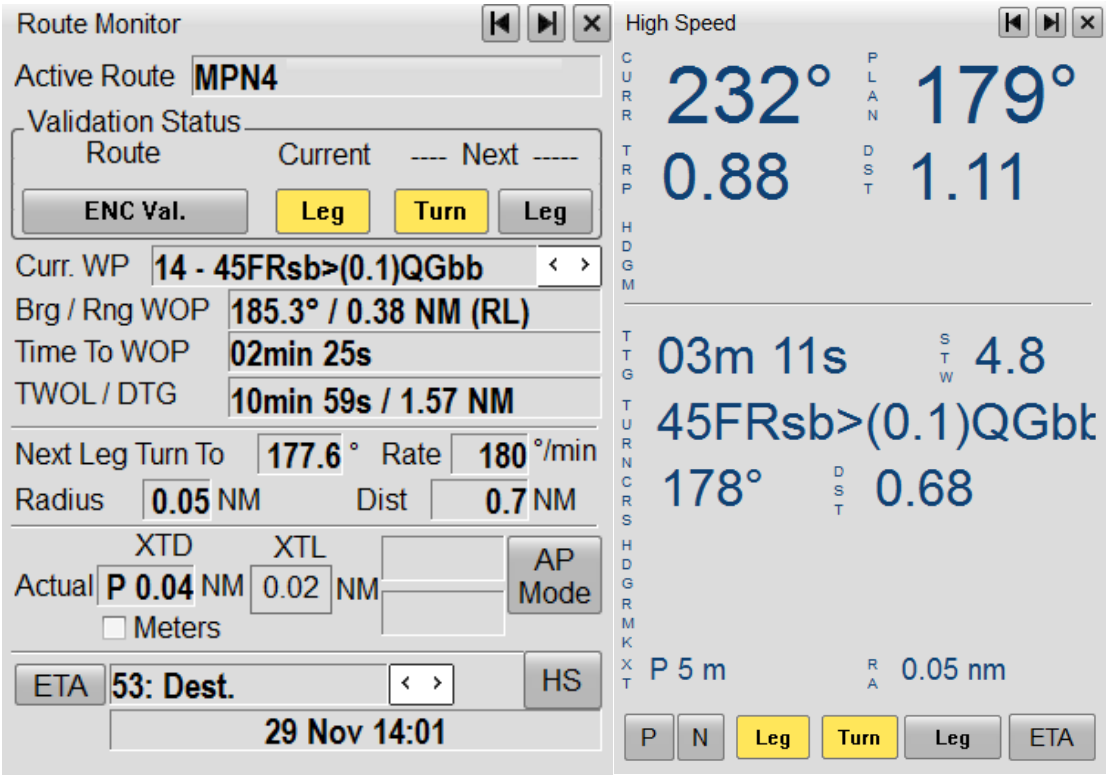
belastning. Man kan selvfølgelig ha flere briller som man bytter mellom, men det er både ressurskrevende og lite sømløst. Motargumentet er helt klart at mer batteritid, krever et større batteri, noe som igjen vil kunne påvirke ergonomien til brillesettet og som leder oss inn til den siste bemerkningen. Når det kommer til ergonomi, er brillesettet for noen ubehagelig å bruke over en lengre periode. Brillene oppleves som relativt tungt og klumpete – spesielt ved dårlig tilpasning fra brukerens side. Det kan sies at for noen vil det en tilvenningssak, mer enn et problem med designet.

6.2 Design av AR-brukergrensesnitt

Dette kapittelet kommer til å ta for seg vurderinger gjort under utviklingen av AR-brukergrensesnittet. Det vil gjort rede for hvordan de ulike design-valgene ble slik de ble. Det vil bli gjort rede for vår inspirasjon, prototypens utforming, samt begrunnelse for våre valg. Videre vil vi gå inn på widgets, gester og kommentere bruken av sidepanelet. Til slutt vil vi drøfte bruken av interaksjon med virtuelle objekter.

Figur 73 Augmented reality løfter informasjon opp fra panelene..

6.2.1 Inspirasjon



Figur 74 Venstre: ECIDS rutemonitor. Høyre: Høyhastighetsversjon.

Som beskrevet i *kapittel 3.1* har vi brukt flere forsøk og gjentatte tester for å komme frem til det endelige brukergrensesnittet. Brukergrensesnittet har hentet stor inspirasjon fra ECDIS sitt rutemonitoreringsverktøy som blir brukt av Sjøforsvaret. Videre er det også hentet inspirasjon fra ECDIS sitt høyhastighets-rutemonitoreringsvindu. Vinduet kan på en måte ses på som en videreføring av det originale rutemonitoreringsverktøyet - og vårt program som en videreføring av dette igjen. Vår påstand er at AR-brillene vil ha størst nytte om vi løfter nettopp dette vinduet til synsfeltet. På den andre siden er det ikke alltid nok å kopiere allerede etablerte brukergrensesnitt og implementere de i et nytt prosjekteringsmedium. Vi har gjennom gjentatte tester erfart at det kreves både større og mindre modifikasjoner underveis for å få informasjonen til å flyte på en hensiktsmessig måte. En av de mest tydelige endringene fra høyhastighetsvinduet er at vi har splittet kursnotasjonen på midten. Dette gjorde vi etter en tilbakemelding ved første test som omtalt i *underkapittel 5.4.1*. Dette på grunn av at vi som har utviklet prototypen ikke er navigatører, men også på grunn av at et nytt produkt, krever nyskapning og egne modifikasjoner for at det skal fungere optimalt.

6.2.2 Farger, dybde og størrelse

Et annet aspekt som har vist seg å være overraskende utfordrerne med dette prosjektet er valg av farge, dybde og tekststørrelse. Disse tre faktorene er svært viktig for å få produktet til å føles rett, samt hvor effektivt man klarer å lese av informasjonen som ligger i synsfeltet. Som ingeniørstudenter er det utfordrende for oss å ta stilling til hvilke farger som skal brukes i brukergrensesnittet. Dette begrunner vi med mangelen på undervisning med relevant teori, i kombinasjon med manglende erfaring med praktisk navigasjon. Til sammen gjør dette det utfordrerne for oss å velge optimale farger for et system vi ikke selv skal bruke. Samtidig kan det argumenteres for at litteraturen er relativt enstemmig i at grønn er den optimale fargen for HMD (Wood og Howells, 2001, s. 27). Dette både på bakgrunn fra forsking innenfor flyindustrien, men også med tanke på at grønn-farge er den mest lyssensitive fargen for øyet (Mollandsøy og Pedersen, 2017, s. 26)[11]. Vi mener det er viktig å anerkjenne viktigheten av fargevalg. Navigatørene på sjøen navigerer ofte i varierende miljø, både med tanke på vær- og lysforhold. De er som regel avhengig av å kunne kjenne igjen lanterner, sjømerker og andre objekter som blir gjenkjent på bakgrunn av deres farge og takt. Dersom valg av farger ikke blir gjort på en kløktig måte basert på anerkjent litteratur, samt praktiske erfaringer og operative hensyn, kan dette få katastrofale konsekvenser ved en eventuell implementasjon av slik teknologi. Ettersom vi fant lite litteratur på fargevalg i ett maritimt miljø der forholdene ofte variere mer enn i luftfart, som nesten all litteratur baserer seg på, har vi valgt å gi brukeren av prototypen mulighet til å velge farge selv, men med grønn som utgangspunkt. Dette på grunn av at vi anerkjenner at:

1. Det er brukeren som skal bruke verktøyet som har det beste beslutningsgrunnlaget for å velge hvilken farge vedkommende er mest komfortabel med.
2. Maritime navigasjon foregår i et miljø som konstant endrer seg med årstidene og tid på døgnet. Dette gjør at det nødvendigvis ikke alltid er en farge som er optimal for alle situasjoner.

En annen sentral utfordring vi har måtte ta stilling til er hvordan dybden til informasjonen som skal vises i brillesettet. Brillene har mulighet til å vise tredimensjonale objekter og gir oss mulighet til å endre dybdeføleIsen. Den optimale dybden har vist seg å være utfordrende å finne. Opplevelsen av dybde er svært forskjellig fra person til person, og det er vanskelig å sette en

[11] Se Augmented Reality og Head Mounted Display - Navigatørens Verktøy i en Teknologisk Fremtid av Mats Kristian Mollandsøy og Peter Heistad Pedersen kaptittel 2.12 side 26-29 for teorigjennomgang

dybdestandard som skal passe for en stor andel mennesker. Selvfølgelig kan man argumentere for at ved å standardisere så mye som mulig, vil det bli lettere å implementere det i en operativ setting ettersom det vil kreve færre personlige justeringer ved bytte av bruker. Samtidig, så er oppfattelsen av dybde subjektivt. Ved å gi brukeren mulighet til å tilpasse dette selv, kan teknologien oppleves som mer naturlig og behagelig å bruke over lengre tid.

Skriftstørrelse på informasjon er sentralt for brukeropplevelsen. Man kan bruke mye av den samme argumentasjonsrekken som med dybde. En større skriftstørrelse vil fylle større deler av synsfeltet ditt, og fjerne fokuset fra det som ligger bak – den ekte verden. Lesbarhet er avhengig av flere faktorer, og størrelsen er en av disse. Vi opplevde at dersom teksten ble for liten, krevde det større konsentrasjon av brukeren for å lese teksten, spesielt dersom det var bevegelse i fartøyet. Dette ville sannsynligvis vært enda tydeligere om bord på ett fysisk fartøy da bevegelsene er reelle. Samtidig var brukeren i større grad avhengig av å stå stille. I motsetning var større tekst lettere å få med seg, både stillestående så vel som i bevegelse. Som følge av momentene over, vil slike parametere kunne ha en operativ konsekvens. Derfor mener vi at vurderinger rundt valgt av størrelse burde tas med større beslutningsgrunnlag som baserer seg både erfaring og testing. Til tross for dette har vi valgt å la brukeren kunne justere dette selv, med begrunnelse at erfarne navigatører vil kunne justere størrelsen slik de anser det som mest hensiktsmessig.

6.2.3 Plassering og utbytte av utplassert informasjon

Figur 75 Testperson bruker peilesøyle med brillesettet på.

Elementene våre er plassert ut på en gjennomtenkt måte. Det øverste tekstfeltet viser informasjon om nåværende fart og kurs. Tanken er at dette er dynamiske data som navigatøren raskt må kunne skille ut, og blir derfor presentert separat fra de andre feltene. Dersom man vet farten og kursen til et fartøy, vil det kunne fortelle mye om situasjonen til fartøyet en kort periode frem i tid. Vi har også muligheten til å legge inn andre dynamiske parametere som kan kommuniseres over NMEA-strenger. Eksempelvis: vindfart, posisjon i form av bredde- og lengdegrad og rorutslag. Vi mener at disse parameterne ikke gir tidskritisk og informasjon som vil være relevant å trekke frem i et så spisset produkt. Slik datafremvisning kan resultere i at det vil danne seg støy i synsfeltet fremfor mentalt overskudd. På den andre siden kan det godt tenke seg at disse parameterne kan implementeres i andre brukergrensesnitt til andre operasjoner. Det være seg for eksempel når fartøyet skal ligge seg til kai, ved en eventuell mann over bord situasjon, ved ubåtjakt eller ved havari. Her er det bare fantasien som setter grenser[12].

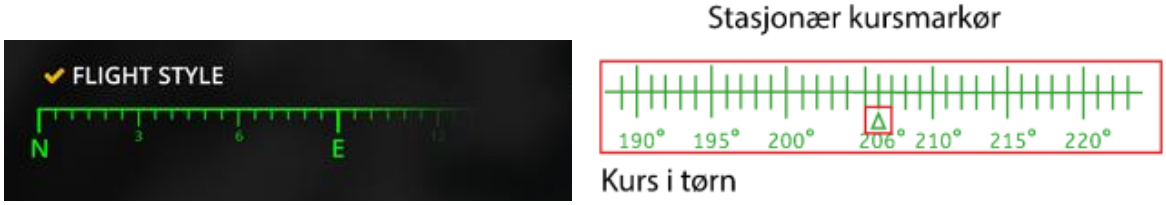
Tekstfeltet til venstre har tre linjer. Det skal vise hvilket siktepunkt fartøyet skal ha, hvilken kurs som skal holdes og gjenværende avstanden til turn. Dette tekstfeltet ble utviklet etter ønske og med tett samarbeid fra våre to første testpersoner. Denne informasjonen opplevdes som kjent, og brukerne var vant til å referere fra ECIDS sitt rutemonitoreringsvindu. Baktanken er at man skal få økt mentalt overskudd i situasjoner hvor man seiler fra veipunkt til veipunkt. I

[12]For videre lesning se Ocean Concept lab sin feltstudie på KV Svalbard.

tillegg til at mer av tiden brukes på å se ut, fremfor ned i ulike paneler. Hvorfor den er plassert akkurat til venstre er litt opp til tilfeldighetene. Det føltes naturlig å lese informasjonen «kronologisk» fra venstre til høyre. Det kan argumenteres for at hvorvidt man har inneværende kurs på høyre eller venstre side ikke har stor praktisk betydning. Etter det vi kjenner til, finnes det ikke noen etablerte standarder på dette. Med dette som utgangspunkt ville det nok kun ha blitt en tilvenningssak, uavhengig av hvilken side den var plassert på.

Med referanse til *underkapittel 5.1.1* vil tekstfeltet til høyre ta for seg informasjonen om neste tørn, samt neste leg. Videre vil ikke informasjonen, foruten tidsvariabelen, vises før det er mindre enn tre minutter til tørn som vist på *Figur D.66* i *vedlegg D.* Denne avveiningen om å ha informasjonen kontinuerlig tilgjengelig eller å la den vises ved behov, har blitt testet ved flere anledninger. Vi kom fram til at inntil det er tre minutter tørn, vil det være lite hensiktsmessig å vise turninformasjonen i brillene[13]. Vår mening er at man vil være mer tjent med færre forstyrrelser under lengre distanser.

6.2.4 Widgets og håndsignaler



Figur 76 Flightstyle compass bar fra Unity Asset store og tidligere bacheloroppgave[14].

Vi hadde store ambisjoner om å bruke widgets da vi startet prosjektet. Dette var i stor grad basert på at flere tidligere skildringer av maritim augmented reality har widgets. Flere av deres «ideelle» brukergrensesnitt bygges opp av widgets. Videre var vår ambisjon også farget av at flere av brukergrensesnitt man ser i både film- og spillverdenen aktivt benytter seg av widgets. Vi tenkte at slike moduler gjorde opplevelsen mer interaktiv, og datapresentasjonen mer gripbar og iøynefallende. Derimot fikk vi tilbakemeldinger på at dette ville oppleves rotete og ta oppmerksomheten bort fra informasjonen. I tillegg var navigatørene godt vant til å lese av for eksempel kurs kun ved å lese av kun tallverdien. Dette opplevde vi som interessant, og man kan på mange måte argumentere for at widgets vil kunne føre til økt mengde syntetisk støy. På den andre siden kan man tenke seg at slik informasjonsfremvisning ville gjort informasjonen lettere å prosessere - da den kan tolkes grafisk. Men til slutt er det som regel hva brukeren ønsker som

[13] *Det er vanlig at navigasjonsassistent oppdaterer navigatør når det gjenstår 3 minutter til tørn.*
[14] *«Augmented Reality og Head Mounted Display», Mollandsøy og Pedersen (2017).*

er avgjørende for designbeslutninger. Gjennomsnittet av våre undersøkelser forteller oss at navigatører ønsker et widget-fritt brukergrensesnitt.

En annen ting vi vurderte var muligheten for å interagere med systemet ved hjelp av gestikulasjon. HoloLens har innebygde gestikulasjoner som kan brukes til å utføre ulike funksjoner. Vi valgte etter hvert å unngå dette da vi anså dette som upresist og vanskelig å bruke for nye brukere. Dette med bakgrunn i at flere hadde problemer med å navigere i HoloLens sin egen meny – som vi anser som ganske simpel i bruk. I tillegg var det utfordrende å implementere *Mixed Reality Toolkit* sammen med *Vuforia*, som vi går inn på i *delkapittel 6.3.*

6.2.5 Kommentarer til sidefeltet

I *kapittel 5.1.2* ble sidefeltet presentert. En viktig presisering her er at for utenom videoavspillingen, siste melding og status på stemmeassistanse, så var de andre variablene statiske plassholdere for implementering av senere dynamiske verdier. Dette er et resultat av at ikke finnes NMEA-strenger fra simulatoren som inneholder drivstoffkapasitet og motorbelastning. Dersom vi hadde hatt tilgang til dette på et reelt fartøy, ville det kunne implementeres ved å utvide allerede implementert funksjonalitet. Samtidig kan man på mange måter se hvordan det kunne sett ut dersom man hadde hatt reelle data, og vurdere hvorvidt dette er hensiktsmessig eller ikke. Kallesignal, dypgang, lengde og toppfart, vil alle være statiske variabler som man kan endre selv, for eksempel i et konfigurasjonsvindu i Node-RED. Eventuelt til og med ulike forhåndslagre profiler.

6.2.6 Interaksjon med objekter

Vi hadde en målsetning, som nevn under *kapittel 1.2*, om å kunne markere og interagere med objekter i rommet. Denne målsetningen ble ikke oppfylt, da vi ikke fikk det til å fungere i det sammensatte programmet. Dette var et resultat av at slik funksjonalitet er avhengig av Mixed Reality Toolkit. Vi oppdaget tidlig at Mixed Reality Toolkit hadde kompatibilitetsproblemer med Vuforia, noe som resulterte i at vi ikke fikk det til å fungere. Videre anså vi Vuforia som viktigere for prosjektet, da Vuforia har vært grunnlaget for kalibreringen av alle elementers plassering. Derfor la vi gestikulasjoner til side. Samtidig har vi selv erfart gjennom jevnlig bruk av HoloLens, samt gjennom erfaringsutvekslinger med Ocean Industries Concept Lab at flytting av objekter kan være utfordrerne for brukere. Slik interaksjon stiller krav til at brukeren har god kjennskap til systemet. I forlengelsen av dette ser vi også for oss at det å flytte objekter og markører ville vært vanskelig på ett fartøy når det er utfordrende værforhold. I tillegg vil det kunne føre til at man gjør uønskede endringer på grunn av brukerfeil. Hypotetisk kan dette føre

til at mye tid, oppmerksomhet og ressurser blir bunnet opp for å utbedre feil, i motsetning til å skape overskudd. På den andre siden kan man argumentere for at en stor styrke med augmented reality er fleksibiliteten som det å kunne flytte ting i rommet gir. I tillegg til muligheten til å interagere med omverdenen. Uavhengig av dette har vi ikke fått tid til å implementere funksjonalitet slik at vi kan flytte objekter, men ville helt klart undersøkt disse kompatibilitetsutfordringene, og hvordan denne funksjonen hadde fungert og blitt opplevd i praksis dersom vi hadde tid.

6.3 Vuforia Engine

Vuforia kan på mange måter betegnes som fundamentet til prototypen vår. Dette på grunn av at hele den virtuelle verden i Unity bruker bildemålet i Vuforia som posisjonsreferanse. Dette manifesterer seg tydelig når man bruker bildemålet til å starte – og re-kalibrere brukergrensesnittet. Dette vises på *Figur 63* under *underkapittel 5.1.4.* Videre kan man argumentere for at vi har brukt Vuforia motoren på en ny måte. Motoren blir som regel brukt til mindre hologrammer i forbindelse med utvikling av IOS og Android apper (Library.vuforia 2019). Til tross for at vi har brukt motoren som en sentral brikke i systemet, har det vært relativt rett frem å implementere. Ettersom motoren er ferdiglaget, er den enkel å ta i bruk. På den andre siden har også Vuforia resultert i noen begrensninger. Den første begrensningen er at det ikke er mulig å strømme sanntidsvideo fra device manageren til HoloLens, på grunn av kompatibilitetsproblemer med Vuforia. Det er dog mulig å filme og ta bilder selv om man bruker Vuforia. Den andre begrensingen er at vi ikke har klart å bruke Mixed Reality Toolkit og Vuforia i samme prosjekt, da de har kompatibilitetsproblemer. Vi valgte å prioritere Vuforia ettersom den utgjør en så viktig del av systemet.

6.4 Bruk av Node-RED til programmering

Etter å ha brukt Node-RED i store deler av programmeringen for dette prosjektet har vi noen erfaringer angående verktøyet. Et problem med Node-RED er etter vår oppfatning at programmer kan bli ustrukturert og lite oversiktlig når prosjektene begynner å bli av større omfang. Spesielt sett i sammenligning med C# hvor man enkelt kan lage forskjellige klasser og funksjoner. Selvfølgelig er det mange ting man kan gjøre for å strukturere den visuelle programmeringen med tanke på ulike faner og systematisering av noder. Alt dette tatt i betraktning, vil vi argumentere for at tekstbasert programmeringsspråk vil stille sterkere på bakgrunn av de bidrar til mer oversiktlig programmering, i tillegg til at man har dypere kontroll over hva som faktisk skjer. Samtidig var det mye lettere å feilsøke i Node-RED og teste ut nye konsepter. Dette kommer av at JavaScript ikke krever kompilering. Som et resultat deployeres

ny kode til Node-RED i løpet av få sekunder til sammenligning med Unity og Visual Studio som til sammen bruker flere minutter på å kompilere og deployere kode. Dersom man ser for seg flere deployeringer per feil, vil dette fort bli mye bortkastet tid. Videre finnes det også langt mer dokumentasjon på etablerte tekstbasert programmeringsspråk. På den andre siden er Node-RED mye enklere å lære, og det tilbyr gode, ferdige verktøy for å knytte sammen maskinvare, APIs og nettverksbaserte tjenester og fungerer dermed bra til prototype utvikling. Siden Node-RED bygger på JavaScript, vil man med riktig kunnskap ha mange av de samme mulighetene som de utelukkende tekstbaserte programmeringsspråkene.

6.5 TCP eller UDP?

Vi ha valgt å bruke TCP som transport-lag for overføring av data i dette prosjektet. Dette er primært basert på det faktum at TCP er robust, samt at det kan sende data i begge retninger som beskrevet under *delkapittel 2.9.* Systemet vårt er avhengig av å motta rett data til rett tid, ettersom navigatørene ikke kan stole på systemet dersom det kommer feil data eller det oppstår kommunikasjonsproblemer. I tillegg til dette er vi avhengig av å kunne sende data i begge retninger for å kunne styre prosesser på serveren fra HoloLens. Dette resulterer i at TCP vil være en bedre løsning i enn UDP. På den andre siden kan man argumentere for at systemet trenger fart, noe som UDP i aller høyeste grad bidrar til. Ettersom variablene har en oppdateringsfrekvens i spennet 1-10 Hz fra det integrerte navigasjonssystemet, opplever vi at TCP har tilstrekkelig fart. Igjen kan man argumentere for at UDP er mindre ressurskrevende, men etter vår erfaring er ikke dette like nødvendig for oss, da bruken av TCP ikke har påvirket ytelsen til systemet i stor grad. Alt i alt ser vi at UDP, kunne være et greit alternativt, men vi mener at TCP er den mest optimale løsning for vårt system sett i lys av at robusthet og toveiskommunikasjon er viktigere enn ytelsen i vår prototype.

6.6 Hvorfor ikke bruke AIS-strenger?

I løpet av prosjektet ble en av de viktige avveiningene vi har tatt vært hvorvidt vi skulle bruke AIS–datastrenger eller bruke alternative datastrenger. I starten av prosjektet brukte vi kun AIS ettersom det var lett tilgjengelig. Problemet med disse strengene var at mye av dataen som ble sendt ikke var informasjon vi ønsket å bruke. Store deler av båndbredden ble brukt til å levere fartøyets ID, kallenavn og dimensjoner, noe som har svært liten relevans for oppgaven vår. Dette ble dog løst ved hjelp av et par linjer med JavaScript som filtrerer ut uønsket data. Fremdeles kan det argumenteres for at dette er ikke den mest optimale løsning, og sett i lys av at overføringsfrekvensen (da) ikke var høyere enn en hertz var det vanskelig å slå seg til ro med

AIS-strenger. Personell fra navkomp mente at det ville være bedre å bruke NMEA-strenger fra det integrerte navigasjonssystemet på bro. Hovedsakelig på grunn av at vi fikk tilgang til mer informasjon, samt høyere overføringsfrekvenser. Selvfølgelig kan man diskutere hvorvidt man trenger en frekvens på ti hertz, som vi benytter oss av nå ved projeksjon av navigasjonsdata, men med tanke på at vi fikk tilgang på mer informasjon vil dette øke fleksibiliteten til de ulike brukergrensesnittene. På den andre siden har AIS en styrke ved at det er lett tilgjengelig, og de fleste moderne fartøy, har AIS plugger som loser kan koble seg på. Selv om det ikke er umulig å bruke AIS-datastrenger til projeksjon av navigasjonsdata, vil vi argumentere for at det vil være en sub-optimal løsning med tanke på overføringshastighet og tilgjengelig informasjon. Som nevnt førte dette til at vi valgte å bruke NMEA-strenger.

6.7 Bruk av stemmekommandoer

Vi ønsket innledningsvis å benytte oss av både stemmestyring og knapper for å kunne interagere med systemet. Denne funksjonaliteten skulle gjerne vært testet mer. Derimot har det under testene våre enda ikke hendt at kommandoer har blitt aktivert uten at det var ønsket av brukeren. Sett vekk ifra aktive forsøk for å aktivere kommandoer i den hensikt å vise testpersoner deres funksjonalitet. Dermed er det rimelig å anta at det er ett relativt sikkert system. På den andre siden opplevde vi noen ganger at responstiden var noe treg. Denne forsinkelsen førte enkelte ganger til at brukeren ga kommandoen flere ganger etter hverandre, grunnet tregheten i systemet. Dersom brukeren gjorde dette, ble kommandoen registrert en gang for mye. Dette forbedret vi ved å øke frekvensen på informasjonsetterspørslene fra HoloLens. Det vil fremdeles være noe treghet fra man gir kommando til den blir utført, men dette er viktig for at stemmegjenkjenning skal forstå at brukeren er ferdig å snakke. En viktig grunn for at vi ønsket å implementere stemmestyring, var fordi vi mener at det utløser det fulle potensiale til AR-opplevelsen. Ved å bruke stemmestyring blir systemet helt frittstående. Man opplever å være uavhengig av alt annet enn dataoverføring. Opplevelsen blir en utvidelse av ens virkelighet. Dette har resultert i at navigatøren fysisk sett har mye større overskudd og oftere ledige hender da han kan utføre flere oppgaver fra valgfritt sted på bro ettersom han ikke trenger å gå bort til ECDIS konsollen for å lese og bytte informasjon som vises.

6.7.1 Stemmekommandoer versus knapper

Det vi opplevde under evalueringene våre var at stemmekommandoer trengte tilvenning ettersom det var uvant for navigatørene. Etter ønske fra testpersonene våre i del I implementerte vi virtuelle knapper på ett nettbrett. Disse ble derimot ikke brukt én eneste gang under del II av testene. Dette tolker vi dithen at stemmestyring er nytt for brukerne, og fungerer bedre etter

hvert som man blir vant til dem. I tillegg var nok brukerne i overkant skeptisk til slik styring, og så kanskje ikke gevinsten ved økt fleksibilitet til å løse andre oppgaver. Derimot har vi ikke valgt å fjerne de virtuelle knappene da det er en god reserveløsning dersom det oppstår feil i stemmestyringen - for å skape ett mer robust system. Vi kunne også bruke disse knappene til å sømløst hjelpe dersom en navigatør «*rotet seg bort*». Videre mener vi det er en god reserveløsning dersom det er mye lyd på bro, enten av at det er mye personell og snakking, eller for eksempel en videoavspilling. I en slik situasjon kan stemmegjenkjenneren fungere dårligere samt at det kan være uhensiktsmessig å snakke høyt til HoloLens i alle tilfelle.

6.8 Automatisk bytte av veipunkt og avstand til veipunkt

Funksjonen som endrer veipunkt automatisk, er en av de funksjonene vi mener ikke har vært svært vellykket. Til tross for at funksjonen gjør det vi ber den om, har vi selv erfart i samspill med testpersoner at den ikke fungerer optimalt i praksis. Dette på bakgrunn av at den bytter på en unaturlig måte, som kommer svært tydelig frem dersom en sammenlignet med ECDIS sin algoritme. Vi har ikke fått sett på ECIDS sin algoritme, men det kan tenkes, basert på våre erfaringer og testpersoner sine innspill at algoritmen tar utgangspunkt i tørnlinje. På den andre siden kan man argumentere for at denne funksjonen ikke er nødvendig for vårt system, og at ved hjelp av stemmestyring og dashbord vil brukeren ha tilstrekkelige muligheter for å bytte veipunkt ved behov. Ideelt så ønsket vi å synkronisere vår veipunktangivelse med ECDIS sin, men som nevnt som en begrensning under *kapittel 1.3* har vi ikke tilgang til ECDIS. Dersom man tenker på et ferdig produkt, vil vi helt klart anbefale at man enten: inkluderer *Kongsberg Digital* slik at man kan få tilgang til algoritmen (eventuelt tilgang til data fra ECDIS), eller fortsette med videre testing og utvikling slik at man kan designe en algoritme som kan matche ECDIS sin. Videre i dette delkapittelet vil det også være naturlig å ta for seg avstand fra fartøy til veipunkt algoritmen vår. Etter den første heuristiske evalueringen fikk vi tilbakemelding på at testpersonene ønsket seg avstanden fra fartøy til tørnpunkt, til fordel for veipunkt. Akkurat som med problemstillingen ovenfor er dette noe ECDIS selv regner ut, og vi har ikke klart å erverve oss denne algoritmen fra *Kongsberg Digital*. Formelen vi har kommet opp med, skal etter vår kjennskap stemme rent matematisk, men vi opplever at den ikke samsvarer med ECDIS sine data. Etter gjentatte forsøk har vi kommet frem til at det kanskje ikke er sikkert at matematikken vår stemmer, og at det kanskje må til enda mer testing før man får til en algoritme som fungerer i alle tilfeller. Til tross for dette mener vi at arbeidet ikke er forgjeves, og at dette vil kunne være et naturlig startpunkt dersom noen ønsker og jobbe videre med oppgaven i fremtiden.

6.9 Retning til brillesettet i simulatoren

Planen var å bruke kompass-sensoren til å gi headingen til brillene i forhold til den ekte verden. Tanken var at man skulle legge dette inn som et valgfritt virtuelt overlegg som kunne veksles av og på – som å ta peilesøylen opp og ned. På denne måten kunne navigatøren bruke brillesettet til å ta peilinger. Et problem med å teste i simulatoren, er at kompass-sensoren gir retning i den virkelige verden, men ikke simulatoren. Vi har sett på hvordan vi kan få riktig retning i forhold til simulatoren, men har ikke fått til et tilfredsstillende resultat. Dette kan gjøres ved å sammenligne kompassbrikken sin retning med fartøyets retning. Til tross dette mener vi at det kan tenkes at 45° og 90° peilingene, i samspill med muligheten til å bruke peilesøylen med brillesettet på, fungerer på en god måte. Videre vil understreke at dette vil kunne være et viktig verktøy, og at et videre arbeid bør ta dette til etterretning i en eventuell videreføring av prosjektet.

6.10 Den videreutviklende prototypemodellen

I *delkapittel 3.1* ble den videreutviklende prototypemodellen gjort rede for. På mange måter kan man spørre seg hvorvidt det er nødvendig med en slik modell på et relativt lite prosjekt, med så kort tidshorisont som dette. Tankegangen vår var at til tross for omfanget på oppgaven, ville en felles modell for hvordan vi ønsket å strukturere arbeidet gjøre det lettere å dra i samme retning. Samtidig mener vi at det vil kunne føre til et bedre produkt, og mer læring. Vi kunne tenke oss å jobbe på denne måten i fremtidige prosjekter. Dette på bakgrunn av at når arbeidsmetoden er satt, vil arbeidet bli mer strukturert. Økt grad av struktur vil igjen underbygge påstanden om at produktet blir bedre. I prosjektet har vi tilstrebet å bruke modellen i de fleste designprosesser. Eksempler på dette er måten vi designet brukergrensesnittet på. Vi startet med en grov prototype, og har kontinuerlig utviklet denne basert på erfaringer og tilbakemeldinger. Dersom vi ikke hadde brukt denne utviklingsmodellen, hadde vi kanskje startet fra begynnelsen hver gang vi skulle utvikle brukergrensesnittet. En erfaring med slik videreutvikling er at det blir vanskelig å se tilbake på tidligere versjoner, da de glir inn i hverandre uten klare skiller. Som en forlengelse blir det vanskelig å se når designvalg blir tatt. Andre konkrete eksempler på dette er utviklingen av bøylen som holder kompass-sensoren. Bøylen ble ikke optimal den første, andre eller tredje gangen. Fremfor å starte på nytt, ble det heller gjort justeringer på den eksisterende bøylen. Til slutt fikk vi da en prototype som fungerte slik vi hadde sett for oss - basert på det originale designet. Hvorvidt dette alltid er den riktige fremgangsmåten å bruke er lite sannsynlig, og man må anerkjenne det faktum at noen ganger må man starte på nytt og gjøre noe annet.

6.11 Valg av modell for testene

Vi har som nevnt valgt å benytte oss av resultatene fra Nielsen og Landauer sin studie. Som et resultat av dette har vi derfor ønsket å ha minst 5 testerpersoner, ettersom det er i det området man oppnår høyest nytte i forhold til kost som man ser i *Figur 18*. Derimot om man ser på *Figur 17* får man med 5 testpersoner gjennomsnittlig kun avdekket i underkant av 75% av feilene. Siden vi hadde 6 testpersoner, vil vi da ifølge deres modell kun ha avdekket i overkant av 75% feilene. Disse tallene er som beskrevet tidligere basert på et gjennomsnitt. Ettersom vi har et system med relativt få funksjoner og dermed også færre muligheter for feil, kan tallet for oppdagede feil muligens være noe høyere. Videre et begrenset antall navigasjonskyndige testpersoner, samtidig som vi har begrenset tilgang til simulatoranlegget. Dermed har vi konkludert med at å ha høy kost/nytte effekt var hensiktsmessig for totalresultatet av oppgaven. Videre, ettersom vi har utviklet en prototype, vil det være naturlig å gjøre grundigere tester før et eventuelt sluttprodukt blir ferdigstilt. Dersom noen i fremtiden vil jobbe videre med oppgaven, vil vi anbefale og gjøre flere tester, da vi anser dette som viktig for produktutviklingen.

6.12 Kommentarer til de heuristiske evalueringene

I dette kapittelet blir det gjort rede for forskjellen på testgjennomføringene samt våre erfaringer og vurderinger vedrørende gjennomføringen av de praktiske testene.

6.12.1 Praktiske tester: Del I

Av feilkilder under denne evalueringen, er det i hovedsak to ting som burde nevnes. Vi hadde ikke et optimalt antall testpersoner i henhold til Nielsen og Landauer sin modell. Det kan dog argumenteres for at det siden dette er første test, og vi hadde forutsett en rekke feil, ville det vært lite hensiktsmessig med flere testpersoner ettersom det var ønskelig å utbedre de feilene vi fikk fra første tilbakemelding før vi fikk nye meninger om prototypen. Den andre feilkilden er at testpersonene fikk stille en rekke spørsmål, og observatørene var mer delaktig i undersøkelsen enn det Nielsen anbefaler i sin litteratur (Nielsen 1994). Det kan også nevnes at under den første testen var primærfokuset å få tilbakemeldinger på relevante verdier og oppsettet til brukergrensesnittet.

6.12.2 Praktiske tester: Del II

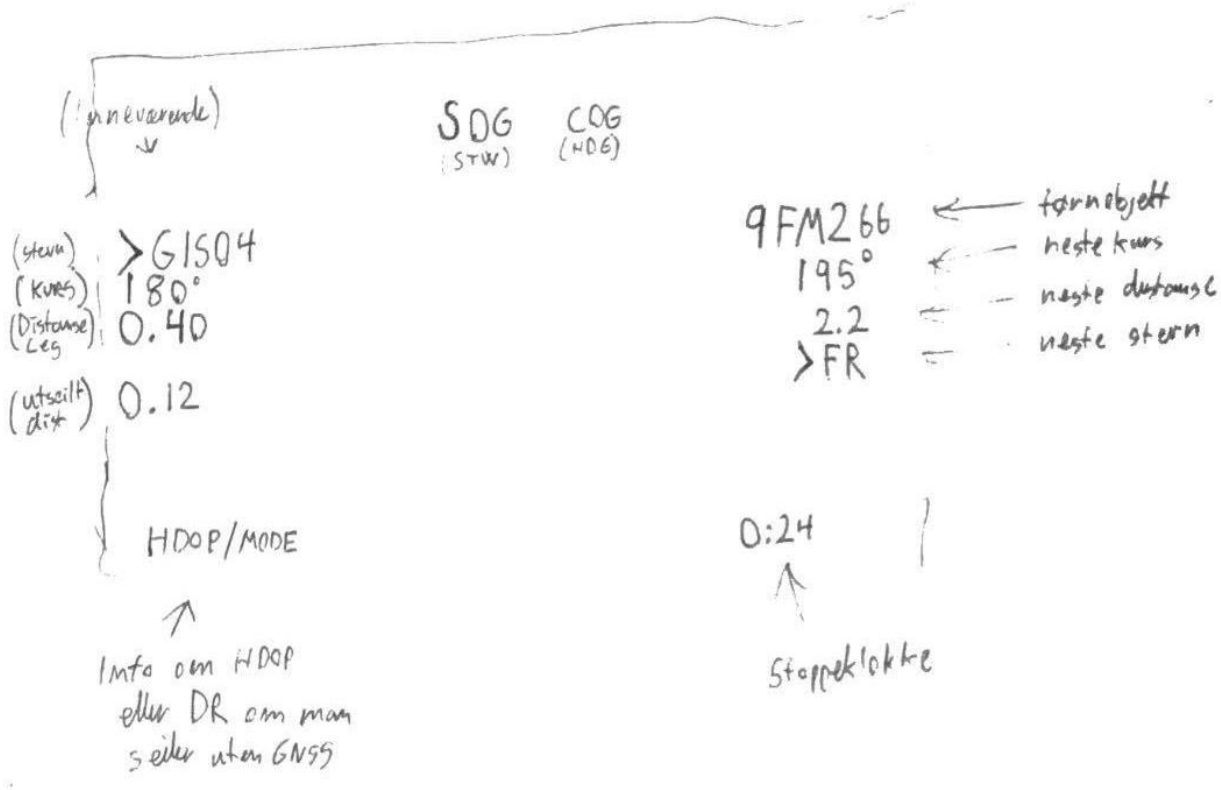
Av kritikkverdige ting som burde nevnes er at testene ikke ble holdt på identisk måte. Gjennomgangen av systemet var ikke formelt skrevet ned slik at den ble gitt på samme måte under begge gjennomføringene. Videre ble det også fikset noen mindre feil i systemet som ble

oppdaget fra gjennomføring en til to, noe som resulterer i at produktet de testet ikke var likt. Dette er dog prinsippet bak den rapide prototyputviklingen. Involveringen fra testholdere var også varierende fra de to gjennomføringene. Under denne testen var fokuset i større grad rettet mot å teste systemet sin funksjonalitet, å oppdage mindre feil. Dette på grunn av at utformingen til brukergrensesnittet i mye større grad var fastsatt, og testpersonene fikk ikke like stor kunstnerisk frihet som den første testen.

6.12.3 Evolusjon som et resultat av brukertester

Figur 77 Drøfting angående løsninger etter test.

En av tingene vi har vært opptatt av gjennom prosjektperioden er at utviklingen av prototypen skal i stor grad være et resultat fra tilbakemeldingen fra de ulike testene. Dette på grunn av at vi har valgt den videreutviklende prototypemodellene som arbeidsmetode, men også på grunn av prototypen da vil forme seg etter brukerens ønsker. Dette kan eksemplifiseres med samtalene vi hadde med noen av testpersonene etter den første testen. Etter de hadde fylt ut spørreskjemaet, fikk testpersonene lov å tegne sine tanker på papir, som vist i *Figur 78*. Vi realiserte deretter noen av forslagene i Unity, slik at de fikk se hvordan tankene deres vil se ut i brillesettet.



Figur 78 Et brukergrensesnittforslag fra testperson.

6.13 Implementasjon på fartøy

Det skal i teorien være enkelt å koble vårt system til et ekte fartøy da vi er koblet til samme tilkoblingspunkt som en los ville brukt. Det viktigste vil være å motta NMEA-strenger fra fartøyet. Vi har dessverre ikke testet på marinens fartøyer, men gitt at man får fysisk tilgang til NMEA-busen på fartøyet, skal det være fult mulig å lese av data. Grunnen til at dette vil kunne fungere, er på bakgrunn av at simulatoren bruker samme NMEA-standard som de andre fartøyene. Når det er sagt, kan det hende at vi må endre på baudraten til USB-noden, da simulatoren etter hvorvidt oss bekjent ikke bruker standard baudrate. Gitt at dette fungerer og man får tilgang til dataene, vil all databehandling være lik. Implementering vil kreve at man tilpasser brukergrensesnittet til det valgte fartøyet ettersom vårt brukergrensesnitt er designet spesielt for småbroene i simulatoranlegget. I tillegg må kalibreringsbildets plassering bli definert på nytt. Oppsummert ville det sannsynligvis ikke vært noe problem å implementere systemet på et ekte fartøy. Utfordringen ville potensielt vært å koble opp seg på fartøyet og få kalibrert brukergrensesnittet etter det valgte fartøyet.

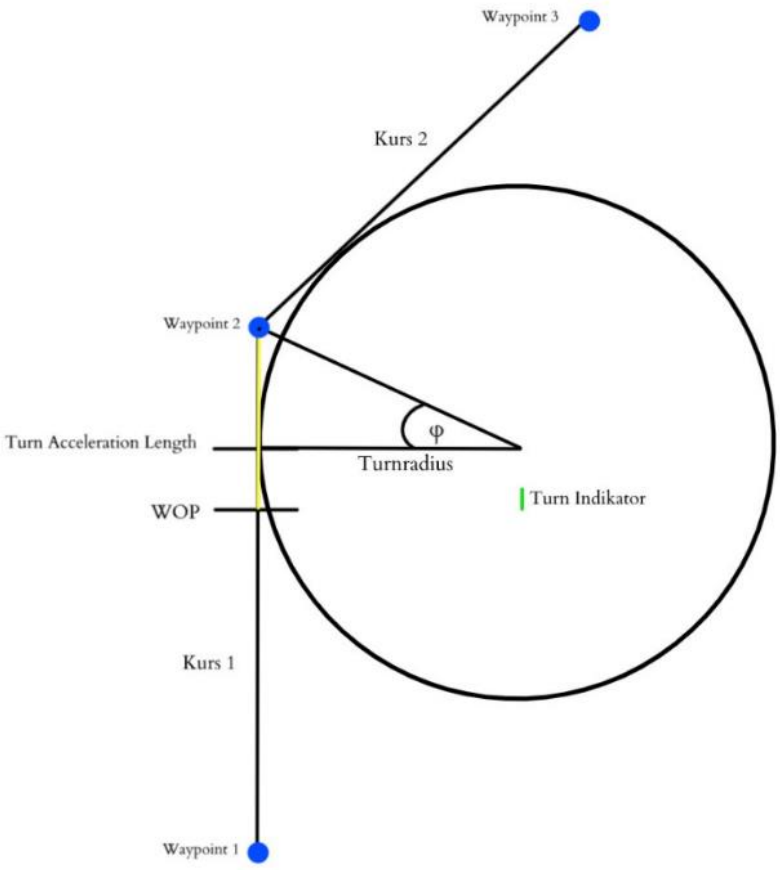
6.14 Ubrukte funksjoner, feil og mangler

Det har blitt utviklet en funksjon som kan lese opp informasjon om neste turn og leg. Stemmen som brukes er fra Microsoft Azure sitt stemmebibliotek. Det måtte legges inn et valg om å skru denne stemmen av og på, da det ble gitt tilbakemeldinger om at dette oppleves som unødvendig støy for flere av navigatørene under testene. Til tross for dette kan det tenkes at en slik funksjon kan brukes ved lengre seilas, der tiden mellom turn er betraktelig større. Informasjonen som blir lest opp kan også bearbeides ytterligere, og vi har fortsatt troen på at slik opplesning kan bli benyttet i et sluttprodukt.

Andre ubrukte funksjoner er blant annet muligheten til å bytte scener. Dette innebærer at man kan bytte ut eksempelvis frontpanelet med andre brukergrensesnitt. Tankegangen her er at man kan bruke systemet på andre fartøy og til andre operasjoner. Programmeringsmessig har vi fått dette til, men vi har ikke produsert andre scener vi kan bytte til og denne funksjonen vil da forbli ubrukt.

Av feil og mangler kan det nevnes at den presentere kursen i frontpanelet er basert på «*course over ground*». Den vil da ikke samsvare med "*true heading*". Til tross for dette har vi gjennom prosjektet fått anbefalt å bruke kurs over grunn fremfor sann kurs i brillene. Videre kan det nevnes at videospilleren-vinduet vil være grått fra start, og vil være synlig om man velger å se på det. Det grå «lerretet» vil vises fra start helt til "*remove video*" stemmekommandoen blir gitt. Vi fikk ikke til å skru av videoskjermen ved oppstart, og siden dette var en av de siste funksjonalitetene vi implementerte – ikke tid til å undersøke videre. Videre er det ikke mulig å endre hvilken video som avspilles dynamisk, da denne er forhåndsprogrammert inn i kildekoden. Dette kunne dog blitt løst med en «*URL*»-variabel.

Andre feil som kan være naturlig å nevne er avstanden til tørnpunkt. Testpersonene ønsket avstanden fra fartøyet til tørnpunkt, fremfor veipunkt, ettersom det var her de startet tørnet. Rent matematisk burde ikke dette vanskelig å få til, men vi klarte det heller ikke. På neste siden vil begrunnelsen på avstand til tørnpunkt gjøres rede for.



Figur 79 Geometrien i en tørn.

Geometrien i et tørn er relativt rett frem, og man skal i utgangspunktet bare trenge å ta avstanden til veipunktet og trekke fra det gule linjestykket som er markert på *Figur 79* Geometrien i en tørn.

Det gule linjestykket kan defineres som:

$$Gul\ linje = turn\ acceleration\ length + turnradius \cdot \tan\left(kurs_2 - kurs_{1)}\right)\frac{1}{2}$$

Erfaring viser at dette ikke fungerer i praksis ettersom resultat ikke stemmer overens med ECDIS sine data - selv med litt godvilje. Det har blitt testet med ulike kombinasjoner av turn acceleration length med ulike fartøy, men har ikke fått formelen til å stemme med ECIDIS sine verdier. Om ECDIS bruker en annen erfaringsbasert algoritme for å regne denne avstanden vet vi ikke, men det kan tenkes at *Kongsberg Digital* har gjort en rekke tester for å justere

algoritmen. Hvorvidt oss bekjent er også formelen ovenfor matematisk korrekt, men fungerer da ikke tilsvarende. Dette er noe som helt klart burde testet videre, ved et eventuelt videre arbeid.

Vuforia har gjort at det har oppstått et par begrensninger. Den første er at det ikke er mulig å strømme «*synet*» fra HoloLens til en datamaskin gjennom Mixed Reality Capture når Vuforia benyttes. Det er derimot mulig å ta bilder og filme. Det har på grunn av dette ikke vært ett stort problem. Det andre problemet, som nevnt tidligere, er at vi støtte på kompatibilitetsproblemer ved bruk av både Mixed Reality Toolkit og Vuforia i samme prosjekt. Dette problemet «*unngikk*» vi da ved å ikke bruke Mixed Reality Toolkit ettersom Vuforia allerede var implementert i oppgaven som en sentral del. Sannsynligvis ville det ført til merarbeid dersom vi skulle ha funnet alternative løsninger hvor både MRTK og Vuforia ble anvendt.

6.15 Begrensningenes påvirkning

Det er helt klart at mangelen på tilgang til data fra ECIDS sitt rutemonitoreringsverktøy har påvirket våre løsninger. Vi ønsket i utgangspunktet ikke å håndtere veipunktene manuelt, men som et bevis på konsept fungerer det utmerket. Vi har støtt på utfordringer når det kommer til å samkjøre verdier som vises på broen og i brillene. For eksempel viser våre avstandsberegninger en litt annen avstand enn det ECDIS gjør, men våre kurser er helt korrekt. Det kan også være at navigatøren glemmer å bytte veipunkt i brillene. Når dette skjer stemmer ikke veipunktet og dataene mot det tilsvarende i de fysiske panelene. Dette kan igjen skape frustrasjon og øker terskelen for implementasjon. I teorien vil det bare være å koble ECDIS på vår logikk, og programmet vil fungere mer sømløst.

6.16 Egne tanker og refleksjoner

Etter å ha testet augmented reality i forbindelse med navigasjon, ønsker vi å legge ved noen tanker vi har gjort oss under perioden. Vi er absolutt positive til bruken av augmented reality, men føler det er et par punkter som er viktig å ta stilling til når det gjelder utvikling og implementering av teknologien. Tekst-til-tale er et kraftfullt verktøy, med stort potensiale. Fra et ingeniørperspektiv er det flott å kunne bruke en stemme til å fortelle brukeren data. Det er dog lett å glemme det psykologiske aspektet ved at datamaskinen mater navigatøren med noe som kan sammenlignes med ordre. Eksempelvis kan vi få HoloLens til å fortelle navigatøren at vedkommende skal begynne å tørne om 30 sekunder. Hvem er det da som styrer fartøyet? Den erfarne navigatøren eller ingeniøren bak den enkle algoritmen? Før en eventuell implementasjon er det viktig å teste i hvilken grad navigatøren lar seg påvirke ukritisk til

informasjon de får fra systemet. Samhandling er et annet aspekt vi ønsker å trekke frem som en bekymring. Vi har ikke gjort tester på hvordan det vil være å samarbeide dersom mer av informasjonen blir løftet til den virtuelle verden. Det vil potensielt bli vanskelig å «*peke*» og vise i det fysiske domenet hva man referer til ettersom informasjon er fordelt mellom den fysiske og den virtuelle verden. Det blir sammenlignbart med å gi skildringer over telefonen. Det er vanskelig å beskrive ting når ikke alle i rommet har samme referansepunkt, den fysiske verden. På denne måten kan det hende at augmented reality-teknologi vil kunne motvirke dannelsen av felles situasjonsforståelse.

6.17 Augmented Reality - prematurt eller implementertbart?

Til tross for at vi ikke har utviklet et system som fungerer hundre prosent og kan implementeres øyeblikkelig, ser vi fremdeles store muligheter for teknologien i fremtiden. Med det sagt, trengs det et bedre brillesett før AR-teknologi kan benyttes operativt. HoloLens har en rekke utfordringer med tanke på at brillesettet er relativt tungt, har et innsnevret synsfeltet og kort batteritid. I en operativ kontekst er man avhengig av å bruke produktet over en lengre periode, med en lang batteritid. Dette fører til at vi anser det som relevant å utvide batterikapasiteten i fremtiden. Dette vil på den andre siden kunne påvirke tyngden til brillene. Det kan tenkes at mulige løsninger kan være batteripakker festet i beltet eller klærne, batteriryggsekker eller kablede briller. Det kan også løses med ett rulleringssystem der man for eksempel, har flere brillesett på ladning, som kan rulleres på en relativt sømløs måte, slik at systemet av flere brillesett kan brukes over lengre tid.

Videre er ergonomi en viktig problemstilling når det kommer til fremtiden for teknologien. Et tungt og klumpete brillesett vil være ukomfortabelt for folk å bruke. Det kan tenkes at det vil kunne kan være forstyrrende for navigatøren. Dersom navigatøren eller andre brukere må anstrenge seg, eller opplever at brillene er ubehagelig, har vi erfart at brillesettet slutter å være et verktøy. Eksempler på dette er når navigatøren under testing midlertidig tar av seg brillene, eller løfter de opp på pannen for å få en kort pause. Det er med dette som utgangspunkt viktig at brillene i fremtiden oppleves som behagelige å bruke over lengre tid, samt at det har gode justeringsmulighet for individuell tilpasning.

En annen ting vi ser på som relevant i fremtiden er muligheten for større samspill mellom den virtuelle og reelle verden. En erfaring vi ha gjort oss gjennom prosjektet er at teknologien går fra å være et nyttig, til å være revolusjonerende når man får muligheten til å interagere med omverden. Det å kunne markere fartøy man ser, markere staker, få opp AIS-data fra fartøy man

ser på eller ta ut peilinger med øynene, er alle fremtidige funksjoner som kan være svært nyttig for navigatøren. Det finnes også andre som forsker på akkurat dette. Ocean Industries Concept Lab gjennomførte våren 2019 en feltstudie under KV Svalbard sitt tokt til Svalbard (Eikenes, Mallam og Fauske, 2019, s. 3). Herunder var bruk av AR-teknologi i et arktisk og maritimt miljø en av tingene som ble forsket på (Eikenes et al., 2019, s. 2). I ettertid ble det publisert en erfaringsrapport, som kommer frem til en rekke funn og erfaringer som vi kan relatere oss til, men også noen nye betraktninger som vi ikke har tenkt på. Av funn som vi anser som relevante for oppgaven er det å vise frem planlagt ruter i tre dimensjoner, vise «*no-go*»-soner[15], vise retningsvektorer på andre skip og vise fyrtårn-soner (Eikenes et al., 2019, s. 2). Selv om mange av disse funne ble gjort på et Kystvaktfartøy i et arktisk klima, vil erfaringen også være relevant innenfor innaskjærs navigasjon. Dette på grunn av at mange av de samme informasjonsutfordringene som forekommer under navigasjon under arktiske forhold, også finnes i innaskjærs navigasjon.

6.18 HoloLens II - Er fremtiden her?

Figur 80 HoloLens 2.

Siden vi gjennom dette prosjektet har basert oppgaven på Microsoft sine første generasjons HoloLens briller, kan det være relevant å nevne at HoloLens II er blitt lansert når denne oppgaven er publisert. Til tross for at vi ikke har fått testet brillene selv, skal mange av utfordringene belyst i forrige delkapittel angivelig være løst. Av relevante oppgraderinger fra forrige versjon er økt prosesseringskraft, utvidet synsfelt, bedre batteritid og et lettere brillesett med en bedre ergonomisk utforming (Microsoft, 2019). Det kan med bakgrunn i dette tenkes at brillesettet vil kunne være et steg nærmere å brukes i den operative enden av sjøforsvaret, med forbehold at utfordringene faktisk har blitt løst. Uavhengig av dette kan det argumenteres for at teknologien er på vei i riktig retning og med en rask utvikling mot fremtiden. Dette gjør at fremtiden til både HoloLens og andre AR-brillesett vil bli spennende å følge.

[15] Steder man ikke burde seile basert på dybde eller tykkelsen på isen.

6.19 Videre arbeid og relevante erfaringer

Ettersom vi synes dette var et svært spennende tema å fordype seg i, har vi en rekke tanker og anbefalinger angående videre arbeid. Dersom du ikke er interessert i videre arbeid, kan du hoppe over dette delkapittelet.

Kanskje en av de viktigste erfaringene vi har gjort er anskaffelse av brillesett. Uavhengig av når det videre arbeidet foregår, er vår anbefaling at man burde prioriterer å få tak i så moderne briller som mulig, med god dokumentasjon fra både utgiveren, men også utviklere. Dette på grunn av at man da vil få færre teknologiske utfordringer. Dette vil kunne gi en bedre forståelse av mulighetene til teknologien, samt at nyere brillesett naturligvis vil ha flere funksjoner. I forlengelsen av dette følger programmering- og utviklingsmiljøene rundt augmented reality utviklingen, og at det på den måten er lettere å finne relevante biblioteker, ressurser og brukerinstrukser dersom man har et mer moderne brillesett.

En viktig lærdom vi har gjort oss gjennom prosjektet, er viktigheten av å utforske og lete etter ny kunnskap og erfaringer. Gjennom prosjektet har vi deltatt på Norshipping 2019, Media City Bergen sin *future week og* AR-workshop i regi av Ludenso. I tillegg til dette har vi titalls mailtråder med relevante aktører innenfor fagfeltet augmented reality i Norge. Gjennom denne kontinuerlige søken etter kunnskap har vi plukket små verktøy og lærdommer fra de ulike arrangementene vi har deltatt på, som vi igjen har inkorporert i prosjektet vårt. Til videre arbeid anbefaler vi at dette også gjøres, ettersom vi mener det resulterer i bedre forståelse av det overordnet tema for oppgaven. Dette på bakgrunn av at man får et innblikk i hva andre aktører i bransjen jobber med, fokuserer på og tenker om den nye teknologien. I tillegg til dette er det også etter vår erfaring en god mulighet til å få teknisk hjelp der det trengs. Vi har også fått føle på viktigheten av nettverksbygging og samarbeid. I forlengelsen av dette, selv om det for så vidt ikke direkte relevant for oppgaven, kan vårt utvidede nettverk innen miljøet være nyttig i fremtiden.

Andre erfaring vi har gjort oss er at det er viktig å få kartlagt verktøyene man ønsker å bruke til å utvikle med. Vi har i dette prosjektet brukt en kombinasjon av JavaScript, C# og Unity for å danne rammeverket rundt den utvida virkeligheten. Det er med utgangspunkt i dette viktig å så tidlig som mulig begynne å lære seg verktøyene, slik at man har kompetanse til å utvikle den funksjonaliteten man ser for seg. Vi brukte noen uker på å finne ut av hvilke verktøy som var hensiktsmessige å bruke, noe som gjorde at vi fikk kortere tid til å fordype oss i de valgte språkene og verktøyene.

Til slutt legger vi ved det som vi mener er relevant videre arbeid:

- Utvikle flere ulike brukergrensesnitt basert på ulike fartøy og operasjoner. Spesielt tenker vi da på et eget brukergrensesnitt til korvettene.
- Se på muligheten for å interagere med omverden, herunder muligheter for å kunne ta peilinger med brillene, markere objekter og få informasjon når man ser på ulike objekter.
- Utvide prosjektet slik at man kan hente ut data og bruke augmented reality på fysiske fartøy. Dette innebærer også at man burde se på hvordan ulike AR-briller orienterer seg. Utvikle et orienteringssystem som kan ta høyde for bølgenes påvirkning på fartøyet. Våre tanker rundt dette er at man kanskje må bruke eksterne sensorer plassert rundt på skipsbroen for å hjelpe brillesettet å orientere seg. Som for eksempel VR-settet HTC Vive gjør.
- Se på muligheten for å integrere nåværende sambandsløsninger om bord slik at et eventuelt brillesett kan sømløst flettes sammen med allerede etablerte systemer på bro. På denne måten kan man oppnå flere funksjoner med færre fysiske enheter. En større symbiose mellom brillesettet og broen.

7 Konklusjon

Denne oppgaven hadde som mål å ta for seg hvordan man kan designe, implementere og teste et augmented reality system til bruk under hurtigbåtnavigasjon. Gjennom prosjektet har vi laget et program med et grafisk brukergrensesnitt som kan vise dynamiske data i sanntid fra simulatoranlegget ved Sjøkrigsskolen. Brukergrensesnitt inkluderer digitale peilingsmarkører som kan brukes til stevning, samt muligheten til å kontrollere og interagere med systemet ved hjelp av stemmefunksjoner og et digitalt brukergrensesnitt.

Figur 81 HoloLens gir overskudd, også ved peilesøylen.

Resultatene fra testene viser at systemet oppleves som lett å forstå, og kan integreres i etablerte navigasjonsrutiner. Ifølge testpersonene oppleves informasjonen plassert på en gjennomtenkt og hensiktsmessig måte. Til tross for dette har teknologien enda utfordringer, der resultatene viser blant annet at testpersonene opplever brillesettet sin ergonomi som problematisk. Herunder er justeringsmuligheter og synsvinkel gjentagende tilbakemeldinger som begrenser utbytte til systemet. Til tross for dette mener vi at teknologien har et stort potensialt i fremtiden, og dersom utfordringene som gjelder ergonomi blir løst i samspill med en økt ytelse og gode programutviklere, vil teknologien kunne bli et viktig verktøy for fremtidens navigatører.

8 Bibliografi

Assev, Sigurd M. og Mikalsen, Arne B. (2012). *Drift av lokalnettverk Design og sikkerhet*. 7.utgave. Trondheim: Akademika forlag.

Azuma, Ronald T. (1997). A survey of Augmented reality, *Hughes Research Laboratories.* Hentet 15.06.2019 fra https://www.cs.unc.edu/~azuma/ARpresence.pdf

Aniwaa. (2019). HoloLens 2, *Aniwaa.* Hentet 22.08.2019 fra https://www.aniwaa.com/product/vr-ar/microsoft-hololens-2/

Bothner-By, Halvor (2018). Datakommunikasjon, *SNL*. Hentet 01.12.2019 fra https://snl.no/datakommunikasjon

Dvergsdal, Henrik og Ulseth, Trond. (2018). Internettprotokoll, *SNL*. Hentet 01.12.2019 fra https://snl.no/Internettprotokoll

Egkarchos, Konstantinos. (2019). Hololens spatioal mapping – a developer's guide, *Lightbuss.* Hentet 29.11.2019 fra https://lightbuzz.com/hololens-spatial-mapping/

Grabowski, Martha. (2014). Research on Wearable, Immersive Augmented Reality (WIAR) Adoption in Maritime Navigation, *Journal of Navigation.* Hentet 01.12.2019 fra https://www.cambridge.org/core/journals/journal-of-navigation/article/research-on-wearable-immersive-augmented-reality-wiar-adoption-in-maritime-navigation/EE0F6FE5E521343D35CAF6ACF7E34F38

Hansen, Kjell Toft og Mallaug, Tore. (2008). *Databaser.* 2. utgave. Oslo: Gyldendal akademiske.

Kjerstad, Norvald. (2019). *Elektroniske og akustiske navigasjonssystemer for maritime studier*. 6. utgave. Bergen: fagbokforlaget.

Pollefeys, Marc.(2018). Microsoft HoloLens facilitates computer vision research by providing access to raw image sensor streams with Research Mode, *Microsoft.* Hentet 29.11.2019 fra https://www.microsoft.com/en-us/research/blog/microsoft-hololens-facilitates-computer-vision-research-by-providing-access-to-raw-image-sensor-streams-with-research-mode/

Myren, K. Sverre. (2017). Baud, *SNL*. Hentet 01.12.2019 fra https://snl.no/baud

Microsoft. (2019). What is the Mixed Reality Toolkit, *Github.* Hentet 12.11.2019 fra https://microsoft.github.io/MixedRealityToolkit-Unity/README.html

Microsoft. (2019). XML for nybegynnere, *Support.Office.*

Hentet 12.08.2019 fra https://support.office.com/nb-no/article/xml-for-nybegynnere-a87d234d-4c2e-4409-9cbc-45e4eb857d44

NMEA. (2019). NMEA 0183 Interface Standard, *NMEA.*

Hentet 01.12.2019 fra https://www.nmea.org/content/STANDARDS/NMEA_0183_Standard

Nielsen, Jakob. (1994). How to Conduct a Heuristic Evaluation

Hentet 01.12.2019 fra https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/

RaspberryPi. (2019). What is a Raspberry Pi?, *Raspberry Pi.*

Hentet 02.06.2019 fra https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/

Schmalstieg, Dieter og Hölleren, Tobias. (2016). *Augmented reality: principles and practice*,

Boston: Addison-Wesley.

Sadabadi , A. Tizkar og Tabatabaei, Naser M. (2009). Rapid prototyping for software projects

with user interfaces, *State Engineering University of Armenia.*

Hentet 01.12.2019 fra https://www.researchgate.net/publication/256839827_Rapid_Prototyping_for_Software_Projects_with_User_Interface

Straw, John. (2015). Disrupted Television – Virtual Reality and Holographics Converge,

Disruption. Hentet 01.12.2019 fra https://disruptionhub.com/disrupted-television-virtual-reality-and-holographics-converge/ Med kryssreferanse https://www.brainyquote.com/quotes/tim_sweeney

TheAVLtube. (11.07.2017). *Using Vuforia For HoloLens Static Room*

Calibration In Unity (Tutorial) [Videoklipp]. Hentet fra https://www.youtube.com/watch?v=W7z2sggDGoo

Tronico. (2019). The NMEA 0183 Protocol, *Tronfico.*

Hentet 01.12.2019 fra https://www.tronico.fi/OH6NT//docs/NMEA0183.pdf

Urke, Eirik Helland. (2018). *VR og AR – en norsk introduksjon til virtual og augmented*

reality. Oslo: Cappelen Damm.

Wood, Robert b., Peter J. Howells. (2001). *The Avionics Handbook.* Williamsburg: CRC

Press.

9 Appendiks

Her vil leseren kunne finne en del mer tekniske detaljer og beskrivelser relevant for prosjektet. Følgende tekst vil kreve større teknisk forståelse og leses etter behov eller ønske. Vedleggene omfatter ulike skjemaer brukt under testing, resultater fra disse, kode, programvareversjoner, variabelliste, beskrivelse av Node-RED brukergrensesnitt, detaljert forklaring av implementering og konfigurasjon av programvare.

Vedlegg A: Samtykkeskjema

Vil du delta i prototypetesting i forbindelse med bacheloroppgave?

Dette er en forespørsel til deg om å delta i vårt prosjekt hvor formålet er å vurdere prototypen vi har utviklet i forbindelse med bacheloroppgaven. I dette skrivet gir vi deg informasjon om målene for testen og hva deltakelse vil innebære for deg.

Formål

Formålet med denne testen er å funksjonsteste prototypen vi har utviklet med relevante brukere. Vi ønsker å få brukerens erfaringer med tanke på komfort, hurtighet, presisjon, funksjonalitet og brukervennlighet.

Hvem er ansvarlig for forskningsprosjektet?

Forsvarets høyskole avdeling Sjøkrigsskolen er ansvarlig for prosjektet.

Hvorfor får du spørsmål om å delta?

Du er valgt ut til å delta på bakgrunn av din formelle kompetanse innenfor maritim navigasjon.

Hva innebærer det for deg å delta?

Hvis du velger å delta i prosjektet, innebærer det at du fyller ut et spørreskjema. Det vil ta deg ca. 10 minutter. Spørreskjemaet inneholder spørsmål om komfort, hurtighet, presisjon, funksjonalitet og brukervennlighet av prototypen. Spørreundersøkelsen besvares på papirform, men svarene vil bli lagret elektronisk.

Hvis du velger å delta i prosjektet, innebærer det at du vil bli observert av alle medlemmene av bacheloroppgaven under funksjonstesting av prototypen. Dette vil kunne ta ca. 30-60 minutter.

Dersom du godtar, vil du bli avbildet under testing av prototypen. Bildene vil bli anonymisert for å unngå personidentifisering.

Det er frivillig å delta

Det er frivillig å delta i prosjektet. Hvis du velger å delta, kan du ikke etter gjennomført besvarelse trekke tilbake opplysninger på et senere tidspunkt. Alle opplysninger om deg vil bli anonymisert. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta i testingen.

Ditt personvern – hvordan vi oppbevarer og bruker dine opplysninger

Vi vil bare bruke opplysningene om deg til formålene vi har fortalt om i dette skrivet. Vi behandler opplysningene konfidensielt og i samsvar med personvernregelverket. Det vil bare være medlemmer av bachelorgruppen som vil ha tilgang til dine utfylte opplysninger. Deltagerne i denne prototypetesten vil ikke kunne gjenkjennes basert på utfylling av spørreskjema, da navn, alder og kjønn ikke vil være relevant informasjon.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?

Prosjektet skal etter planen avsluttes 4.desember.

Dine rettigheter

Så lenge du kan identifiseres i datamaterialet, har du rett til:

- innsyn i hvilke personopplysninger som er registrert om deg,
- å få rettet personopplysninger om deg,
- få slettet personopplysninger om deg,
- få utlevert en kopi av dine personopplysninger (dataportabilitet), og
- å sende klage til personvernombudet eller Datatilsynet om behandlingen av dine personopplysninger.

Hvor kan jeg finne ut mer dine rettigheter og personvern?

Hvis du har spørsmål til studien, eller ønsker å benytte deg av dine rettigheter, ta kontakt med:

NSD – Norsk senter for forskningsdata AS, på epost (personverntjenester@nsd.no) eller telefon: 55 58 21 17.

Med vennlig hilsen

Christer Algrøy, Markus Øvstedal og Julian T. Venstad

Jeg har mottatt og forstått informasjon om prototypetesting i forbindelse med bacheloroppgave, og har fått anledning til å stille spørsmål. Jeg samtykker til:

- å delta i spørreundersøkelse etter testing av prototype
- å bli avbildet under testingen
- at jeg blir observert av medlemmer av prosjektgruppen under testing

Jeg samtykker til at mine opplysninger behandles frem til prosjektet er avsluttet, ca. 4 desember.

(Fullt navn i blokkbokstaver)

(Signert av prosjektdeltaker, dato)

Vedlegg B: Heuristisk evaluering av brukergrensesnitt
Heuristisk evaluering av brukergrensesnitt

Dato:

Sted: **Simulator:**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på riktig måte. Eksempelvis er det ikke hensiktsmessig å skrive «*jeg likte ikke dette*», da dette er lite informativt for utviklingsteamet.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

Stikkord: Oppløsning, field of view, refreshrate, treghet

Kommentar

Vedlegg C1-C7: Utfylte Spørreskjemaer

Vedlagt ligger utfylte spørreskjemaer som er digitalisert. Spørreskjemaene er anonymiserte. Verdt å merke seg at skjema 1 og 2 er gjennomført av samme testperson under henholdsvis del I og del II. Skjema C1 og C3 er fra del I, og resten fra del II. Resultatene legges ved for videre bruk, samt for å gjøre våre erfaringer og funn sporbare. Samtidig ønsker vi å påpeke at en god andel av tilbakemeldinger foruten om disse skjemaene ikke er blitt dokumentert like godt.

Vedlegg C1:
Heuristisk evaluering av brukergrensesnitt

Testperson: 1 **Dato: 10.10.19**

Sted: SKSK **Simulator: NAVKOMP bro E**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

Forventer at utstyret vil være tungt og uvant. Forventer at informasjonen vil blokkere mye av synet.

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
		X

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på en informativ måte slik at utviklingsteamet forstår evalueringen.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

- Dybde på tekst -> Trolig for nært
- Intensitet: For sterk:
- Informasjon må opp og vekk fra der hvor vi trenger å se ut.
- Tekst må bli mer gjennomsiktig
- Informasjon: Få inn det vesentlige slik vi pratet om.

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

- Vil kreve noe opplæring for å ta i bruk
- Har potensialet -> men hjelper ikke før alt er på plass
- Posisjon i lat/long er unødvendig. Planned speed er unødvendig
- Current heading: Unødvendig
- Må få inn current course
- Må få inn next course og next dist.

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Enkelt. Passformen var litt dårlig for min hodeform. Voicefunksjonen virket tungvint. En fysisk knapp vil være like enkelt og ev mindre «sårbart.»

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

Bra. IÅM.

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

Stikkord: Oppløsning, field of view, refreshrate, treghet

God oppløsning. Føltes ellers ut som det var hurtig refreshrate. Treghet med voice command funksjonen.

Kommentar

Blank

Vedlegg C2:
Heuristisk evaluering av brukergrensesnitt

Testperson: 1 **Dato: 15.11.19**

Sted: SKSK **Simulator: E**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

Blank

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
		X

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på en informativ måte slik at utviklingsteamet forstår evalueringen.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

Ok farge og lysstyrke. God dybde. God plassering. Færre bugs enn tidligere.

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

Delay i noe data som blir presentert. Føler at man kan stole mindre på systemet da. Spesielt i tørn hvor kursen endres fort.

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Enklere nå enn første gang. Må fortsatt holde hode ganske høyt opp.

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

NIL.

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

Stikkord: Oppløsning, field of view, refreshrate, treghet

Fortsatt noe tregt.

Kommentar

BZ.

Vedlegg C3:
Heuristisk evaluering av brukergrensesnitt

Testperson: 2 **Dato: 10.10.19**

Sted: SKSK **Simulator: Navkomp Bro E**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

Forventer at informasjonen på brillene vil bli forstyrrende til å begynne med. Forventer at brillene bidrar til overskudd på sikt.

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
		X

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
		X

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på en informativ måte slik at utviklingsteamet forstår evalueringen.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

Presenterte relevant informasjon. Løfte informasjon over horisont. Rød tekst. Bruke forkortelse IHT SNP-500

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

Fysiske knapper vil alltid være enklere å betjene

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Komfort kunne vært bedre. Utover det var brukervennlighet bra.

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

Veldig bra, opplever systemet som dynamisk

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

Stikkord: Oppløsning, field of view, refreshrate, treghet

Treg refreshrate på bytte av waypoint Fin størrelse på teksten

Kommentar

Vedlegg C4:
Heuristisk evaluering av brukergrensesnitt

Testperson: 3 **Dato: 15.11.19**

Sted: SKSK **Simulator: Echo**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

Mindre bruk av titting i kart. Bedre tørn

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på en informativ måte slik at utviklingsteamet forstår evalueringen.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

Synligheten er veldig god. Bra font.

Noe ustabilt, ser for meg utfordringen i høye hastigheter og røff sjø.

Trenger noe finpuss på hvilken info som vises.

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

Savner utseilt distanse på inneværende leg.

Litt «blurry» å se gjennom

Forstyrrende med den britiske damen som prøver seg som assistent. Kunne eventuelt blitt erstattet av et pip på 2min til tørn og 30s.

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Enkelt og forstå på kort tid.

Kan bli en utfordring og ha på i 6 timer, men for forbigående utfordrende område tror jeg systemet kan bli verdifullt.

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

Drømmen må være at systemet er linket til navigatørens stoppeklokke og logg. Den våte drømmen er å koble den til gyroen slik at RPD'en på korvett. Da kan man få opp info om AIS og ARPA targets og peiling til stevn osv.

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

Stikkord: Oppløsning, field of view, refreshrate, treghet

Tar litt for lang tid å bytte mellom WP's. Burde ha hardbuttons til dette.

Kommentar

Veldig imponert over det dere har fått til så langt. Har troa på at dette kan bli et vitig hjelpemiddel i fremtiden. Bistår gjerne med støtte hvis dere trenger mer.

Vedlegg C5:
Heuristisk evaluering av brukergrensesnitt

Testperson: 4 **Dato: 15.11.19**

Sted: SKSK **Simulator: E**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

-Lettere tilgang på navigasjonsviktig informasjon. Dermed mer tid til å se ut og ha SA.

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
		X

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på en informativ måte slik at utviklingsteamet forstår evalueringen.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

-Informasjonen som ble vist var nyttig når den stemte. Problemet var derimot at infoen var ofte feil. Infoen byttet også side fra nøyre til venstre veldig tidlig og uten forvarsel.

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

- Blir de feil som er nevnt over korrigert kan systemet bli veldig bra. I tillegg var det «Blury»(utydelig) å se gjennom brillene, dette gjørende objektidentifisering meget vanskelig.

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Enkelt. Det krever nok en del tilvenning, men systemet er enkelt uansett.

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

Eg har ikke blitt orientert om det er mulighet for å endre noe på systemet. Jeg vil derimot foretrekke om det hadde vært mulig å personliggjøre.

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

Stikkord: Oppløsning, field of view, refreshrate, treghet

Systemet trodde jeg turnet før jeg faktisk gjorde det. Distansen gjenværende på legget så ut til å stemme og var tids aktuell.

Kommentar

-Potensielt meget nyttig og spennende -En del software optimalisering kreves -En bør løse problemet med at det er utydelig å se gjennom brilleglasset

Vedlegg C6:
Heuristisk evaluering av brukergrensesnitt

Testperson: 5 **Dato:15.11**

Sted: SKSK Navkomp **Simulator: Echo**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

Jeg ønsker SOG, kurs, planlagt kurs, 30s tørninfo på neste, log og tid. Kanskje det blir for mye info

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på en informativ måte slik at utviklingsteamet forstår evalueringen.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

Farge: OK. Størrelse: Kanskje prioritert kurs og dist til tørn og mindre skrift i forhold på annen info.

Type info var bra. Eg tror det kan bli forvirrende med mer info.

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

Veldig bra

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Kunne vært mer brillevennlig god passform. At den er intuitiv er ikke krav fra meg, opplæring kreves jo uansett

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

Bra å kunne justere høyde. Kritisk farlig om teksten hindrer meg i å se lykter.

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

Stikkord: Oppløsning, field of view, refreshrate, treghet

Bra oppløsning. Men eg hadde foretrukket at infoen alltid var i synsfeltet og ikkje låst til over baugen.

Kommentar

BZ.

Vedlegg C7:
Heuristisk evaluering av brukergrensesnitt

Testperson: 6 **Dato: 15.11.19**

Sted: SKSK **Simulator: E**

FØR BRUK

Beskriv kort dine forventninger til bruk av AR i navigasjon:

At informasjonen blir lettere tilgjengelig enn tidligere.

Hvor teknisk anlagt er du?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
X		

Hvor god anser du deg selv i praktisk navigasjon i forhold til personer med lik erfaring?

Under gjennomsnittlig	Gjennomsnittlig	Over gjennomsnittlig
	X	

ETTER BRUK

Du skal nå evaluere brukergrensesnittet basert på det som kalles de heuristiske faktorene. Det er i denne sammenheng viktig at svarene blir gitt på en informativ måte slik at utviklingsteamet forstår evalueringen.

Spørsmål 1

Beskriv kort dine erfaringer med tanke på informasjonen som blir vist i brillesettet.

Stikkord: farge, størrelse, ulike typer informasjon, skrifttype.

Informasjonen var oversiktlig satt opp. Fargene og skriftstrl. tok ikke vekk oppmerksomheten fra selve seilasen. Lett å forholde seg til

Spørsmål 2

Hvordan føler du systemet fungerer med tanke på funksjonalitet?

Stikkord: standarder, enheter, framvisning av informasjon, navigasjonstekniske detaljer

Fremvisningen av informasjon ga overskudd, da en slipper å se ned i kartet eller «huske på» detaljene på neste tørn. Gradpilene hadde for min del vært mer praktisk om hadde tatt utgangspunkt, at en står bak peilesøylen, slik at en ikke må flytte seg for å benytte de

Spørsmål 3

Hvor lett opplevde du det var å ta systemet i bruk?

Stikkord: oppkobling, justering, endring, innstillinger, sømløshet, passform

Lett å anvende og koble til.

Spørsmål 4

Hvordan føler du fleksibiliteten til systemet?

Stikkord: Endringer, valgmuligheter, preferanser

God fleksibilitet

Spørsmål 5

Hvordan opplevde du systemet med tanke på hurtighet og presisjon?

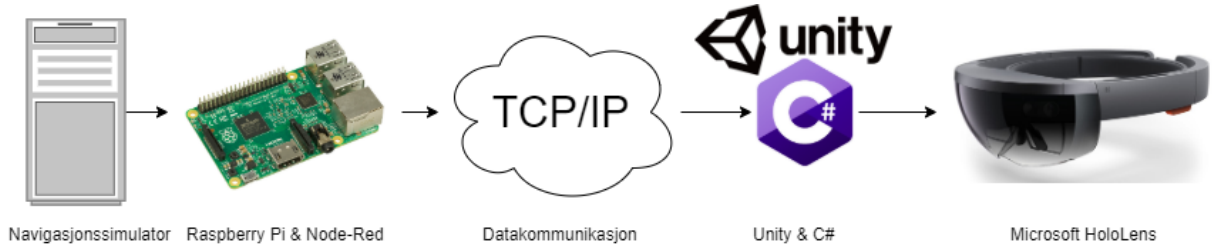
Stikkord: Oppløsning, field of view, refreshrate, treghet

Noe delay i course over ground, ellers presist på resten. God oppløsning. Selve brillen sløret til blikket noe slik at det ble vanskelig å oppdage blinker langt unna.

Kommentar

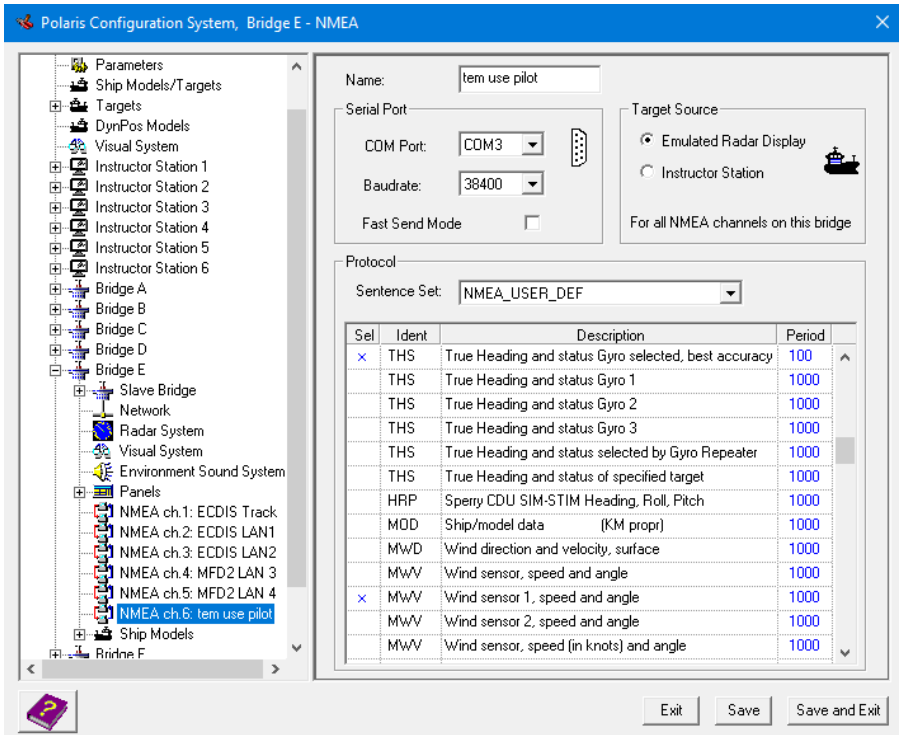
Vedlegg D: Detaljert forklaring av implementering

1. Uthenting av data fra simulatoranlegget

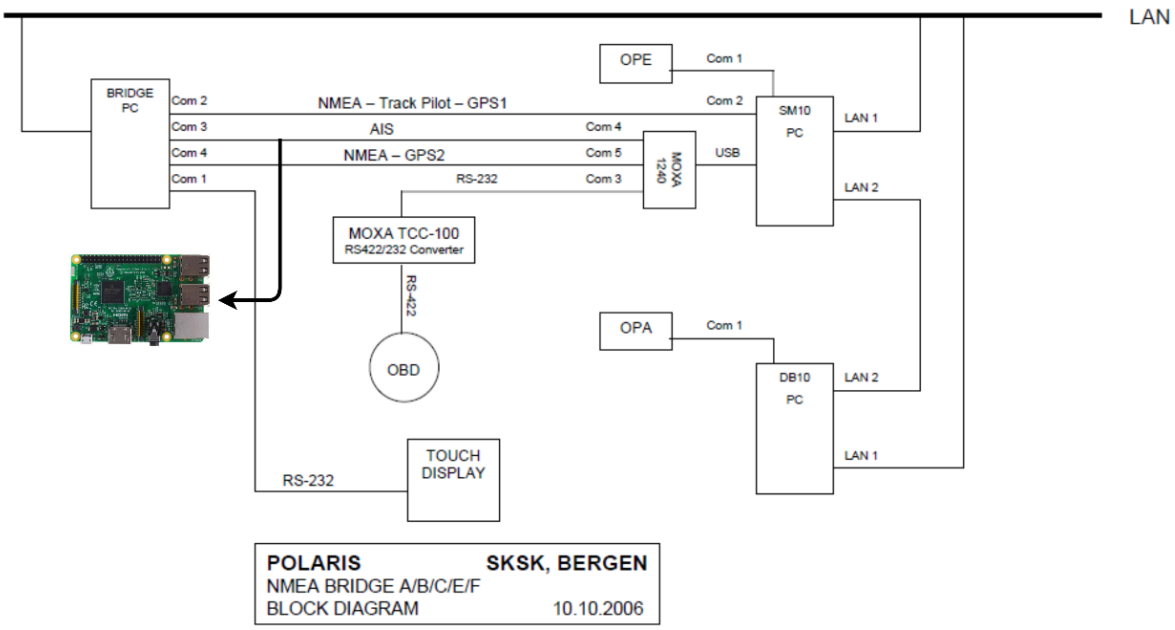


Figur D.1 Dataens vei fra simulator til brillesettet.

Dette delkapittelet vil ta for seg hvordan vi har hentet ut data fra simulatoranlegget på Sjøkrigsskolen. For å kunne vise navigasjonsinformasjon i AR-brillene, kreves det en form for datauthenting i bunn. For å gjøre dette koblet vi oss på en RS-232 utgang på COM port 3. I utgangspunktet blir denne brukt til å sende AIS datastrenger til et SeaCross-system. Ved hjelp fra Kongsberg Maritime fikk vi rekonfigureret utgangen slik at den sender relevant informasjon i henhold til vår kravspesifikasjon, med en høyre frekvens enn det var i utgangspunktet. Eksempelvis som vist på *Figur D.2****Feil! Fant ikke referansekilden.***. *Figur D.3* og *Figur D.6* viser hvordan vi koblet oss til og henter ut data ved hjelp av seriell kommunikasjon. *Figur D.3* er et flyt-diagram over simulatoranlegget. Sentral informasjon å bemerke seg er at vi henter ut data fra COM 3 fra *Bridge PC* som er en seriell port, for å lese av denne dataen bruker vi en RS232 adapter.

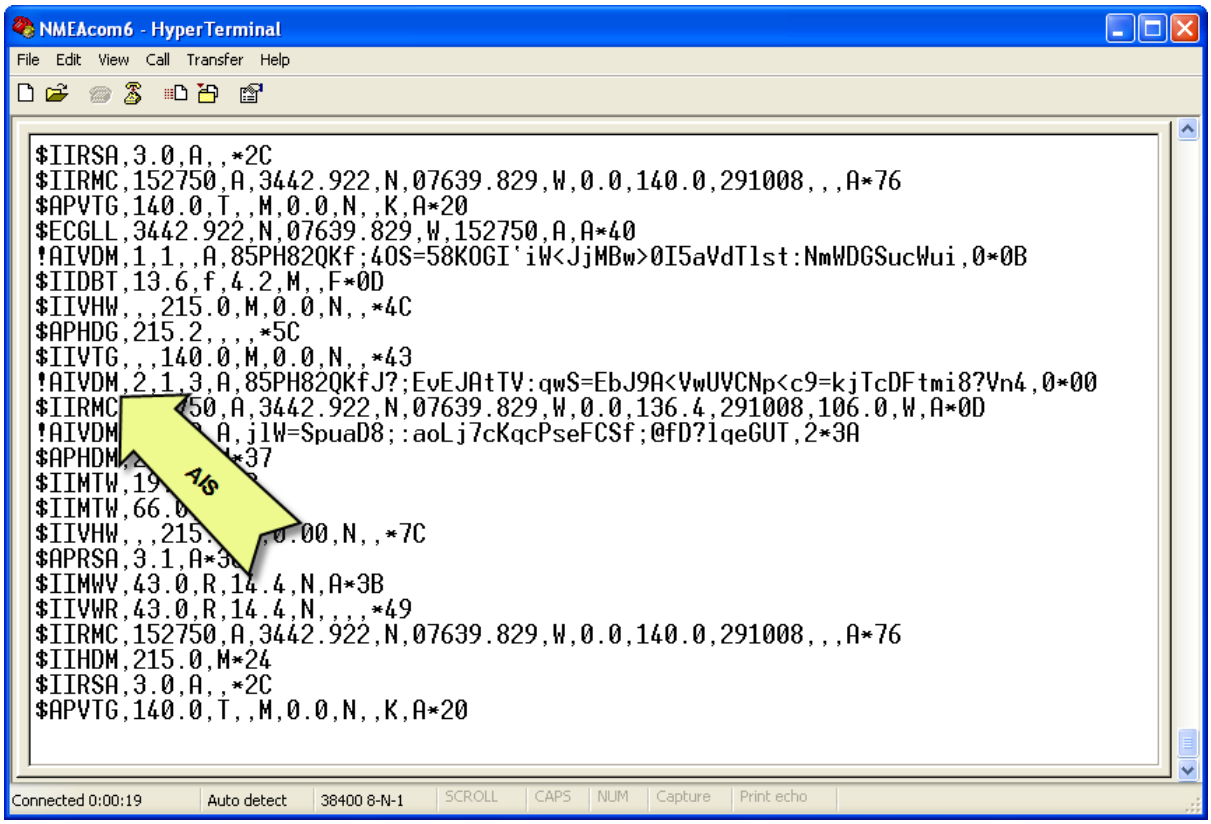


Figur D.2 Polaris konfigurasjon



Figur D.3 Raspberry Pi koblet på COM 3.

Figur D.4 Datauthenting i simulatoranlegget med RS232 adapter og HyperTerminal.



Figur D.5 HyperTerminal med AIS-strenger.

Figur D.6 RS232 adapter.

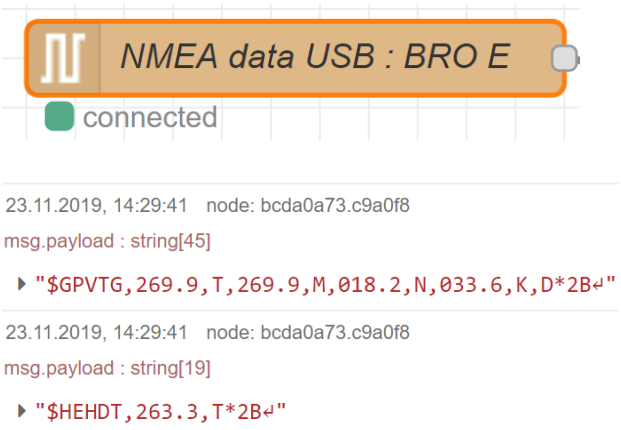
I denne sammenheng er det også viktig å legge til at USB-porten som tar imot data er formatert med riktige innstillinger. Disse innstillingene fant vi ved å først bruke *HyperTerminal* og en bærbar PC som vist på *Figur D.4.* Innstillingene for å få ut dataene riktig ser man i *Figur D.7.* Som du ser overstemmer ikke baudrate med den vi definerte som NMEA 0183 standard, som var 4800. Dette kommer av at simulatoren sitt system har en høyere baudrate, dermed benytter vi oss av følgende konfigurasjon:

Baudrate	38400
Databits	8
Paritet	Ingen
Stoppbit	1

Figur D.7 Innstillinger for datauthenting.

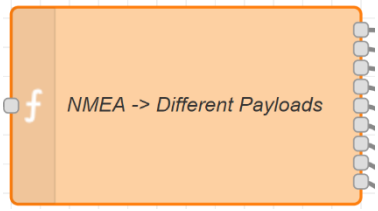
2. **Databehandling i Node Red**

Etter at dataen har blitt samlet inn fra simulatoranlegget kan den bli behandlet i Node Red. Den første noden som kan nevnes er seriell-port-noden, som tar imot dataen fra simulatoren.



Figur D.8 Serial-port node og output fra noden.

Deretter blir dataen sendt videre til noden som heter *"NMEA -> Different Payloads".* Noden skiller mellom de ulike NMEA-strengene, før den henter ut den relevante informasjonen ved hjelp av slice-funksjonen. Funksjonen ekstraherer de relevante karakterene fra tekststrengen og setter de som en ny variabel.

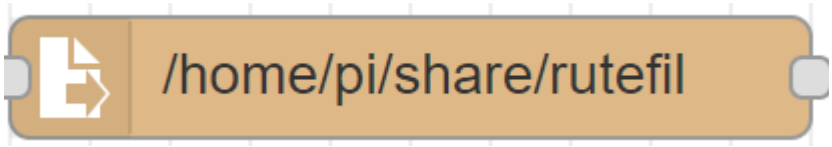



```
//If right NMEA string
if (msg.payload.includes('GPZDA,'))
{
//Split string into right size. The characters from 7-16 represents the time value
let TIME = msg.payload.slice(7,16);
//The parseFloat() function parses a string and returns a floating point number.
var numberValue = parseFloat(TIME);
context.set(TIME);

//The payload will be set to the value of numberValue (Time). Topic will just be time.
msg1= {payload: numberValue, topic: "Time"};
//Will set the global variabel time to the value of numbervalue
global.set('Time',numberValue);

}
```

Figur D.9 Kodeutsnitt av noden som skiller strengene.

3. Behandling av informasjon fra ruteplanlegging

I tillegg til å hente ut NMEA strenger fra simulatoren bruker vi også ruteplanleggingsfiler. Siden *Kongsberg Digital* ikke lot oss hente ut data fra ECDIS systemet, må filene legges inn på en Raspberry Pi før seilasen starter. Filen leses av med en fil-node, der man må spesifisere filbanen til filen.



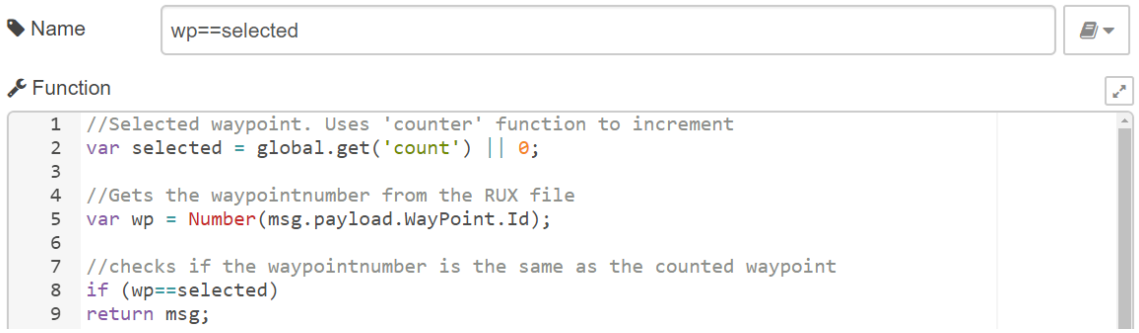
Figur D.10 Valg av filbane.

Siden ECDIS bruker rux-format på ruteplanleggingsfilene, må vi ha en node som gjør at vi kan lese av dataen. Det tar den neste noden seg av. Det er en XML-konverter-node. Noden deler også informasjonen i ulike payloader, slik at det blir lettere bruke dataen.

Figur D.11 Avlesning av XML-formatert data.

.

Etter XML-konverter noden, møter man en funksjonsnode. Denne noden har som oppgave å kun slippe igjennom veipunktet som er valgt i Node-RED. Noden sammenligner veipunktnummeret fra filen med en variabel som angir valgt veipunkt. Dersom disse stemmer overens, vil veipunktet bli sendt videre.



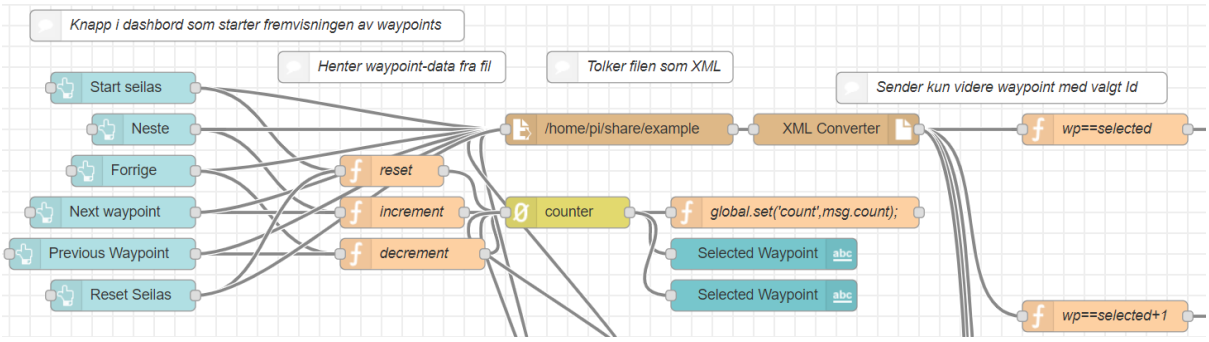
Figur D.12 Sender videre valgt veipunkt.

Det valgte veipunktet går videre inn i en ny funksjonsnode. Denne noden har som oppgave å hente ut verdier fra rux-filen, samt å sette disse til globale variabler for videre bruk i Node-RED.

Figur D.13 Lagrer verdier i globale variabler.

På bildet under kan man se hvordan flyten er bygget opp. De tre knappene man ser til venstre, er tilknyttet brukergrensesnittet. Trykker man start seilas, vil telleren nullstilles. Når man trykker på neste/forrige vil telleren telle opp eller ned, og det er slik man angir hvilket veipunkt

man ønsker. Ved hvert trykk vil veipunktfilen leses av og bli filtrert på måten som beskrevet over.



Figur D.14 Deler av flyten som velger veipunkt.

4. Avstand og tid til veipunkt

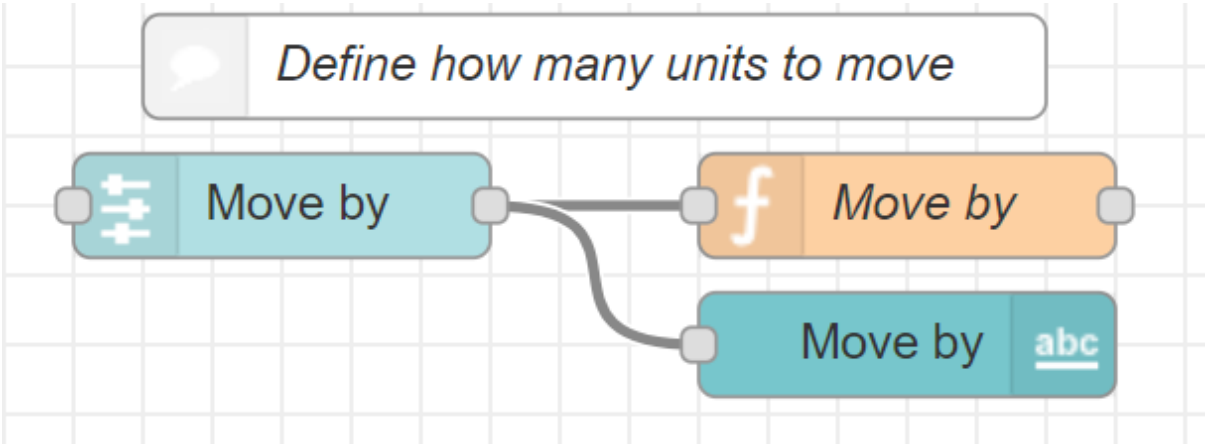
På grunn av manglende ECDIS-data måtte vi bruke trigonometri for å kunne regne ut enkelte verdier. Det første vi måtte regne ut var avstanden mellom veipunkt og fartøy. Til dette brukte vi deres respektive bredde og lengdegrader. For å regne ut dette brukte vi Haversine-formelen. Kodesnutten på *Figur D.15* viser hvordan dette blir løst i JavaScript.

```
1  //Function to convert value into radians
2  Number.prototype.toRad = function() {
3     return this * Math.PI / 180;
4  }
5
6
7  //Distance between waypoint and the vessel
8  //based on the Haversine Formula: https://www.movable-type.co.uk/scripts/latlong.html
9  var lat1 = parseFloat(global.get("Lat")); //Vessel Latitude
10 var lon1 = parseFloat(global.get("Lon")); //Vessel Longitude
11 var lat2 = parseFloat(global.get("Waypointlat")); // Waypoint Latitude
12 var lon2 = parseFloat(global.get("Waypointlon")); // Waypoint Longitude
13
14 var R = 6371; // Earth Radius in km
15 var x1 = lat2-lat1;
16 var dLat = x1.toRad();
17 var x2 = lon2-lon1;
18 var dLon = x2.toRad();
19 var a = Math.sin(dLat/2) * Math.sin(dLat/2) +
20                 Math.cos(lat1.toRad()) * Math.cos(lat2.toRad()) *
21                 Math.sin(dLon/2) * Math.sin(dLon/2);
22 var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
23 //The distance between the vessel and the waypoint
24 //divided by 1.852 to get it in nautical miles
25 var d = (R * c/1.852);
26
```

Figur D.15 Utregning av gjenværende avstand til veipunkt..

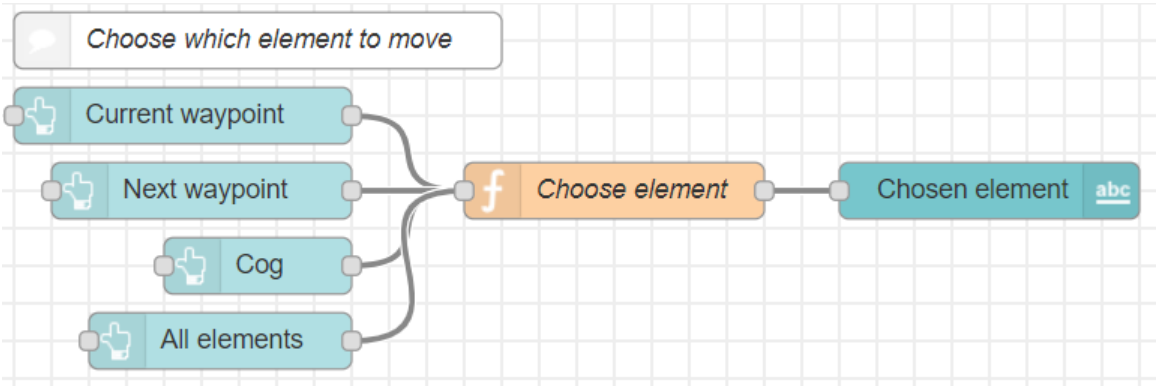
5. UI Editor

For å kunne endre brukergrensesnittet uten å måtte kompilere på nytt, lagde vi et panel som kan endre variabler som styrer brukergrensesnittet samtidig som brillene er i bruk. Vi har laget skyver kalt "*move-by*", som bestemmer hvor mange meter man vil flytte de ulike elementene i den virtuelle scenen. Move-by fuksjonsnoden endrer den globale variabelen "*Moveby*" til ønsket verdi, mens tekst-noden viser den satte verdien i dashbordet.



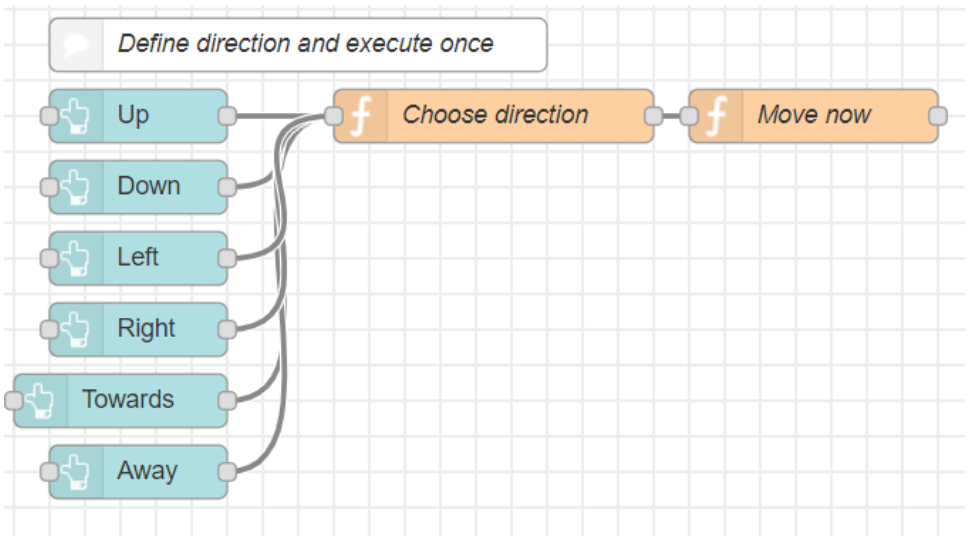
Figur D.16 Bestemmer lengden til neste flytt.

For å velge de ulike elementene har vi lagt inn knapper. Etter man har valgt et element, vil funksjonen "*Choose element*" sette den globale variabelen "*Element*" til valgt element. Det er også en "*Chosen element*" tekst-node som viser valgt element i dashbordet.



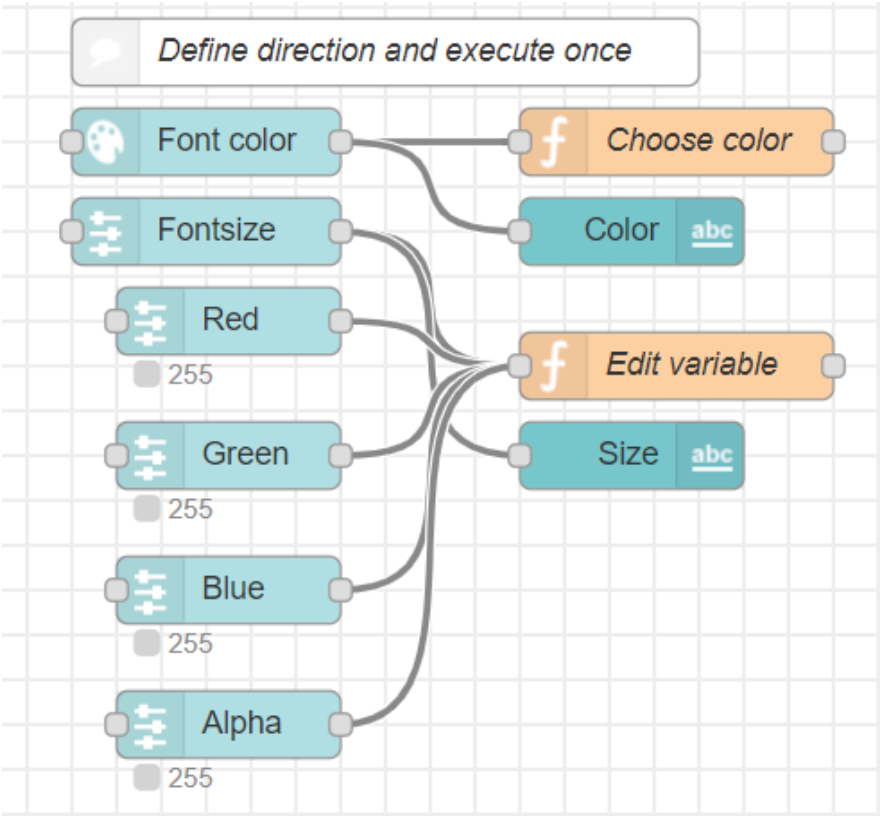
Figur D.17 Angir element som skal flyttes.

For å velge retning man skal flytte de ulike elementene, har vi seks ulike knapper. Opp, ned, venstre og høyre styrer bevegelse i planet som brukergrensesnittet befinner seg i, notasjonene er i forhold til frem på båten. Mot og vekk knappene styrer bevegelsen av planet mot og vekk fra personen som har på seg brillene. Oppsettet er likt som "*choose element*" nodene. dashbord-knapper brukes for å velge retning, en funksjonsnode endrer den globale variabelen "*Direction*". Når retning er valgt vil "*Movenow*" settes til True – og tekstfeltet flyttes. Dersom man ønsker å flytte elementene ytterligere, trykker man på retning igjen.



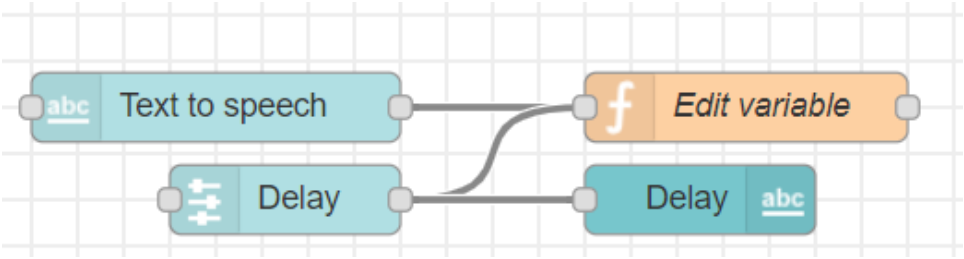
Figur D.18 Velger retning og gjennomfører flytt.

Rent kosmetisk, har vi også lagt inn en meny for endring av fontstørrelse og farge. I RGBA bestemmes en farge av fire komponenter gitt med heltall i spennet 0-255. Man kan enten velge farge basert på Node-RED sin egen fargevelger, eller endre på sammensetningen av rød, grønn og blå for å endre farge. Videre kan man også endre alpha-verdien som er gjennomsiktheten til teksten. Både Choose-color-funksjonen og Edit-variable-funksjonen endrer de globale variablene etter hvilken input den blir gitt.



Figur D.19 Angir farge på tekst i brillene.

Videre har vi også lagt inn et tekstfelt som vil bli lest opp av en tekst til tale funksjon i brillene. I tillegg til tekstinput kan man også ved hjelp av en skyver endre forsinkelsen på meldingen. Det vil si hvor mange sekunder systemet skal vente etter varselet om innkommende melding før meldingen blir lest opp.

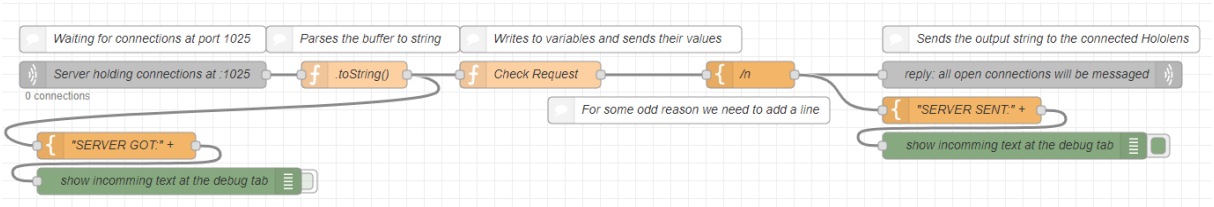


Figur D.20 Tekst som skal leses av test-til-tale.

6. TCP kommunikasjon i Node-RED – Server

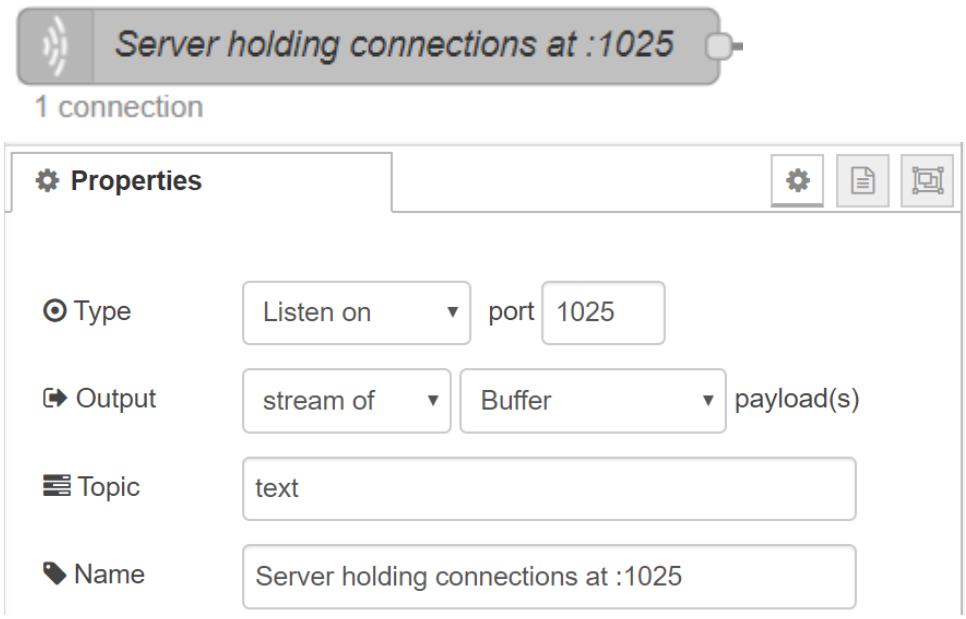
Figur D.21 Node-RED (server), HoloLens (klient).

Vi valgte å bruke gjøre mest mulig databehandling på Raspberry Pi enheten. Vi anså det som hensiktsmessig at kun det nødvendige ble gjort i selve brillene, da dette ville påvirke ytelsen – noe vi senere fikk oppleve.



Figur D.22 TCP-flyten.

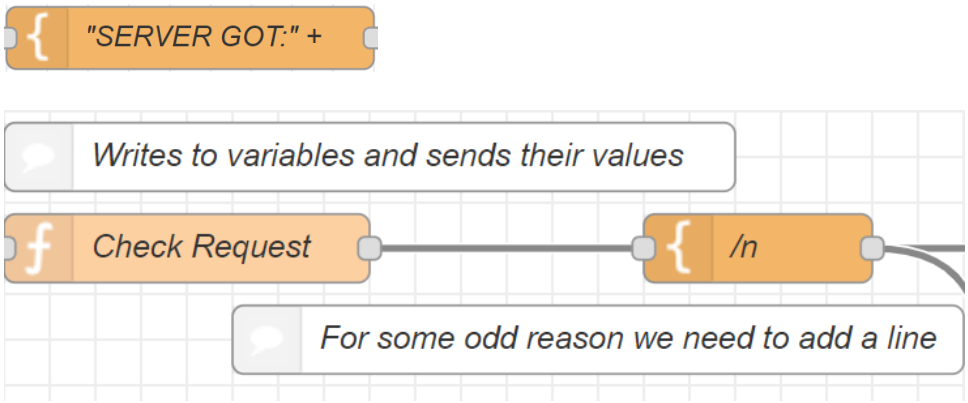
Bildet ovenfor viser hvordan TCP-flyten i Node-RED er satt sammen. Vi vil gå gjennom oppsettet fra venstre til høyre Det første vi ser er en TCP-in-node.



Figur D.23 Konfigurasjon av TCP-in-noden..

Ut ifra innstillingene kan vi se at den lytter på port 1025, samt at den sender ut en "*stream of buffers*". Kort fortalt er dette en delmengde av rå data som mellomlagres i hurtigminne (RAM), uten noen definert datatype. For å kunne lese bufferdataene bruker vi JavaScript funksjonen toString(), for å konvertere dataen til datatypen streng.

Ut ifra toString()-funksjonsnoden går det to linjer. Den ene går til en "*template-node*", som skriver "SERVER GOT" + tekststrengen den mottar. Dette blir brukt til feilsøking, for å sjekke hvilke meldinger som faktisk mottas av serveren.

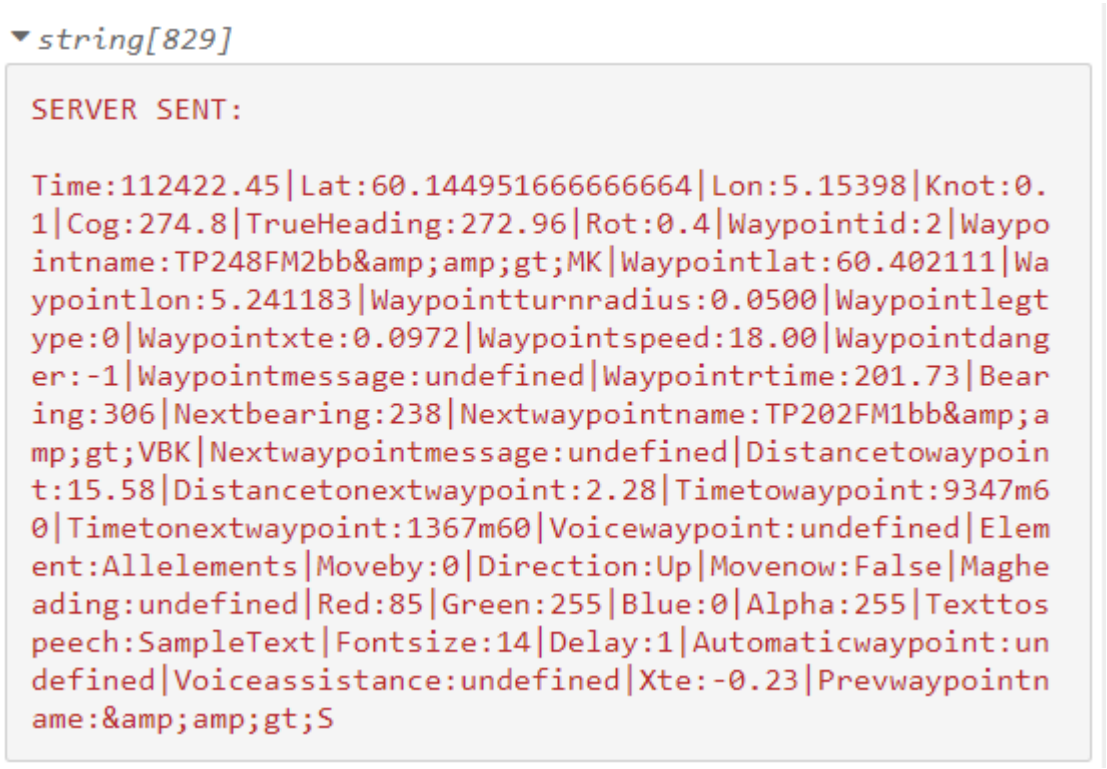


Figur D.24 Tar imot forespørsel fra klient.

```
//Available variables
var keyword=[  "Time", "Lat", "Lon", "Knot", "Cog", "TrueHeading", "Rot", "Waypointid", "Waypointname", "Waypointlat", "Waypointlon",
"Waypointturnradius", "Waypointlegtype", "Waypointxte", "Waypointspeed", "Waypointdanger", "Waypointmessage", "Waypointrtime", "Bearing",
"Nextbearing", "Nextwaypointname", "Nextwaypointmessage", "Distancetowaypoint", "Distancetonextwaypoint", "Timetowaypoint", "Timetonextwaypoint", "Voicewaypoint",
"Element", "Moveby", "Direction", "Movenow", "Magheading", "Red", "Green", "Blue", "Alpha", "Texttospeech", "Fontsize", "Delay", "Automaticwaypoint"];

//Writes data to variables with given value after ':', sent from the Hololens
if(msg.payload.includes(":")){
var splitstring = msg.payload.split(':');

for (i = 0; i < keyword.length; i++) {
if (splitstring[0]==keyword[i]){
    global.set(splitstring[0],splitstring[1]); //Example: "Alarm:True" --> Sets the global variable Alarm to True
    }

}
}

//Formats the output string with seperators as '|' and ':'
msg.payload="";
for (i = 0; i < keyword.length; i++) {
     if(i===0){
    msg.payload= msg.payload + keyword[i]+":"+global.get(keyword[i]);
      i++;

     }
    msg.payload= msg.payload + "|"+ keyword[i]+":"+global.get(keyword[i]);
}
return msg;
```

Figur D.25 Behandler forespørsel

Deretter møter man en funksjons-node og en template-node. "*Check Request*" noden har som oppgave å sjekke meldingen fra klienten. Dersom tekststrengen fra klienten inneholder et kolon ("Alarm:True"), vil tekststrengen bli splittet på kolonet (Alarm, True). Etter dette vil den aktuelle variabelen før kolon bli endret, med oppdatert verdi etter kolon. Deretter vil alle de gitte variablene bli samlet i en streng som vist på *Figur D.16*. Dersom meldingen fra klienten ikke inneholder et kolon, vil serveren ikke endre noen variabler, og forberede et svar på samme måte.

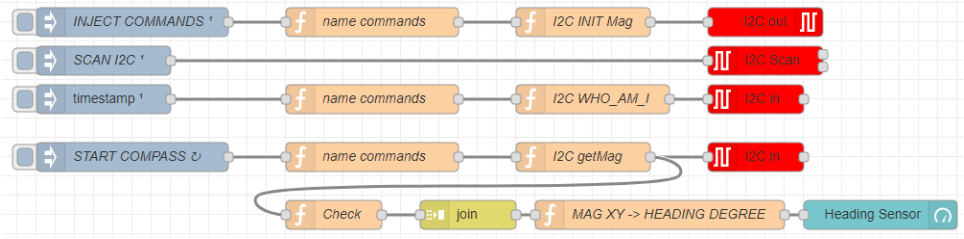


Figur D.26 Melding sendt fra server.

Til slutt blir den nye tekststrengen, som er ble laget i forrige node, sendt tilbake til klienten. Det vil også bli skrevet en melding i konsollviduet som sier "*Server sent*:", etterfulgt av tekststrengen som TCP-noden sender.

Implementering av HoloLens-retning

Det ble brukt en 6-akset-akselerometer og kompass fra Grove, basert på LSM303D sensormodulen, sammen med Node-RED for å få kompassretningen som brillene peker.



Figur D.27 Flyten brukt for å hente ut kompassbrikkens retning..

Figur D.28 Oppstart av kompassbrikke.

Den nest nederste linjen består av en inject-node som igangsetter funksjons-noden "*name commands*".

Name: name commands

Function

```
Object.prototype.merge = function(obj2) {
    for (var attrname in obj2) {
        this[attrname] = obj2[attrname];
    }
    //Returning this is optional and certainly up to your implementation.
    //It allows for nice method chaining.
    return this;
};

var topics = ["TEMP_OUT_L", "TEMP_OUT_H", "STATUS_REG_M", "OUT_X_L_M",
"OUT_X_H_M", "OUT_Y_L_M", "OUT_Y_H_M", "OUT_Z_L_M", "OUT_Z_H_M", "WHO_AM_I",
"INT_CTRL_M", "INT_SRC_M", "INT_THS_L_M", "INT_THS_H_M", "OFFSET_X_L_M",
"OFFSET_X_H_M", "OFFSET_Y_L_M", "OFFSET_Y_H_M", "OFFSET_Z_L_M", "OFFSET_Z_H_M",
"REFERENCE_X", "REFERENCE_Y", "REFERENCE_Z", "CTRL_REG0", "CTRL_REG1", "CTRL_REG2",
"CTRL_REG3", "CTRL_REG4", "CTRL_REG5", "CTRL_REG6", "CTRL_REG7", "STATUS_REG_A",
"OUT_X_L_A", "OUT_X_H_A", "OUT_Y_L_A", "OUT_Y_H_A", "OUT_Z_L_A", "OUT_Z_H_A",
"FIFO_CTRL", "FIFO_SRC", "IG_CFG1", "IG_SRC1", "IG_THS1", "IG_DUR1", "IG_CFG2",
"IG_SRC2", "IG_THS2", "IG_DUR2", "CLICK_CFG", "CLICK_SRC", "CLICK_THS", "TIME_LIMIT",
"TIME_LATENCY", "TIME_WINDOW", "ACT_THS", "ACT_DUR", "MAG_SCALE_2", "MAG_SCALE_4",
"MAG_SCALE_8", "MAG_SCALE_12"];
var commands = [0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0F, 0x12, 0x13,
0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1A, 0x1B, 0x1C, 0x1D, 0x1E, 0x1F, 0x20, 0x21, 0x22,
0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2A, 0x2B, 0x2C, 0x2D, 0x2E, 0x2F, 0x30, 0x31,
0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3A, 0x3B, 0x3C, 0x3D, 0x3E, 0x3F, 0x00,
0x20, 0x40, 0x60];
var command = {};
for (var i = 0; i < commands.length; i++){
    command.merge({[topics[i]] : commands[i]});
}
delete command.merge;
msg.commands = command;
command = undefined;

return msg;
```

Figur D.29 Forbereder kommandoer.

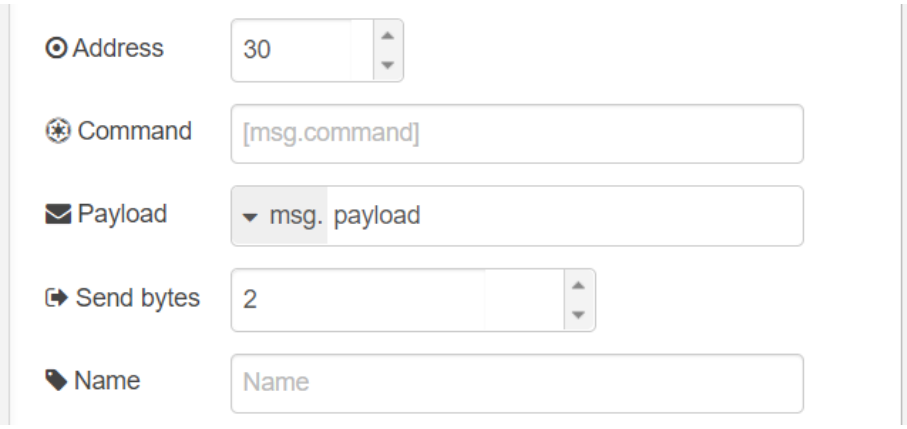
Variabellisten "*topics*" inneholder de ulike kommando-navnene, som skal settes sammen med de ulike adressene senere. For-sløyfen setter sammen de ulike kommando-navnene med de ulike adressene. Deretter blir de ulike kommandoene returnert.

Function

```
//https://github.com/DexterInd/GrovePi/blob/master/Software/Python/grove_6axis_acc_compass/lsm303d.py
// self.write_reg(0x57, self.CTRL_REG1)             # 0x57 = ODR=50hz, all accel axes on
// self.write_reg((3<<6)|(0<<3), self.CTRL_REG2)    # set full-scale
// self.write_reg(0x00, self.CTRL_REG3)             # no interrupt
// self.write_reg(0x00, self.CTRL_REG4)             # no interrupt
// self.write_reg((4<<2), self.CTRL_REG5)           # 0x10 = mag 50Hz output rate
// self.write_reg(self.MAG_SCALE_2, self.CTRL_REG6)     # magnetic scale = +/-1.3Gauss
// self.write_reg(0x00, self.CTRL_REG7)             # 0x00 = continouous conversion mode

//Source: https://github.com/Seeed-Studio/Grove_6Axis_Accelerometer_And_Compass_v2/blob/master/LSM303D.cpp

msg.command = msg.commands.CTRL_REG5;
msg.payload = (4<<2);
node.send(msg);
msg.command = msg.commands.CTRL_REG6;
msg.payload = msg.commands.MAG_SCALE_2;
node.send(msg);
msg.command = msg.commands.CTRL_REG7;
msg.payload = 0x00;
node.send(msg);
```

Figur D.30 Kjører kommandoer.

Deretter kommer "*I2C INIT MAG"*-funksjonsnoden. Den forbereder kommandoene slik at sensoren blir satt med parametere som kommentert på *Figur D.30* øverst til høyre. Dette blir sendt til I2C-out noden som setter de ønskede verdiene. Sensoren settes til 50 hertz og setter den magnetiske skalaen til pluss-minus 1.3 Gauss.



Figur D.31 Konfigurasjon av kompass-node.

Figur D.32 Start av kompass-avlesninger.

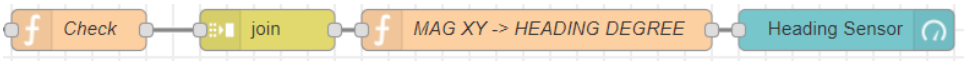
Den neste linjen er ganske lik som den forrige. Den begynner med en inject-node, for deretter den samme "*name commands*"-noden som vist ovenfor i *Figur D.32.* Deretter kommer en funksjonsnode som heter I2C getMag.

Function

```
1  msg.command = msg.commands.OUT_X_H_M;
2  node.send(msg);
3  msg.command = msg.commands.OUT_X_L_M;
4  node.send(msg);
5  msg.command = msg.commands.OUT_Y_H_M;
6  node.send(msg);
7  msg.command = msg.commands.OUT_Y_L_M;
8  node.send(msg);
9  msg.command = msg.commands.OUT_Z_H_M;
10 node.send(msg);
11 msg.command = msg.commands.OUT_Z_L_M;
12 node.send(msg);
```

Figur D.33 Ønskede ut-verdier..

Funksjonsnoden sender videre ulike kommandoer til I2C in noden, slik at den returnerer ønskete verdier. Deretter møter vi en funksjonsnode som sjekker at de er riktige verdier som kommer inn som vist på *Figur D.34.*



Figur D.34 Viser retningen til brillene i dashbordet.

Videre er det en join-node som setter sammen de siste tre verdiene til x, y og z komponenten til samme payload.

Function

```
1  //Source: https://blog.digilentinc.com/how-to-convert-magnetometer-data-into-compass-heading/
2
3  var value;
4  //The arctangent of the y/x magnetic flux vectors
5  value =Math.atan2(msg.payload.OUT_Y_M, msg.payload.OUT_X_M) * 180 / Math.PI; //Converting to Deg
6
7  //If negative value -> + 360 degrees in order to avoid negative headings
8  if(value < 0)
9  {
10 value = (value + 360);
11 }
12
13 //Only two decimals
14 msg.payload=value.toFixed(2);
15 //Sets the global variabel magheading to the msg.payload
16 global.set('Magheading',msg.payload);
17 return msg;
18
```

Figur D.35 Regner ut kompassbrikkens retning.

Den siste noden regner arcustangens av y og x komponenten til den magnetiske fluksen. Den vil deretter legge på 360 dersom vinkelen blir negativ. Deretter setter blir den globale variabelen Magheading satt til den utregnede verdien. Se *kapittel 6.9* for drøfting av resultatet.

7. Videospiller

Vi ønsket å gi brukeren muligheten for å spille av videoer. Ambisjonen er å fremvise for eksempel en navigasjonsbrief eller oppdragsbrief fra sjefen, overført trådløst. Dette gjorde vi ved å plassere ut et plan hvor vi ønsket videoen i Unity. Vi la til en videospiller-komponent som ble plassert på planet for å tilrettelegge for videoavspilling. Lenken til videoen som er lastet opp på internett blir gitt til videospilleren fra C# skriptet “VoiceCommands”. Dette gir muligheten til å starte og stoppe videoen med bruk av stemmekommandoer. Ved å starte og stoppe avspillingen vil planet med videospilleren komme til synet eller bli usynlig.

8. MyTcpClient.cs

Det blir naturlig å begynne med «*MyTcpClient*» når vi skal beskrive C# skriptene. Skriptet tar hånd om all kommunikasjon mellom server (Raspberry Pi, Node-RED) og klient (HoloLens, Unity). På grunn av kompatibilitet har vi blitt nødt til pakke inn enkelte kodesegmenter for

Unity ved bruk av plattform avhengig kompilering[16]. *Figur D.36* viser et eksempel på plattformavhengig kode:

```
24  #if !UNITY_EDITOR
25  using System.Threading.Tasks;
26  #endif
```

Figur D.36 Plattformavhengig kode..

En slik ramme gjør at koden kompileres og kjøres eksklusivt for enkelte plattformer. Dette fører til at Unity ikke gir feilmeldinger over funksjoner som ikke eksisterer i dens kontekst. Dette resulterer i at koden ikke kan feilsøkes i Unity ved bruk av feilmeldinger og advarsler. Disse kodesegmentene må feilsøkes imens programmet kjører, eller eventuelt i Visual Studio hvor det tillates.

På *Figur D.37* kan man se to lister med en rekke nøkkelord. Dette er alle variablene tilgjengelig for senere bruk og lagres lokalt på HoloLens. I listen "*AvailableVariables"* ligger navnene på alle variablene, imens i "*UpdatedValues"* ligger alle verdiene som blir registrert av HoloLens. Verdien vil være sitt eget navn frem til den blir skrevet over av en ny verdi.

```
// The available variables to read / write
// The two lists neeed to have identical indexes for variables
public List<string> AvailableVariables = new List<string>(new string[]
{"Time", "Lat", "Lon", "Knot", "Cog", "TrueHeading", "Rot", "Waypointid", "Waypointname", "Waypointlat", "Waypointlon",
    "Waypointturnradius", "Waypointlegtype", "Waypointxte", "Waypointspeed", "Waypointdanger", "Waypointmessage", "Waypointrtime", "Bearing",
    "Nextbearing", "Nextwaypointname", "Nextwaypointmessage", "Distancetowaypoint", "Distancetonextwaypoint", "Timetowaypoint","Timetonextwaypoint",
    "Voicewaypoint","Element", "Moveby", "Direction", "Movenow", "Magheading", "Red", "Green", "Blue", "Alpha", "Texttospeech", "Fontsize", "Delay",
    "Automaticwaypoint", "Voiceassistance", "Xte", "Prevwaypointname" });

// The values will be written to this list
public List<string> UpdatedValues = new List<string>(new string[]
{"Time", "Lat", "Lon", "Knot", "Cog", "TrueHeading", "Rot", "Waypointid", "Waypointname", "Waypointlat", "Waypointlon",
    "Waypointturnradius", "Waypointlegtype", "Waypointxte", "Waypointspeed", "Waypointdanger", "Waypointmessage", "Waypointrtime", "Bearing",
    "Nextbearing", "Nextwaypointname", "Nextwaypointmessage", "Distancetowaypoint", "Distancetonextwaypoint", "Timetowaypoint","Timetonextwaypoint",
    "Voicewaypoint", "Element", "Moveby", "Direction", "Movenow", "Magheading", "Red", "Green", "Blue", "Alpha", "Texttospeech", "Fontsize", "Delay",
    "Automaticwaypoint", "Voiceassistance", "Xte", "Prevwaypointname" });
```

Figur D.37 Lister med nøkkelord og avleste verdier.

På *Figur D.39* kan vi se funksjonen *ExchangePackets()* som står for avlesning av alle variabler. Denne metoden blir helt i starten av skriptet satt til å kjøre ti ganger i sekundet som vist på *Figur D.38.* HoloLens sender «*GiveMeUpdate*» som en streng til serveren. Dette får serveren til å respondere med nåværende verdier på alle variabler som HoloLens trenger. Svaret blir

[16] Platform dependent compilation: https://docs.unity3d.com/Manual/PlatformDependentCompilation.html se også på #if, #elif, #else, and #endif directives (C/C++): https://docs.microsoft.com/en-us/cpp/preprocessor/hash-if-hash-elif-hash-else-and-hash-endif-directives-c-cpp?view=vs-2019

lagret i strengen «*PreviousResponse*». Deretter blir strengen splittet på tegnene '|' og ':', og dens verdier blir filtrert og plassert i korrekt variabel.

```
// Calls the function "ExchangePackets" after a two second delay @10Hz
InvokeRepeating("ExchangePackets", 2.0f, 0.1f);
```

Figur D.38 Etterspør data fra server ti ganger i sekunder.

.

```
public async void ExchangePackets()
{

    try
    {
        // Sends "GiveMeUpdate" to the server
        writer.Write("GiveMeUpdate");

        // Writes the recieved message to PreviousResponse
        PreviousResponse = await reader.ReadLineAsync();

        //Writes response to console in Visual Studio
        Debug.Log(PreviousResponse);

        //Splits the string twice and updates the variables with their new values
        string[] elements = PreviousResponse.Split('|');
        int index = 0;
        foreach (var elem in elements)
        {
            if (elem == "")
                continue;
            string[] split = elem.Split(':');
            index = findIndex(AvailableVariables, split[0]);
            UpdatedValues[index] = split[1];
            index++;
        }
    }

    catch (Exception e)
    {
        Debug.Log(e.ToString());
    }
}
```

Figur D.39 Leser variabler inn i listene vist på Figur D.37.

Under kan vi se eksempel på en PreviousResponse:

"Time:93559.25|Lat:60.39872|Lon:5.26885|Knot:12.8|Cog:165.1|TrueHeading:160.47|Rot:-663|Waypointid:2|Waypointname:TP248FM2bb>MK|Waypointlat:60.402111|Waypointlon:5.241183|Waypointturnradius:0.0500|Waypointlegtype:0|Waypointxte:0.0972|Waypointspeed:18.00|Waypointdanger:-

1|Waypointmessage:undefined|Waypointrtime:201.73|Bearing:306|Nextbearing:238|Nextwaypointname:TP202FM1bb>VBK|Nextwaypointmessage:undefined|Distancetowaypoint:0.52|Distancetonextwaypoint:2.28|Timetowaypoint:02m26|Timetonextwaypoint:10m41|Voicewaypoint:undefined|Element:Allelements|Moveby:0|Direction:Up|Movenow:True|Magheading:undefined|Red:108|Green:255|Blue:75|Alpha:255|Texttospeech:Putyourtexthere|Fontsize:14|Delay:1|Automaticwaypoint:undefined|Voiceassistance:undefined|Xte:0.78|Prevwaypointname:>S "

Funksjonen *findIndex*(), *Figur D.40,* er funksjonen som finner og returnerer rett indeks fra «*AvailableVariables*» til variabelen som skal skrives til. Dersom variabelen ikke ble funnet vil verdien "-1" bli returnert.

```
// Returns index of a given keyword
int findIndex(List<string> myList, string myString)
{
    for (int i = 0; i < myList.Count; i++)
    {
        if (myList[i] == myString)
        {
            return i;
        }
    }
    return -1;
}
```

Figur D.40 Returnerer indeksen til nøkkelordet..

Funksjonene «*forceSend*» brukes for å sende data fra HoloLens til serveren (Les: Node-RED). Funksjonen vises på *Figur D.41* og eksempel på bruk av funksjonen kan ses på *Figur D.42,* der den blir brukt i funksjonen *«Voicecommands»*

```
// Sends a given string to the server. This is used to set (/change) variables on the server
public async void forceSend(string input)
{
    writer.Write(input);
    Debug.Log("forceSend: " + input);
    PreviousResponse = await reader.ReadLineAsync();
}
```

Figur D.41 Sender angitt melding til serveren.

```
//Makes the user able to switch to next and previous
void NextWaypoint()
{
    MyTcpClient.forceSend("Voicewaypoint:Next");
    int nextWaypoint = int.Parse(MyTcpClient.giveMe("Waypointid")) + 1;
    TextToSpeech.Say("Waypoint number " + nextWaypoint);
}
```

Figur D.42 Ber serveren bytte til neste veipunkt.

Strengen «*Voicewaypoint:Next*» blir sendt til serveren hvor strengen blir videre behandlet av følgende kode vist i *Figur D.43.* Funksjonen *NextWaypoint()*, bruker funksjonen forceSend til å sende strengen «*Voicewaypoint:Next*» som skal indikere at systemet skal bytte waypoint vist i brillesettet. Dersom strengen blir sendt med et kolon (‘ : ’) vil den tolkes slik: <variabelnavn>:<verdi>.

Name: Check Request

Function

```
//Available variables
var keyword=[  "Time", "Lat", "Lon", "Knot", "Cog", "TrueHeading", "Rot", "Waypointid", "Waypointname", "Waypointlat", "Waypointlon",
"Waypointturnradius", "Waypointlegtype", "Waypointxte", "Waypointspeed", "Waypointdanger", "Waypointmessage", "Waypointrtime", "Bearing",
"Nextbearing", "Nextwaypointname", "Nextwaypointmessage", "Distancetowaypoint", "Distancetonextwaypoint", "Timetowaypoint", "Timetonextwaypoint", "Voicewaypoint",
"Element", "Moveby", "Direction", "Movenow", "Magheading", "Red", "Green", "Blue", "Alpha", "Texttospeech", "Fontsize", "Delay", "Automaticwaypoint", "Voiceassistance" ,
"Xte", "Prevwaypointname"];

//Writes data to variables with given value after ':', sent from the Hololens
if(msg.payload.includes(":")){
var splitstring = msg.payload.split(':');

for (i = 0; i < keyword.length; i++) {
if (splitstring[0]==keyword[i]){
    global.set(splitstring[0],splitstring[1]); //Example: "Voiceassistance:True" --> Sets the global variable Voiceassistance to True
    }
}
}
```

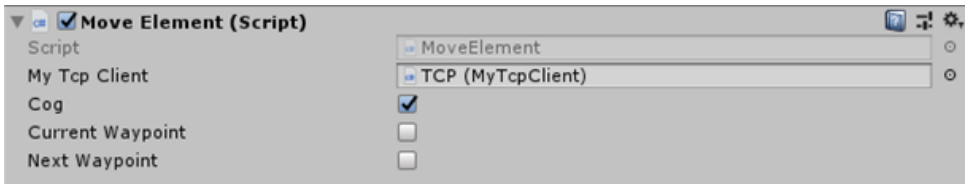
Figur D.43 Behandler klientens forespørsel.

9. MoveElement.cs

Dersom man sender «*Movenow:True*» vil variabelen «*Movenow*» bli satt til «*True*». Dernest vil en ny oppdatert variabelliste bli sendt til HoloLens hvor den blir mottatt og behandlet. For å kunne stille på objektene hver for seg ble må man ha muligheten til å skille på dem. Vi løste dette ved å legge ved en avkryssingsrute som vist i *Figur D.45* for hvert av objektene vi ønsket å flytte på. Disse er definert som vist på *Figur D.44.*

```
// Tick the variable in Unity to match the element
public bool cog;
public bool currentWaypoint;
public bool nextWaypoint;
```

Figur D.44 Variabel som angir objektets identitet (C#).



Figur D.45 Velger objektets identitet (Unity).

For å ha kontroll når flytting blir gjort la vi til en variabel som vi kalte «*Movenow*». Dersom denne settes til «*True*», vil skriptet flytte det valgte elementet.

```
// If the variable is set to "True" the function will continue
if (Movenow == "True" && !moved)
```

Figur D.46 Sjekker om noe skal flyttes.

```
//Checks which element should move
bool Match = false;
string Element = MyTcpClient.giveMe("Element");
if ((Element == "Cog" && cog) || (Element == "Currentwaypoint" && currentWaypoint) ||
    (Element == "Nextwaypoint" && nextWaypoint) || (Element=="Allelements"))
    Match = true;
```

Figur D.47 Sjekker hva som skal flyttes.

Dersom det har blitt bestemt at objektet skal flyttes, leses retning og lengde på flyttingen av som vist i *Figur D.49.* Merk at retningene er relativt til objektets plassering og orientering. Vi valgte å «*oversette*» retningene i forhold til brukerens perspektiv av objektets egne retninger. For eksempel er «*venstre*» for brukeren «*frem*» for objektet. For å forhindre at objektet blir flyttet på flere ganger per trykk krever vi at variabelen «*Movenow*» har hatt en tilstand utenom «*True*» i etterkant av flytt før nytt flytt kan gjennomføres, logikken er presentert i *Figur D.48.*

```
// Get variable
string Movenow = MyTcpClient.giveMe("Movenow");
if (Movenow != "True")
    moved = false;
```

Figur D.48 Sjekker «Movenow»-variabelen.

```
if (Match)
{
    // Writes to console in Visual Studio
    Debug.Log("Starting to move object");

    // Moveby is how far
    string Movebystring = MyTcpClient.giveMe("Moveby");
    float Moveby = float.Parse(Movebystring);

    // Direction is one of six directions (think of a dice)
    string Direction = MyTcpClient.giveMe("Direction");

    // Defines which component we wish to move (in this case it's the object which the script is attached to)
    RectTransform myRectTransform = GetComponent<RectTransform>();

    // Towards user
    if (Direction == "Towards")
        myRectTransform.localPosition += Vector3.up * Moveby;

    // Away from user
    else if (Direction == "Away")
        myRectTransform.localPosition += Vector3.down * Moveby;

    // down
    else if (Direction == "Down")
        myRectTransform.localPosition += Vector3.left * Moveby;

    // up
    else if (Direction == "Up")
        myRectTransform.localPosition += Vector3.right * Moveby;

    // right
    else if (Direction == "Right")
        myRectTransform.localPosition += Vector3.back * Moveby;

    // left
    else if (Direction == "Left")
        myRectTransform.localPosition += Vector3.forward * Moveby;

    // Sets the variable Movenow to "False", this stops it from looping.
    // There are some delays with setting the variables on the RaspberryPi-server (NODE-Red)
    // Therefore we check if the element has been moved already.
    // In the current context it does not matter due to the low refreshrate of 0.5Hz.

    moved = true;
    MyTcpClient.forceSend("Movenow:False");
}
```

Figur D.49 Gjennomfører flytt i ønsket retning – med ønsket lengde.

10. VoiceCommands.cs

Kodesnutten som vist i *Figur D.50* står for opprettelsen av stemmegjenkjenning og listen over godkjente fraser.

```
// Voice command vars
private Dictionary<string, Action> keyActs = new Dictionary<string, Action>();
private KeywordRecognizer recognizer;

// Start is called before the first frame update
void Start()
{
    Debug.Log("Starting voicecommands!");

    //Avaiable voicecommands
    keyActs.Add("enable voice", EnableVoice);
    keyActs.Add("disable voice", DisableVoice);
    keyActs.Add("next waypoint", NextWaypoint);
    keyActs.Add("previous waypoint", PreviousWaypoint);
    keyActs.Add("thank you", ThankYou);
    keyActs.Add("toggle markers", Toggle45);
    keyActs.Add("play video", PlayVideo);
    keyActs.Add("stop video", StopVideo);

    //Starts the voicerecognizer
    recognizer = new KeywordRecognizer(keyActs.Keys.ToArray());
    recognizer.OnPhraseRecognized += OnKeywordsRecognized;
    recognizer.Start();
    Debug.Log("Voicecommands started!");

}
```

Figur D.50 Ordbok for stemmegjenkjenning.

Stemmegjenkjennings-funksjonen gjør at HoloLens kontinuerlig lytter etter disse nøkkelordene. Siden ordene er gitt på forhånd blir gjenkjenningen lettere da den vet hvilke ord den leter etter. Dersom gjenkjenneren hører brukeren si «*toggle markers*» vil funksjonen «*Toggle45()*» bli kalt på. Denne funksjonen styrer de røde og grønne markørene som peker henholdsvis 45 og 90 grader på hver sin side av fartøyet. Funksjonen veksler markørene av og på. I tillegg får man tilbakemelding fra HoloLens i form av en stemme som sier «*Markers toggled on/off*», kodesnutten er vedlagt i *Figur D.51*.

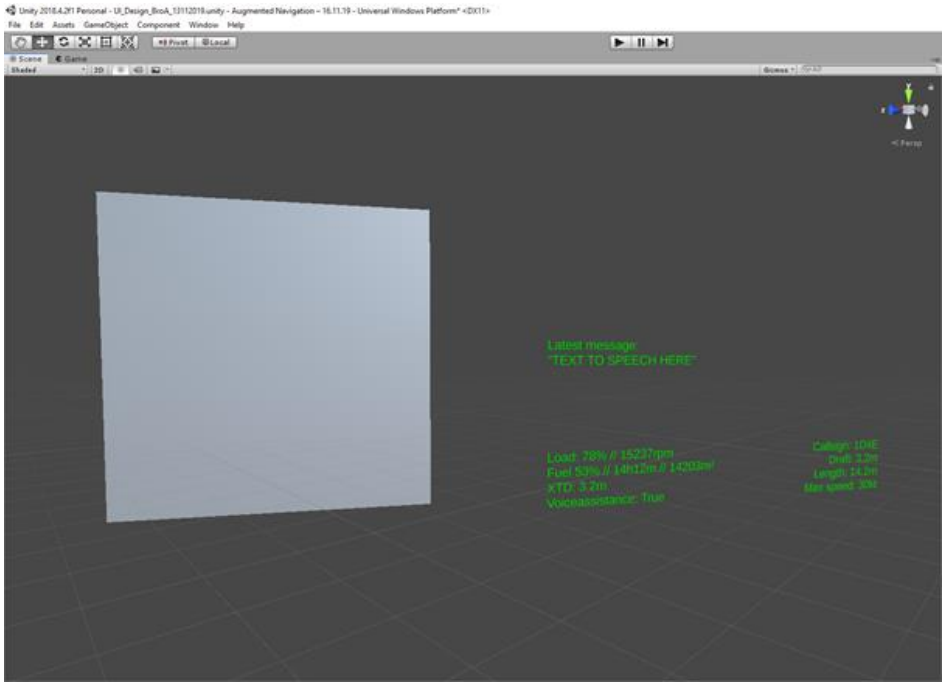
```
//Makes user able to toggle 45°- and 90°-degree markers
void Toggle45()
{
    Renderer Red45 = GameObject.Find("Red45").GetComponent<Renderer>(); ;
    Renderer Green45 = GameObject.Find("Green45").GetComponent<Renderer>();
    Renderer Red90 = GameObject.Find("Red90").GetComponent<Renderer>(); ;
    Renderer Green90 = GameObject.Find("Green90").GetComponent<Renderer>();

    if (isVisible)
        isVisible = false;
    else isVisible = true;

    Red45.enabled = isVisible;
    Green45.enabled = isVisible;
    Red90.enabled = isVisible;
    Green90.enabled = isVisible;
    if (isVisible)
        status = "on";
    else status = "off";

    TextToSpeech.Say("Markers toggled " + status);
}
```

Figur D.51 Veksler markører av og på..

Funksjonene for videospilleren kalles på ved å si henholdsvis «*play video*» eller «*stop video*». Det virtuelle lerretet kan sammenlignes med et lerret som kan trekkes opp og ned, samt vise video fra en prosjektor, funksjonene er vist på *Figur D.52* og lerretet er vist i *Figur D.53.*

```
    //Makes user able to play and stop video
    void PlayVideo()
    {
        GameObject plane = GameObject.Find("Plane");
        Renderer planerend = plane.GetComponent<Renderer>();
        planerend.enabled = true;
        var videoPlayer = plane.AddComponent<UnityEngine.Video.VideoPlayer>();
        videoPlayer.url = "<DIRECTLINK TO VIDEO (.mp4)>";
        videoPlayer.Play();
        TextToSpeech.Say("Video playing");
    }

    void StopVideo()
    {
        GameObject plane = GameObject.Find("Plane");
        Renderer planerend = plane.GetComponent<Renderer>();
        planerend.enabled = false;
        var videoPlayer = plane.GetComponent<UnityEngine.Video.VideoPlayer>();
        Destroy(videoPlayer);
    }
}
```

Figur D.52 Veksler video av og på.



Figur D.53 Sidepanel vist i Unity.

11. UpdateText.cs

"*UpdateText*" skriptet ble laget for å formatere og vise frem data i scenen. Vi oppdaget fort at enkelte symboler ble tolket feil etter de har blitt overført fra server til klient. Vi leste av strengene og fant ut at følgende symboler som *Figur D.54* viser ble mottatt feil.

Symbol sendt	Streng mottatt
>	&gt;
<	>
/	/

Figur D.54 Forvrengte symboler..

Dette løste vi ved å erstatte strengene med sine tilhørende symboler på følgende måte med streng-funksjonen «*Replace()*».

```
string currentwaypointstring = MyTcpClient.giveMe("Prevwaypointname");
// Fixes symbols from TCP-communication
// In the TCP-communication some symbols gets interpeted wrong and needs to be corrected.
// These are the symbols that we found needed to be replaced
currentwaypointstring = currentwaypointstring.Replace("&amp;gt;", ">");
currentwaypointstring = currentwaypointstring.Replace("&amp;lt;", "<");
currentwaypointstring = currentwaypointstring.Replace("&#x2f;", "/");
```

Figur D.55 Erstatter forvrengte symboler.

På inneværende kurs ønsket vi kun å lese av stevnet, altså det kom etter enten '>'- eller '<'-tegnet. Dette ble også løst med streng-funksjonen «.*Contains()*» som vi viser i *Figur D.56.*

```
// Reads the string after '>' or '<'
if (currentwaypointstring.Contains(">"))
    currentwaypointstring = ">" + currentwaypointstring.Substring(currentwaypointstring.LastIndexOf('>') + 1);
else if (currentwaypointstring.Contains("<"))
    currentwaypointstring = "<" + currentwaypointstring.Substring(currentwaypointstring.LastIndexOf('<') + 1);
```

Figur D.56 Leser strenger før og etter skilletegn.

Til slutt ble teksten som skulle vises formatert riktig og skrevet til tekstfeltet i riktig datafelt i brillesettet.

```
// Formats text to display
textToScreen = currentwaypointstring + "\n" +
MyTcpClient.giveMe("Bearing") + "°\n" +
MyTcpClient.giveMe("Distancetowaypoint") + "nm\n";
```

Figur D.57 Formaterer tekst til skjerm.

Vi har mulighet for å få HoloLens til å lese opp turninformasjon ved bytte av veipunkt. Som en konsekvens var vi nødt til å vite om veipunktet hadde endret seg eller ikke, da vi ikke ønsket kontinuerlige påminnelser om neste leg. Dersom veipunktet ikke hadde byttet, vil ikke skriptet gå inn i if-sløyfen vist i *Figur D.58.*

```
// Checks if it's a new waypoint or "just the same as last time"
if (lastwaypointstring != nextwaypointstring)
{
    message = true;
    lastwaypointstring = nextwaypointstring;
```

Figur D.58 Sjekker om veipunkt er blitt byttet.

Vi leser av estimert tid til turn fra server ved hjelp av koden i

```
// Example time = "14m36"
string nexttime = MyTcpClient.giveMe("Timetonextwaypoint");
string[] nextminutesandseconds = nexttime.Split('m');
nextminutes = int.Parse(nextminutesandseconds[0]);
nextseconds = int.Parse(nextminutesandseconds[1]);
```

Figur D.59 Tolker tiden fra serveren.

Vi oppdaget at navigatørene var vant til å si kursene ett og ett siffer fremfor hele tallet. («*en tre fire grader*» fremfor «*hundre og trettifire grader*»).

For at tekst til stemme-funksjonen skulle lese på denne måten måtte vi legge inn mellomrom mellom hvert tall, dette ble gjort som vist i *Figur D.60.*

```
string nextbearing = MyTcpClient.giveMe("Nextbearing");
// Adds a space between the numbers of the bearing for later use in TextToSpeech(TTS)
// Instead of saying "Onehundred and thirtyfour" it should say "one three four"
foreach (char c in nextbearing)
{
    newnextbearing += c;
    newnextbearing += ' ';
}
```

Figur D.60 Legger mellomrom mellom tall

Kursnotasjonen blir delt opp til før og etter '>' eller '<'.

```
if (nextwaypointstring.Contains(">"))
    nextwaypointstring = nextwaypointstring.Substring(0, nextwaypointstring.IndexOf('>'));
else if (nextwaypointstring.Contains("<"))
    nextwaypointstring = nextwaypointstring.Substring(0, nextwaypointstring.IndexOf('<'));

if (nextwaypointstring2.Contains(">"))
    nextwaypointstring2 = ">" + nextwaypointstring2.Substring(nextwaypointstring2.LastIndexOf('>') + 1);
else if (nextwaypointstring.Contains("<"))
    nextwaypointstring2 = "<" + nextwaypointstring2.Substring(nextwaypointstring2.LastIndexOf('<') + 1);
```

Figur D.61 Deler strengen i to på skilletegn.

Dersom tiden blir gitt som 99 minutter og 99 sekunder vil HoloLens tolke dette som at fartøyet ikke har fart. Dette ble løsningen da det var lett å implementere i koden som allerede lå til grunn. Dersom vi sendte tiden i et annet format enn <x>m<y> fikk ulike deler av programmet problemer med å parse strengen til et heltall. Det var også svært lett å feilsøke da det i utgangspunktet ikke er mulig å få mer enn 60 sekunder før det blir et ekstra minutt. På *Figur D.62* kan vi se hvordan det blir skrevet "*99m99*" dersom tiden er uendelig. *Figur D.63* viser hvordan "*99m99*" blir tolket og at "*NO SPEED*" blir vist på skjermen.

```
if (Timetowaypoint == "InfinitymNaN")
global.set("Timetowaypoint","99m99");
```

Figur D.62 Tid til veipunkt settes i Node-RED.

```
    // We have an edge case where the vessel is standing still, and therefore has no speed
    // This messes with some calculations and especially parsing of the variables to numbers
    // This is solved by having the server give the time "99m99" when the time is in reality "infinite"
    // Having 99 seconds should be impossible and only achievable with it being forced like this
if (currentminutes == 99 && currentseconds == 99)
    // Formats text to display
    textToScreen = "NO SPEED";
```

Figur D.63 Tid til veipunkt tolkes i C#.

Dersom fartøyet har fart, vil tid til turn vises – og dersom det er mindre enn tre minutter til turn vil: kursnotasjon, neste kurs, avstand og tid vises. Koden som formaterer teksten, er vist i *Figur D.64 og Figur D.65* viser hvordan tiden blir sjekket.

```
else
textToScreen = currentminutes +"m"+currentseconds+"s" + "\n";
if (checktime())
{
    // Formats text to display
    textToScreen = textToScreen + nextwaypointstring + "\n" +
         MyTcpClient.giveMe("Nextbearing") + "°\n" +
         MyTcpClient.giveMe("Distancetonextwaypoint") + "nm\n" +
         nextwaypointstring2 + "\n" +
         nextminutes + "m" + nextseconds + "s";
}
```

Figur D.64 Formaterer tekst til neste kurs.

```
// Returns true if under treshhold, false is over
bool checktime()
{
    // Example time = "14m36"
    string time = MyTcpClient.giveMe("Timetowaypoint");
    string[] minutesandseconds = time.Split('m');
    // [0] = 14 and [1] = 36
    currentminutes = int.Parse(minutesandseconds[0]);
    currentseconds = int.Parse(minutesandseconds[1]);

    // Check if remaining time is under the given treshhold.
    int totalseconds = currentminutes * 60 + currentseconds;
    if (totalseconds <= 180)
        return true;
    else
        return false;
}
```

Figur D.65 Sjekker tiden til turn..

Hvordan dette til slutt vil se ut for brukeren er vist i *Figur D.66.* Dersom tiden er under grensen, som her er satt til tre minutter (180 sekunder) vil funksjonen *checktime*() returnere true, dersom det er lengre enn terskelen, vil false returneres. I praksis vil dette føre til at kun det over den gule streken vises frem til det er tre minutter igjen. Når det er under 180 sekunder igjen, vil "*Checktime*" returnere true. Dette fører til at resten av kursinformasjonen vises i tillegg til at den leses opp dersom stemmeassistanse er skrudd på.

Figur D.66 Før og etter tidsgrense..

12. TextToSpeech.cs

Vi så først på bruk av lyder for tilbakemeldinger til brukeren fra systemet. Dette ledet til at vi lastet ned lydklipp med «*tekst til tale*». Vi fant ut at det ble tungvint å implementere et lydklipp for hver enkelt tilbakemelding. Det ble da naturlig å se til direkte implementering av tekst til tale. Dette gjorde vi ved bruk av Azure sine kognitive tjenester (*Azure Cognitive Services*). For å få tilgang på disse tjenestene, må man opprette en bruker og ha et aktivt abonnement. Nøkkelen fra abonnementet fører man inn i *speechConfig* i koden. Etter kodesnutten vist i *Figur D.67* vil HoloLens kunne benytte seg av tekst til tale tjenesten.

```
void Start()
{
    // Creates an instance of a speech config with specified subscription key and service region.
    // Replace with your own subscription key and service region (e.g., "westus").
    speechConfig = SpeechConfig.FromSubscription("<YOURSUBSCRIPTIONKEY>", "northeurope");

    // The default format is Riff16Khz16BitMonoPcm.
    // We are playing the audio in memory as audio clip, which doesn't require riff header.
    // So we need to set the format to Raw16Khz16BitMonoPcm.
    speechConfig.SetSpeechSynthesisOutputFormat(SpeechSynthesisOutputFormat.Raw16Khz16BitMonoPcm);

    // Creates a speech synthesizer.
    // Make sure to dispose the synthesizer after use!
    synthesizer = new SpeechSynthesizer(speechConfig, null);
    InvokeRepeating("ToldToSay", 2.0f, 1f);
    Say("Hololens ready");
}
```

Figur D.67 Oppstart av stemmegjenkjenner med abonnement.

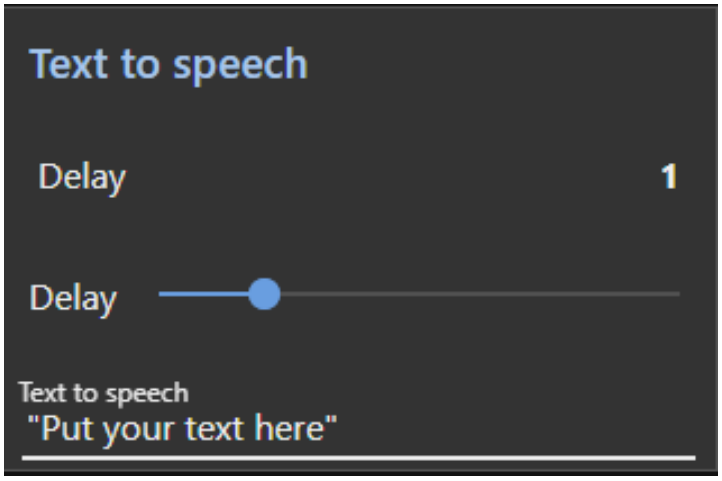
Store deler av skriptet på *Figur D.68* er kopiert fra Microsoft sine ressurser[17], men tilpasset slik at det tar en streng som inn-argument. Denne funksjonen tillater oss å lese opp strenger fra andre steder i programmet. Argumentet til "Say"-funksjonen er til slutt det som ender opp med å bli lest opp.

[17] https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/quickstart-text-to-speech-csharp-unity

```
public void Say(string input)
{
    Debug.Log("Text to speech: " + input);
    lock (threadLocker)
    {
        waitingForSpeak = true;
    }

    string newMessage = string.Empty;

    // Starts speech synthesis, and returns after a single utterance is synthesized.
    using (var result = synthesizer.SpeakTextAsync(input).Result)
    {
        // Checks result.
        if (result.Reason == ResultReason.SynthesizingAudioCompleted)
        {
            // Native playback is not supported on Unity yet (currently only supported on Windows/Linux Desktop).
            // Use the Unity API to play audio here as a short term solution.
            // Native playback support will be added in the future release.
            var sampleCount = result.AudioData.Length / 2;
            var audioData = new float[sampleCount];
            for (var i = 0; i < sampleCount; ++i)
            {
                audioData[i] = (short)(result.AudioData[i * 2 + 1] << 8 | result.AudioData[i * 2]) / 32768.0F;
            }

            // The output audio format is 16K 16bit mono
            var audioClip = AudioClip.Create("SynthesizedAudio", sampleCount, 1, 16000, false);
            audioClip.SetData(audioData, 0);
            audioSource.clip = audioClip;
            audioSource.Play();

            newMessage = "Speech synthesis succeeded!";
        }
        else if (result.Reason == ResultReason.Canceled)
        {
            var cancellation = SpeechSynthesisCancellationDetails.FromResult(result);
            newMessage = $"CANCELED:\nReason=[{cancellation.Reason}]\nErrorDetails=[{cancellation.ErrorDetails}]\nDid you update the subscription info?";
        }
    }

    lock (threadLocker)
    {
        message = newMessage;
        waitingForSpeak = false;
    }
}
```

Figur D.68 HoloLens leser opp gitt tekst.

Fra kodesnutt som vist på *Figur D.70* har vi funksjonen «*ToldToSay*» som leser av verdien fra «*Text to speech*» gruppen i brukergrensesnittet til serveren som vist i *Figur D.69*. Brukeren får først et varsel på at det er en innkommende melding før meldingen blir lest opp. Det er også lagt inn en sjekk om «*ToldToSay*» allerede prater ved den boolske variabelen «*isTalking*». I tillegg er det lagt inn en forsinkelse imellom brukeren får varselet og meldingen blir lest opp.



Figur D.69 Setter forsinkelse og gir tekst til HoloLens.

```
    // Function that reads (TTS) a given string from the server
    public async void ToldToSay()
    {
        currenttts = MyTcpClient.giveMe("Texttospeech");
        if (currenttts!=oldtts && currenttts != "" && currenttts != "Texttospeech" && !isTalking)
        {
            isTalking = true;
            Say("Message incoming.");
            oldtts = currenttts;
            Debug.Log("Told to say: " + currenttts);

            // This delay is set by the server, we have had success with ~1 second and longer delay.
#if !UNITY_EDITOR
            double delay = double.Parse(MyTcpClient.giveMe("Delay"));
            await Task.Delay(TimeSpan.FromSeconds(delay));
#endif
            Say(currenttts);
            isTalking = false;
        }
    }
```

Figur D.70 Leser av sendt melding fra server..

Funksjonen «*AutomaticNext()*», som vist på *Figur D.71*, gir brukeren tilbakemelding om at kursen er nådd og at veipunktet blir skiftet automatisk. Merk at denne funksjonaliteten ikke ble ferdigstilt og fullstendig feilsøkt. Her brukes også «*Say*»-funksjonen.

```
//Gives the user feedback if the waypoint was automatically switched.
public async void AutomaticNext()
{
    if (MyTcpClient.giveMe("Automaticwaypoint") == "True" &&
        (Waypointid != int.Parse(MyTcpClient.giveMe("Waypointid"))) && !isTalking)
    {
        isTalking = true;
        Waypointid = int.Parse(MyTcpClient.giveMe("Waypointid"));
        MyTcpClient.forceSend("Automaticwaypoint:False");
        Say("Waypoint reached, advancing to waypoint number" + Waypointid);
        isTalking = false;
    }
}
```

Figur D.71 Informerer brukeren at veipunktet har byttet automatisk.

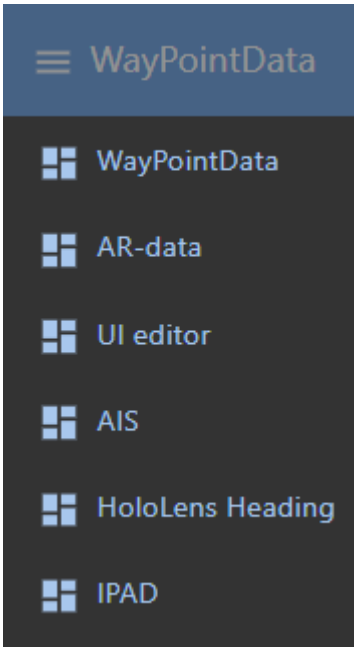
Vedlegg E: Beskrivelse av UI
Bruksanvisning server

Det grafiske brukergrensesnittet er primært tiltenkt til å endre enkelte parametere, samt feilsøke eventuelle feil under utvikling eller bruk. For å komme inn til betjeningspanelene går man inn på IP-adressen til serveren med tillegget «/ui».
Eksempelvis: http://192.168.136.48:1880/ui/
Det er totalt seks forskjellige faner som vist på bildet under. Trykk på de tre horisontale strekene øverst til venstre og trykk på ønsket fane for å bytte.

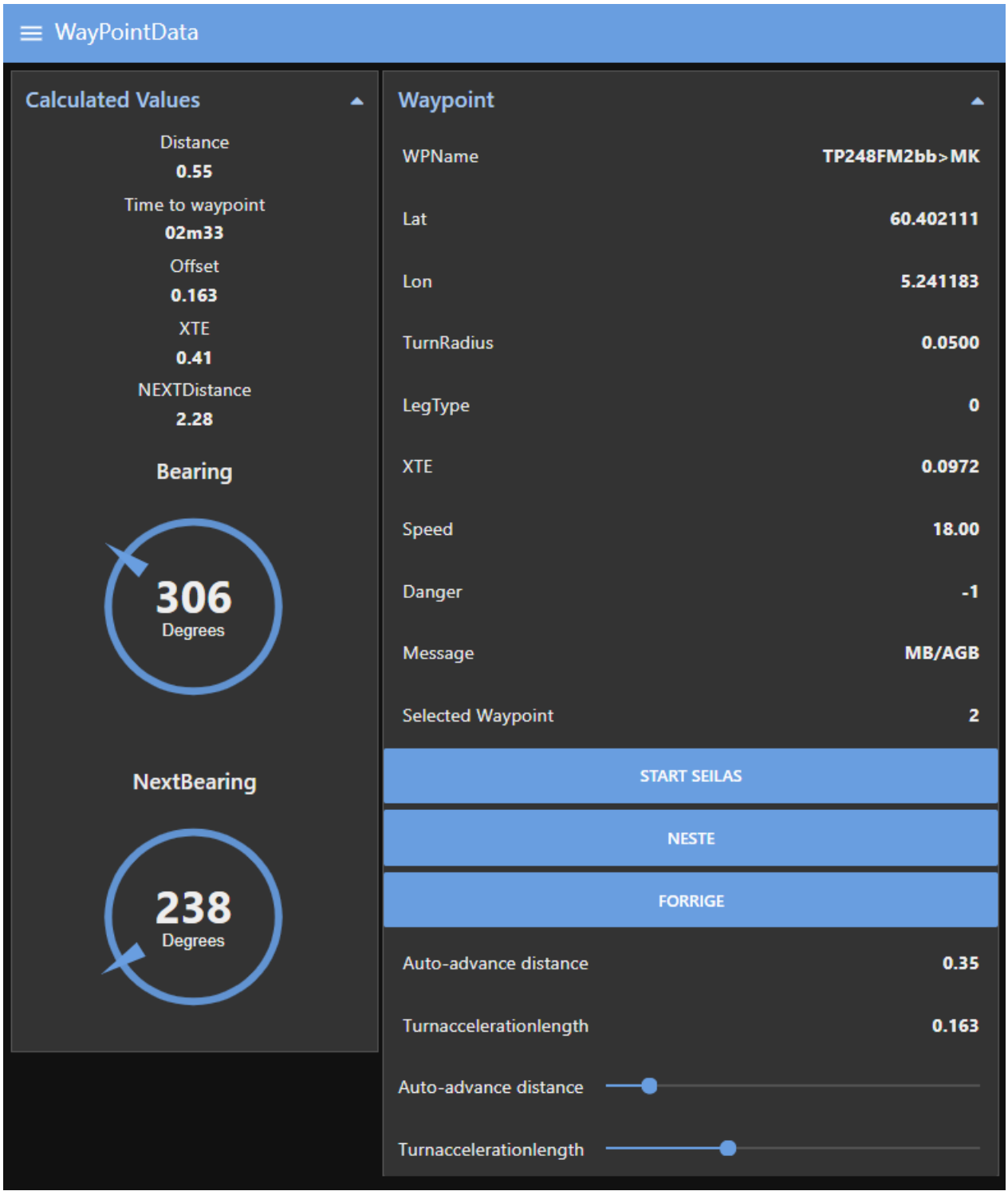
WayPointData

På venstre side under «*Calculated Values*» finner vi et utvalg utregninger. På høyresiden under fanen WayPointData kan man lese av ett og ett veipunkt fra ruteplanleggingsfilen, som er lastet opp. Knappen «*START SEILAS*» stiller ruteansvisningen tilbake til start. De to knappene under er ganske selvforklarende. «*NESTE*» bytter valgt veipunkt til neste, og «*FORRIGE*» bytter til forrige veipunkt.

Under knappene finner man to skyvere hvor man kan stille inn ønsket avstand hvor veipunktene skal skifte automatisk innenfor. Merk at denne funksjonaliteten ikke er ferdig, og ikke i bruk.



Figur E.1 Tilgjengelige faner.



Figur E.2 WayPointData-fanen.

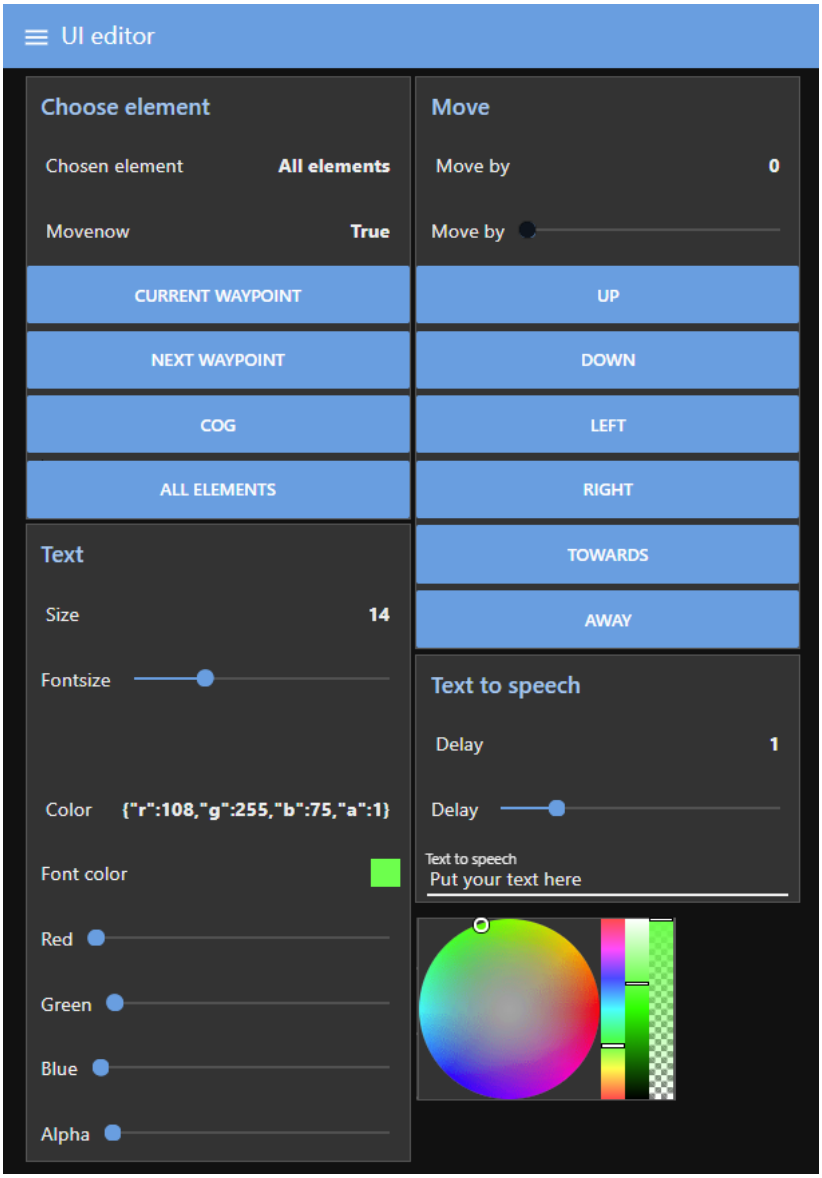
AR-data

Denne fanen blir brukt til å vise hvilke data som blir lest av i sanntid. Til venstre ser man sannkurs, kurs over bakken og svinghastighet. Til høyre ser man farten til fartøyet, vindvinkelen og vindhastigheten. Nederst kan man se Fartøyets posisjon og tid. Merk at denne tiden blir lest av simulatoren, og vil være tiden i scenarioet.

Figur E.3 AR-data-fanen.

UI editor

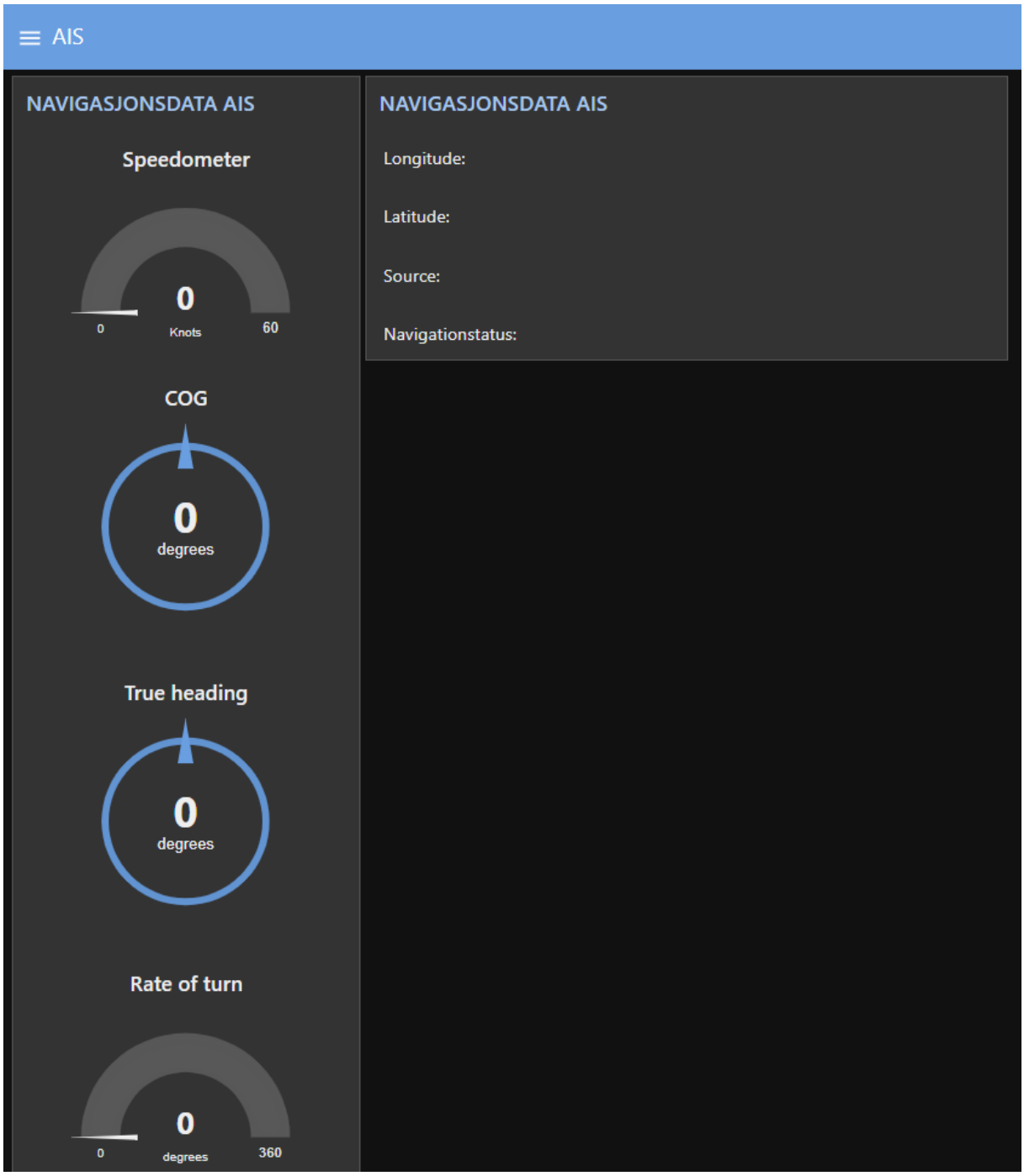
Denne fanen brukes til å tilpasse brukergrensesnittet i sanntid. Under «*Choose element*» velger man hvilket tekstfelt som skal flyttes. Til høyre velger man hvor langt og hvilken retning tekstfeltet skal flyttes. Det blir flyttet hver gang man trykker på en retning. Under «*Text*» velges størrelse på font. Videre ned finner man mulighet for å endre farge på teksten. Dette kan gjøres enten ved å stille på hver enkelt komponent eller ved å bruke fargevelgeren som vist nederst til høyre. Under «*Text to speech*» kan man velge forsinkelse mellom varsel og melding, samt sende en melding til HoloLens som vil bli lest opp for brukeren. Dette gjøres ved å fylle inn ønsket melding under «*Text to speech*» inndatafeltet og trykke enter.



Figur E.4 UI Editor-fanen.

AIS

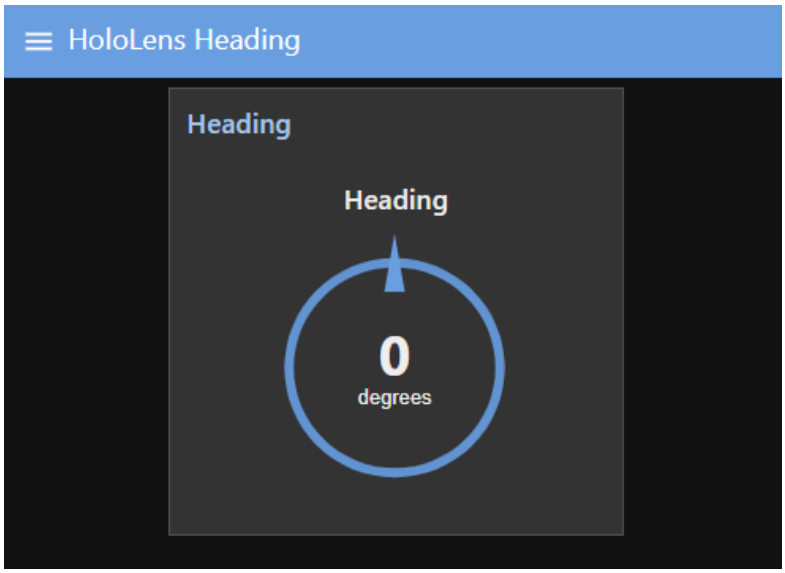
Denne fanen er ikke per nå ikke i bruk, men fult funksjonell. Dersom man kobler serveren til en USB som gir AIS-strenger vil dataene bli filtrert og vist frem i denne fanen. Disse verdiene er dog ikke koblet oppimot HoloLens.



Figur E.5 AI-fane uten inndata.

HoloLens Heading

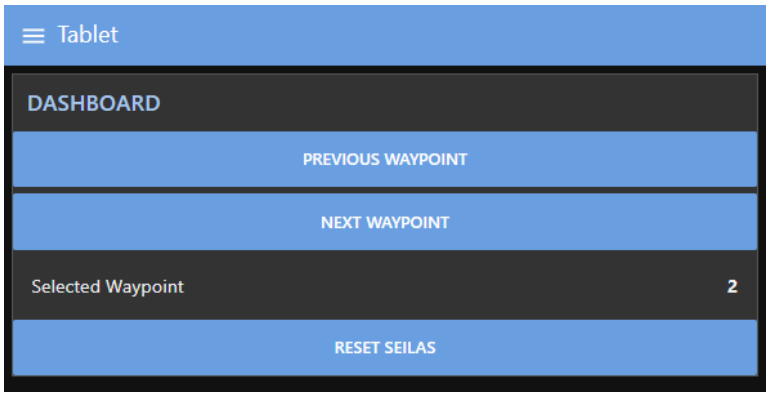
Denne fanen er for å lese av HoloLens sin retning med bøylen installert og koblet til. Denne verdien fant aldri helt veien til brukeren annet enn denne fanen. Ved videre utvikling ville denne verdien blitt vist som et fast element i synsfeltet som kunne skrus av og på av brukeren.



Figur E.6 Heading gitt med kompass-pil.

Tablet

Den siste fanen ble påtenkt, designet og produsert imellom første og andre testing. Den initielle tanken var å lage en matrise med 3X3 trykkeknapper. Vi kom ikke i mål med knappene, og lagde disse virtuelle knappene for å erstatte deres hovedoppgave. Denne fanen er tiltenkt å åpnes på et nettbrett eller lignende slik at brukeren kan få kontrollere veipunktene med trykk fremfor stemme – dersom brukeren skulle foretrekke dette.



Figur E.7 Tablet-fanen.

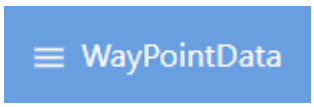
Vedlegg F: Hvordan sette opp systemet klart til bruk

Serveroppstart

Start Raspberry Pi ved å koble til strøm og vent til den har startet fullstendig

Logg på Wi-Fi og gå til http://192.168.136.48:1880/ui

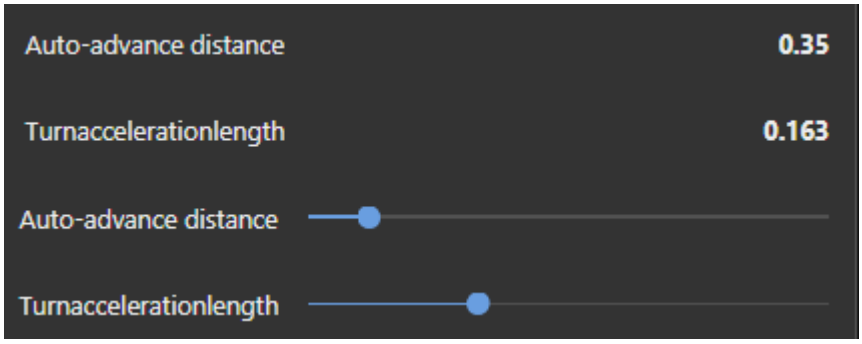
Gå inn i fanen:



Sett ønsket «*Auto-advance distance*».

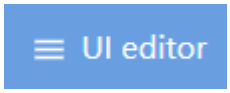
NB! Denne funksjonen er ikke ferdig implementert og er deaktivert.

Sett ønsket Turnaccelerationlength, dette er for å gi wheel-over point fremfor avstand til veipunkt



Trykk på

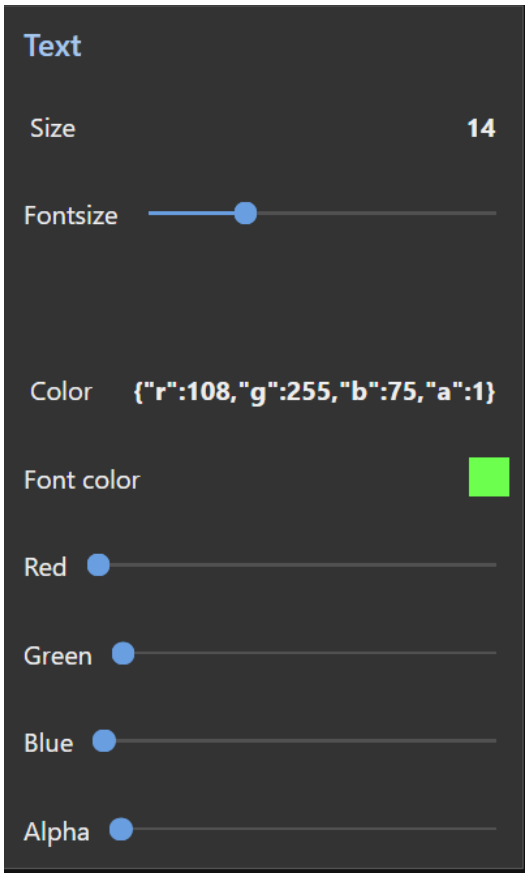
Gå inn i fanen

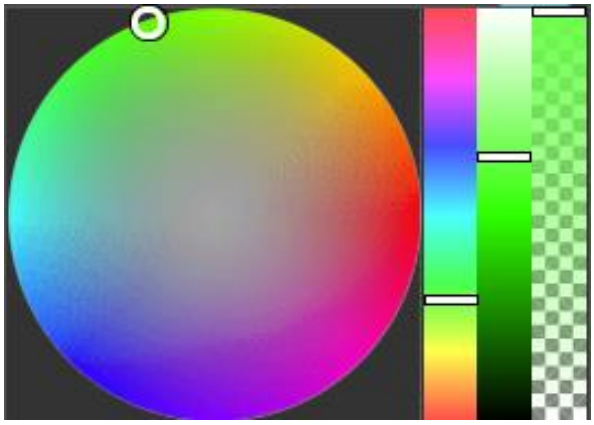


Sett «*Move by*» til 0.

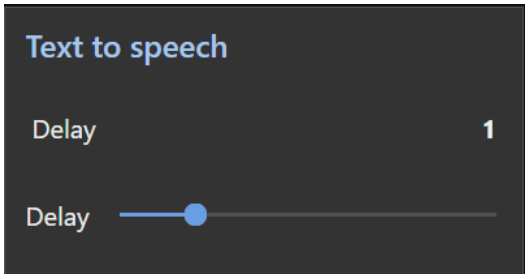
Sett ønsket størrelse (standardverdi: 14) og farge (standardverdi: grønn eller rød) på font

Farge bestemmes med enten RGBA-komponenter eller et fargehjul.





Sett ønsket forsinkelse fra varsel til meldingen blir lest opp (Standardverdi:1)



Vedlegg G: Variabelliste – Delt mellom server / klient

Variabel	Eksempelverdi	Enhet	Beskrivelse	Kilde
Time	191548.55	klokkeslett	Fartøyets tid	$GPZDA
Lat	60.33054	grader	Fartøyets breddegrad	$GPGGA
Lon	5.167593333333	grader	Fartøyets lengdegrad	$GPGGA
Knot	14.3	knop	Fartøyets fart	$GPVTG
Cog	36.7	grader	Fartøyets kurs over grunn	$GPVTG
TrueHeading	208.86	grader	Sann kurs	$HETHS
Rot (Rate of turn)	-38.4	grader/sek	Svinghastighet	$HEROT
Waypointid	5	heltall	Veipunktnummer	.rux
Waypointname	9Vsb>GISO4	streng	Kursnotasjon	.rux
Waypointlat	60.338417	grader	Veipunktbreddegrad	.rux
Waypointlon	5.169372	grader	Veipunktlengdegrad	.rux
Waypointturnradius	0.0500	nautisk mil	Oppgitt svingradius	.rux
Waypointlegtype	0	identifikator	Strekningstype	.rux
Waypointxte	0.0540	nautisk mil	Maks. avstand fra kurslinje	.rux
Waypointspeed	15.00	knop	Ønsket hastighet	.rux
Waypointdanger	1	identifikator	Farer på strekningen	.rux
Waypointmessage	TV/JH	streng	Notat	.rux
Waypointrtime	56.72	minutter	Gjenværende tid av seilas	.rux

Nextwaypointname	9Vsb>GISO4	streng	Neste kursnotasjon	.rux
Nextwaypointmessage	TV/JH	streng	Neste notat	.rux
Prevwaypointname	9Vsb>GISO4	streng	Forrige kursnotasjon	.rux
Distancetowaypoint	4.12	nautisk mil	Avstand til veipunkt	Regnet
Timetowaypoint	2m16	min / sek	Tid til veipunkt	Regnet
Timetonextwaypoint	2m16	min / sek	Tid til veipunkt	Regnet
Bearing	167	grader	Kurs som skal holdes	Regnet
Nextbearing	179	grader	Kurs som skal holdes	Regnet
Magheading	182	grader	Retning brillene peker	GPS-brikke
Voicewaypoint	True	boolsk	Bytter til neste veipunkt	HoloLens
Texttospeech	«*Turn starboard*»	streng	Leses opp av tekst til tale	Dashbord
Delay	1.1	sekunder	Forsinkelse fra varsel til mld.	Dashbord
Element	Currentwaypoint	elementvalg	Valgt element for flytt	Dashbord
Moveby	1.2	meter	Ønsket lengde ved flytt	Dashbord
Direction	Away	retning	Ønsket retning	Dashbord
Movenow	True	boolsk	Flytt elementet nå	NODE-red
Red	108	heltall	Mengde rød	Dashbord

Green	255	heltall	Mengde grønn	Dashbord
Blue	75	heltall	Mengde blå	Dashbord
Alpha	255	heltall	Gjennomsiktighet	Dashbord
Fontsize	14	heltall	Skriftstørrelse	Dashbord
Automaticwaypoint	True	boolsk	Ber HoloLens til å bytte VP.	NODE-red
Voiceassistance	False	boolsk	Angir stemmeassistanse	HoloLens
Xte	-0.5	nautisk mil	Distanse til kurslinje	Regnet

Vedlegg H: Ruteanvisningsfil

<?xml version="1.0" encoding="UTF-8"?>

<KM_Route xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.1" RtName="MPN42A" LastUpdate="20190503 18:36:19">

<WayPoints WpCount="53">

<WayPoint Id="1" WPName=">S" Lat="60.394278" Lon="5.262925" TurnRadius="0.0500" LegType="0" XTE="0.0756" Speed="18.00" Danger="-1" Message="MB" RTime="204.30"/>

<WayPoint Id="2" WPName="TP248FM2bb>MK" Lat="60.402111" Lon="5.241183" TurnRadius="0.0500" LegType="0" XTE="0.0972" Speed="18.00" Danger="-1" RTime="201.73"/>

<WayPoint Id="3" WPName="TP202FM1bb>VBK" Lat="60.381997" Lon="5.176100" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" RTime="194.07"/>

<WayPoint Id="4" WPName="TP143FM1bb>FM1" Lat="60.374878" Lon="5.165164" TurnRadius="0.0500" LegType="0" XTE="0.0540" Speed="18.00" Danger="-1" RTime="192.30"/>

<WayPoint Id="5" WPName="9Vsb>GISO4" Lat="60.338417" Lon="5.169372" TurnRadius="0.0500" LegType="0" XTE="0.0540" Speed="15.00" Danger="-1" RTime="185.00"/>

<WayPoint Id="6" WPName="9SKLsb>MK" Lat="60.333511" Lon="5.159378" TurnRadius="0.0500" LegType="0" XTE="0.0432" Speed="15.00" Danger="-1" RTime="183.28"/>

<WayPoint Id="7" WPName="[0,1]W/3>(0,15)FM" Lat="60.329708" Lon="5.144653" TurnRadius="0.0500" LegType="0" XTE="0.0918" Speed="18.00" Danger="-1" RTime="181.37"/>

<WayPoint Id="8" WPName="9FM2bb>FM1" Lat="60.302233" Lon="5.138814" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" RTime="175.78"/>

<WayPoint Id="9" WPName="9FM1bb>FG" Lat="60.261972" Lon="5.151622" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" RTime="167.58"/>

<WayPoint Id="10" WPName="9GISO6>S" Lat="60.241172" Lon="5.154428" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="163.42"/>

<WayPoint Id="11" WPName="9FM1sb>QR" Lat="60.203344" Lon="5.179236" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" Message="MB/AGB" RTime="157.43"/>

<WayPoint Id="12" WPName="45QGbb>MK" Lat="60.168514" Lon="5.184078" TurnRadius="0.0500" LegType="0" XTE="0.0497" Speed="15.00" Danger="-1" RTime="150.43"/>
<WayPoint Id="13" WPName="45QRsb>(0.05)FRsb" Lat="60.161589" Lon="5.187367" TurnRadius="0.0500" LegType="0" XTE="0.0243" Speed="12.00" Danger="-1" RTime="148.72"/>
<WayPoint Id="14" WPName="45FRsb>(0.1)QGbb" Lat="60.143142" Lon="5.187725" TurnRadius="0.0500" LegType="0" XTE="0.0243" Speed="12.00" Danger="-1" RTime="143.15"/>
<WayPoint Id="15" WPName="45FRbb>(0.1)FRbb" Lat="60.131544" Lon="5.188717" TurnRadius="0.0500" LegType="0" XTE="0.0497" Speed="12.00" Danger="-1" RTime="139.73"/>
<WayPoint Id="16" WPName="9FRbb>MK" Lat="60.125317" Lon="5.197783" TurnRadius="0.0500" LegType="0" XTE="0.0108" Speed="6.00" Danger="-1" RTime="137.47"/>
<WayPoint Id="17" WPName="GeWISO4bb>MIDT" Lat="60.125325" Lon="5.203567" TurnRadius="0.0500" LegType="0" XTE="0.0108" Speed="6.00" Danger="-1" RTime="135.70"/>
<WayPoint Id="18" WPName="[0.15]ISO4bb>???" Lat="60.129411" Lon="5.212936" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" RTime="132.07"/>
<WayPoint Id="19" WPName="" Lat="60.113164" Lon="5.350064" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" RTime="117.90"/>
<WayPoint Id="20" WPName="" Lat="60.103472" Lon="5.385186" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" RTime="113.87"/>
<WayPoint Id="21" WPName="[0.4]ØKLsb>(0.3)W" Lat="60.081425" Lon="5.478153" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" Message="AB/TV" RTime="103.70"/>
<WayPoint Id="22" WPName="" Lat="60.016414" Lon="5.449339" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="93.65"/>
<WayPoint Id="23" WPName="" Lat="60.009525" Lon="5.433792" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="92.05"/>
<WayPoint Id="24" WPName="" Lat="60.006047" Lon="5.427639" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="91.38"/>
<WayPoint Id="25" WPName="" Lat="60.004178" Lon="5.414800" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="90.35"/>
<WayPoint Id="26" WPName="" Lat="60.003106" Lon="5.404503" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="89.57"/>
<WayPoint Id="27" WPName="" Lat="60.002472" Lon="5.390328" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="88.53"/>

<WayPoint Id="28" WPName="" Lat="60.000081" Lon="5.385683" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="87.98"/>
<WayPoint Id="29" WPName="" Lat="59.998778" Lon="5.383744" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="87.77"/>
<WayPoint Id="30" WPName="" Lat="59.997372" Lon="5.383208" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="87.57"/>
<WayPoint Id="31" WPName="" Lat="59.995233" Lon="5.378931" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="87.08"/>
<WayPoint Id="32" WPName="" Lat="59.992861" Lon="5.376258" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="86.67"/>
<WayPoint Id="33" WPName="" Lat="59.990922" Lon="5.374253" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="86.37"/>
<WayPoint Id="34" WPName="" Lat="59.987108" Lon="5.363758" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="85.40"/>
<WayPoint Id="35" WPName="" Lat="59.982664" Lon="5.267269" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="78.12"/>
<WayPoint Id="36" WPName="" Lat="60.002011" Lon="5.219233" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="73.43"/>
<WayPoint Id="37" WPName="" Lat="60.015658" Lon="5.200686" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="70.93"/>
<WayPoint Id="38" WPName="" Lat="60.026014" Lon="5.186178" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="69.03"/>
<WayPoint Id="39" WPName="" Lat="60.050744" Lon="5.153594" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="64.58"/>
<WayPoint Id="40" WPName="" Lat="60.079267" Lon="5.125308" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="59.83"/>
<WayPoint Id="41" WPName="" Lat="60.087286" Lon="5.127719" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1" RTime="58.60"/>
<WayPoint Id="42" WPName="TP355G/3>MK¨" Lat="60.099189" Lon="5.120231" TurnRadius="0.0500" LegType="0" XTE="0.0540" Speed="15.00" Danger="-1" Message="TV/JH" RTime="56.72"/>
<WayPoint Id="43" WPName="9G/3>MK" Lat="60.121850" Lon="5.114261" TurnRadius="0.0500" LegType="0" XTE="0.0540" Speed="15.00" Danger="-1" RTime="51.20"/>
<WayPoint Id="44" WPName="TP340MUB>MUB" Lat="60.127631" Lon="5.114253" TurnRadius="0.0500" LegType="0" XTE="0.0540" Speed="15.00" Danger="-1" RTime="49.83"/>
<WayPoint Id="45" WPName="[0.54]B>(0.2)FM" Lat="60.142214" Lon="5.103208" TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="18.00" Danger="-1" RTime="46.15"/>

```
        <WayPoint Id="46" WPName="9FMbb&gt;(0.1)GISO6s" Lat="60.200619" Lon="5.176331"
        TurnRadius="0.0500" LegType="0" XTE="0.0918" Speed="18.00" Danger="-1"
        RTime="32.35"/>
        <WayPoint Id="47" WPName="9GISO6sb&gt;(0.1)FRb" Lat="60.241961" Lon="5.160203"
        TurnRadius="0.0500" LegType="0" XTE="0.0999" Speed="24.00" Danger="-1"
        RTime="23.88"/>
        <WayPoint Id="48" WPName="9FRbb&gt;FM3" Lat="60.263681" Lon="5.179050"
        TurnRadius="0.0500" LegType="0" XTE="0.0810" Speed="24.00" Danger="-1"
        RTime="20.30"/>
        <WayPoint Id="49" WPName="TP035FG&gt;RAWL" Lat="60.286178" Lon="5.194861"
        TurnRadius="0.0500" LegType="0" XTE="0.0702" Speed="24.00" Danger="-1"
        RTime="16.72"/>
        <WayPoint Id="50" WPName="WeGFM2&gt;FM2" Lat="60.319806" Lon="5.205492"
        TurnRadius="0.0500" LegType="0" XTE="0.0540" Speed="24.00" Danger="-1"
        RTime="11.67"/>
        <WayPoint Id="51" WPName="WeRFM1&gt;FM1rWg" Lat="60.324642" Lon="5.232217"
        TurnRadius="0.0500" LegType="0" XTE="0.0432" Speed="8.00" Danger="-1"
        RTime="9.55"/>
        <WayPoint Id="52" WPName="9Brbb&gt;FR" Lat="60.339592" Lon="5.238525"
        TurnRadius="0.0500" LegType="0" XTE="0.0324" Speed="8.00" Danger="-1"
        RTime="2.60"/>
        <WayPoint Id="53" WPName="" Lat="60.344847" Lon="5.234908" TurnRadius="0.0500"
        LegType="0" XTE="0.0324" Speed="8.00" Danger="-1" RTime="0.00"/>
    </WayPoints>
</KM_Route>
```

Vedlegg I: MoveElements C# script

```
1.  // 13.11.2019 Sjøkrigsskolen
2.  // Bachelor, Augmented Reality
3.  // Written by Christer Algrøy, Julian Tobias Venstad and Markus Øvstedal
4.
5.  // Questions: julian.venstad@gmail.com
6.
7.
8.  using System.Collections;
9.  using System.Collections.Generic;
10. using UnityEngine;
11.
12.
13.
14. public class MoveElement : MonoBehaviour
15. {
16.     public MyTcpClient MyTcpClient;
17.
18.     // Start is called before the first frame update
19.     void Start()
20.     {
21.         // Calls the function "MoveIt" every 2 seconds after 2.2 seconds.
22.         InvokeRepeating("MoveIt", 2.2f, 2f);
23.     }
24.
25.     // Tick the variable in Unity to match the element
26.     public bool cog;
27.     public bool currentWaypoint;
28.     public bool nextWaypoint;
29.
30.     private bool moved = false;
31.
32.     // Moves element if it is asked and allowed to move for a given distance in a g
    iven direction
33.    void  MoveIt()
34.     {
35.         // Get variable
36.         string Movenow = MyTcpClient.giveMe("Movenow");
37.         if (Movenow != "True")
38.             moved = false;
39.
40.         // If the variable is set to "True" the function will continue
41.         if (Movenow == "True" && !moved)
42.         {
43.             //Checks which element should move
44.             bool Match = false;
45.             string Element = MyTcpClient.giveMe("Element");
46.             if ((Element == "Cog" && cog) || (Element == "Currentwaypoint" && curre
    ntWaypoint) ||
47.                 (Element == "Nextwaypoint" && nextWaypoint) || (Element=="Allelemen
    ts"))
48.                 Match = true;
49.
50.
51.             if (Match)
52.             {
53.                 // Writes to console in Visual Studio
54.                 Debug.Log("Starting to move object");
55.
56.                 // Moveby is how far
57.                 string Movebystring = MyTcpClient.giveMe("Moveby");
58.                 float Moveby = float.Parse(Movebystring);
```

```
59.
60.                 // Direction is one of six directions (think of a dice)
61.                 string Direction = MyTcpClient.giveMe("Direction");
62.
63.                 // Defines which component we wish to move (in this case it's the o
    bject which the script is attached to)
64.                 RectTransform myRectTransform = GetComponent<RectTransform>();
65.
66.                 // Towards user
67.                 if (Direction == "Towards")
68.                     myRectTransform.localPosition += Vector3.up * Moveby;
69.
70.                 // Away from user
71.                 else if (Direction == "Away")
72.                     myRectTransform.localPosition += Vector3.down * Moveby;
73.
74.                 // down
75.                 else if (Direction == "Down")
76.                     myRectTransform.localPosition += Vector3.left * Moveby;
77.
78.                 // up
79.                 else if (Direction == "Up")
80.                     myRectTransform.localPosition += Vector3.right * Moveby;
81.
82.                 // right
83.                 else if (Direction == "Right")
84.                     myRectTransform.localPosition += Vector3.back * Moveby;
85.
86.                 // left
87.                 else if (Direction == "Left")
88.                     myRectTransform.localPosition += Vector3.forward * Moveby;
89.
90.                 // Sets the variable Movenow to "False", this stops it from looping
    .
91.                 // There are some delays with setting the variables on the Raspberr
    yPi-server (NODE-RED)
92.                 // Therefore we check if the element has been moved already.
93.                 // In the current context it does not matter due to the low refresh
    rate of 0.5Hz.
94.
95.                 moved = true;
96.                 MyTcpClient.forceSend("Movenow:False");
97.             }
98.         }
99.     }
100.        }
```

Vedlegg J: MyTcpClient C# script

```
1.  // 13.11.2019 Sjøkrigsskolen
2.  // Bachelor, Augmented Reality
3.  // Written by Christer Algrøy, Julian Tobias Venstad and Markus Øvstedal
4.
5.  // Questions: julian.venstad@gmail.com
6.
7.
8.  // THIS CODE WILL ONLY WORK FOR UNIVERSAL WINDOWS PLATFORM (UWP)
9.  // THIS CODE IS ONLY TESTED ON FIRST GENERATION HOLOLENS
10. // We need to use #if "!UNITY_EDITOR  /  #endif" to get certain parts of the code t
    hrough Unity
11. // Unity will complain about not being able to compile due to missing namespaces an
    d functions
12.
13.
14. // With help from: https://medium.com/datadriveninvestor/connecting-the-microsoft-
    hololens-and-raspberry-pi3b-58665032964c
15. using System;
16. using System.Collections.Generic;
17. using System.IO;
18. using System.Text;
19. using System.Threading;
20. using UnityEngine;
21. using UnityEngine.Windows.Speech;
22.
23.
24. #if !UNITY_EDITOR
25. using System.Threading.Tasks;
26. #endif
27.
28. public class MyTcpClient : MonoBehaviour
29. {
30. #if !UNITY_EDITOR
31.     private bool _useUWP = true;
32.     private Windows.Networking.Sockets.StreamSocket socket;
33.     private Task exchangeTask;
34. #endif
35.
36. #if UNITY_EDITOR
37.     private bool _useUWP = false;
38.     System.Net.Sockets.TcpClient client;
39.     System.Net.Sockets.NetworkStream stream;
40.     private Thread exchangeThread;
41. #endif
42.
43.     private Byte[] bytes = new Byte[256];
44.     private StreamWriter writer;
45.     private StreamReader reader;
46.
47.     public async void Start()
48.     {
49.         // Server ip address and port of the server
50.         // In our context that is the Raspberry-Pi running Node-RED
51.         Connect("192.168.136.48", "1025");
52. #if !UNITY_EDITOR
53.         // Delay for Hololens to connect properly before starting to exchange packe
    ts
54.         Debug.Log("Waiting 2 seconds ...");
55.         await Task.Delay(TimeSpan.FromSeconds(2));
56.
57. #endif
```

```
58.
59.         // Calls the function "ExchangePackets" after a two second delay @10Hz
60.         InvokeRepeating("ExchangePackets", 2.0f, 0.1f);
61.         Debug.Log("TCP startup completed!");
62.     }
63.
64.     public async void Connect(string host, string port)
65.     {
66.         if (_useUWP)
67.         {
68.             ConnectUWP(host, port);
69.         }
70.         else
71.         {
72.             ConnectUnity(host, port);
73.         }
74.     }
75.
76. #if UNITY_EDITOR
77.     private void ConnectUWP(string host, string port)
78. #else
79.     private async void ConnectUWP(string host, string port)
80. #endif
81.     {
82. #if UNITY_EDITOR
83.         errorStatus = "UWP TCP client used in Unity!";
84. #else
85.         try
86.         {
87.             if (exchangeTask != null) StopExchange();
88.             successStatus = "Not connected yet!";
89.             Debug.Log(successStatus);
90.
91.             socket = new Windows.Networking.Sockets.StreamSocket();
92.             Windows.Networking.HostName serverHost = new Windows.Networking.HostNam
    e(host);
93.             await socket.ConnectAsync(serverHost, port);
94.
95.             Stream streamOut = socket.OutputStream.AsStreamForWrite();
96.             writer = new StreamWriter(streamOut) { AutoFlush = true };
97.
98.             Stream streamIn = socket.InputStream.AsStreamForRead();
99.             reader = new StreamReader(streamIn);
100.
101.                 successStatus = "Connected!";
102.                 Debug.Log(successStatus);
103.             }
104.             catch (Exception e)
105.             {
106.                 errorStatus = e.ToString();
107.             }
108.     #endif
109.         }
110.
111.         private void ConnectUnity(string host, string port)
112.         {
113.     #if !UNITY_EDITOR
114.             errorStatus = "Unity TCP client used in UWP!";
115.     #else
116.             try
117.             {
118.                 if (exchangeThread != null) StopExchange();
119.
120.                 client = new System.Net.Sockets.TcpClient(host, Int32.Parse(port
    ));
121.                 stream = client.GetStream();
```

```
122.                     reader = new StreamReader(stream);
123.                     writer = new StreamWriter(stream) { AutoFlush = true };
124.
125.                     successStatus = "Connected!";
126.                 }
127.                 catch (Exception e)
128.                 {
129.                     errorStatus = e.ToString();
130.                 }
131.         #endif
132.             }
133.
134.             private bool exchanging = false;
135.             private bool exchangeStopRequested = false;
136.             private string lastPacket = null;
137.
138.             private string errorStatus = null;
139.             private string successStatus = null;
140.
141.             // Returns the value of a given variable
142.             public string giveMe(string giveMe)
143.             {
144.                 int i = findIndex(AvailableVariables, giveMe);
145.                 if (i == -1) {
146.                     Debug.Log("Could not find value for " + giveMe);
147.                     return ("Could not find value for " + giveMe);
148.                 }
149.                 else
150.                 return UpdatedValues[i];
151.             }
152.
153.             public string PreviousResponse;
154.
155.             // The available variables to read / write
156.             // The two lists neeed to have identical indexes for variables
157.             public List<string> AvailableVariables = new List<string>(new string[]

158.             {"Time", "Lat", "Lon", "Knot", "Cog", "TrueHeading", "Rot", "Waypointid"
    , "Waypointname", "Waypointlat", "Waypointlon",
159.                 "Waypointturnradius", "Waypointlegtype", "Waypointxte", "Waypointspe
    ed", "Waypointdanger", "Waypointmessage", "Waypointrtime", "Bearing",
160.                 "Nextbearing", "Nextwaypointname", "Nextwaypointmessage", "Distancet
    owaypoint", "Distancetonextwaypoint", "Timetowaypoint","Timetonextwaypoint",
161.                 "Voicewaypoint","Element", "Moveby", "Direction", "Movenow", "Maghea
    ding", "Red", "Green", "Blue", "Alpha", "Texttospeech", "Fontsize", "Delay",
162.                 "Automaticwaypoint", "Voiceassistance", "Xte", "Prevwaypointname"  }
    );
163.
164.             // The values will be written to this list
165.             public List<string> UpdatedValues = new List<string>(new string[]
166.             {"Time", "Lat", "Lon", "Knot", "Cog", "TrueHeading", "Rot", "Waypointid"
    , "Waypointname", "Waypointlat", "Waypointlon",
167.                 "Waypointturnradius", "Waypointlegtype", "Waypointxte", "Waypointspe
    ed", "Waypointdanger", "Waypointmessage", "Waypointrtime", "Bearing",
168.                 "Nextbearing", "Nextwaypointname", "Nextwaypointmessage", "Distancet
    owaypoint", "Distancetonextwaypoint", "Timetowaypoint","Timetonextwaypoint",
169.                 "Voicewaypoint", "Element", "Moveby", "Direction", "Movenow", "Maghe
    ading", "Red", "Green", "Blue", "Alpha", "Texttospeech", "Fontsize", "Delay",
170.                 "Automaticwaypoint", "Voiceassistance", "Xte", "Prevwaypointname"  }
    );
171.
172.             public async void ExchangePackets()
173.             {
174.
175.                 try
```

```
176.                {
177.                    // Sends "GiveMeUpdate" to the server
178.                    writer.Write("GiveMeUpdate");
179.
180.                    // Writes the recieved message to PreviousResponse
181.                    PreviousResponse = await reader.ReadLineAsync();
182.
183.                    //Writes response to console in Visual Studio
184.                    Debug.Log(PreviousResponse);
185.
186.                    //Splits the string twice and updates the variables with their n
    ew values
187.                    string[] elements = PreviousResponse.Split('|');
188.                    int index = 0;
189.                    foreach (var elem in elements)
190.                    {
191.                        if (elem == "")
192.                            continue;
193.                        string[] split = elem.Split(':');
194.                        index = findIndex(AvailableVariables, split[0]);
195.                        UpdatedValues[index] = split[1];
196.                        index++;
197.                    }
198.                }
199.
200.                catch (Exception e)
201.                {
202.                    Debug.Log(e.ToString());
203.                }
204.            }
205.
206.            // Returns index of a given keyword
207.            int findIndex(List<string> myList, string myString)
208.            {
209.                for (int i = 0; i < myList.Count; i++)
210.                {
211.                    if (myList[i] == myString)
212.                    {
213.                        return i;
214.                    }
215.                }
216.                return -1;
217.            }
218.
219.            // Sends a given string to the server. This is used to set (/change) var
    iables on the server
220.            public async void forceSend(string input)
221.            {
222.                writer.Write(input);
223.                Debug.Log("forceSend: " + input);
224.                PreviousResponse = await reader.ReadLineAsync();
225.            }
226.
227.
228.            public void StopExchange()
229.            {
230.                exchangeStopRequested = true;
231.
232.        #if UNITY_EDITOR
233.                if (exchangeThread != null)
234.                {
235.                    exchangeThread.Abort();
236.                    stream.Close();
237.                    client.Close();
238.                    writer.Close();
239.                    reader.Close();
```

```
240.
241.                     stream = null;
242.                     exchangeThread = null;
243.                 }
244.         #else
245.                 if (exchangeTask != null)
246.                 {
247.                     exchangeTask.Wait();
248.                     socket.Dispose();
249.                     writer.Dispose();
250.                     reader.Dispose();
251.
252.                     socket = null;
253.                     exchangeTask = null;
254.                 }
255.         #endif
256.                 writer = null;
257.                 reader = null;
258.             }
259.
260.             public void OnDestroy()
261.             {
262.                 StopExchange();
263.             }
264.         }
```

Vedlegg K: VoiceCommand C#script

```
1.  // 13.11.2019 Sjøkrigsskolen
2.  // Bachelor for Augmented Reality
3.  // Written by Christer Algrøy, Julian Tobias Venstad and Markus Øvstedal
4.
5.  // Questions: julian.venstad@gmail.com
6.
7.
8.  // With help from https://docs.microsoft.com/nb-no/azure/cognitive-services/speech-
    service/get-started
9.
10.
11. using System;
12. using System.Collections;
13. using System.Collections.Generic;
14. using System.Linq;
15. using UnityEngine;
16. using UnityEngine.Windows.Speech;
17. using UnityEngine.XR.WSA.WebCam;
18. #if !UNITY_EDITOR
19. using System.Threading.Tasks;
20. #endif
21.
22.
23. public class VoiceCommand : MonoBehaviour
24. {
25.
26.     // Voice command vars
27.     private Dictionary<string, Action> keyActs = new Dictionary<string, Action>();

28.     private KeywordRecognizer recognizer;
29.
30.     public MyTcpClient MyTcpClient;
31.     public PlaySound PlaySound;
32.     public TextToSpeech TextToSpeech;
33.
34.     // Start is called before the first frame update
35.     void Start()
36.     {
37.         Debug.Log("Starting voicecommands!");
38.
39.         //Avaiable voicecommands
40.         keyActs.Add("enable voice", EnableVoice);
41.         keyActs.Add("disable voice", DisableVoice);
42.         keyActs.Add("next waypoint", NextWaypoint);
43.         keyActs.Add("previous waypoint", PreviousWaypoint);
44.         keyActs.Add("thank you", ThankYou);
45.         keyActs.Add("toggle markers", Toggle45);
46.         keyActs.Add("play video", PlayVideo);
47.         keyActs.Add("stop video", StopVideo);
48.
49.         //Starts the voicerecognizer
50.         recognizer = new KeywordRecognizer(keyActs.Keys.ToArray());
51.         recognizer.OnPhraseRecognized += OnKeywordsRecognized;
52.         recognizer.Start();
53.         Debug.Log("Voicecommands started!");
54.
55.     }
56.
57.
58.     void OnKeywordsRecognized(PhraseRecognizedEventArgs args)
59.     {
60.         //Recognizes keywords, writes to console and calls given function
```

```
61.         Debug.Log("Command: " + args.text);
62.         keyActs[args.text].Invoke();
63.     }
64.
65.     //This function is mostly for fun, and gives one of six random responses
66.     void ThankYou()
67.     {
68.
69.         System.Random rnd = new System.Random();
70.         int random = rnd.Next(5);
71.         if (random == 0)
72.             TextToSpeech.Say("My pleasure");
73.         else if (random == 1)
74.             TextToSpeech.Say("You're welcome");
75.         else if (random == 2)
76.             TextToSpeech.Say("No problem, darling");
77.         else if (random == 3)
78.             TextToSpeech.Say("Do it yourself next time");
79.         else if (random == 4)
80.             TextToSpeech.Say("I'm happy to help");
81.         else if (random == 5)
82.             TextToSpeech.Say("It was the least I could do");
83.     }
84.
85.     //Proof of concept of toggleing a variable, in this context it's an alarm which
    is set true or false
86.     void EnableVoice()
87.     {
88.         MyTcpClient.forceSend("Voiceassistance:True");
89.         TextToSpeech.Say("Voiceassistance enabled");
90.     }
91.
92.     void DisableVoice()
93.     {
94.         MyTcpClient.forceSend("Voiceassistance:False");
95.         TextToSpeech.Say("Voiceassistance disabled");
96.
97.     }
98.
99.     //Makes the user able to switch to next and previous
100.         void NextWaypoint()
101.         {
102.             MyTcpClient.forceSend("Voicewaypoint:Next");
103.             int nextWaypoint = int.Parse(MyTcpClient.giveMe("Waypointid")) + 1;

104.             TextToSpeech.Say("Waypoint number " + nextWaypoint);
105.         }
106.
107.         void PreviousWaypoint()
108.         {
109.             MyTcpClient.forceSend("Voicewaypoint:Previous");
110.             int nextWaypoint = int.Parse(MyTcpClient.giveMe("Waypointid")) - 1;

111.             TextToSpeech.Say("Waypoint number " + nextWaypoint);
112.         }
113.
114.
115.         private bool isVisible = true;
116.         private string status;
117.
118.         //Makes user able to toggle 45°- and 90°-degree markers
119.         void Toggle45()
120.         {
121.             Renderer Red45 = GameObject.Find("Red45").GetComponent<Renderer>();
    ;
```

```
122.                Renderer Green45 = GameObject.Find("Green45").GetComponent<Renderer>
    ();
123.                Renderer Red90 = GameObject.Find("Red90").GetComponent<Renderer>();
    ;
124.                Renderer Green90 = GameObject.Find("Green90").GetComponent<Renderer>
    ();
125.
126.                if (isVisible)
127.                    isVisible = false;
128.                else isVisible = true;
129.                Debug.Log(isVisible);
130.
131.                Red45.enabled = isVisible;
132.                Green45.enabled = isVisible;
133.                Red90.enabled = isVisible;
134.                Green90.enabled = isVisible;
135.                if (isVisible)
136.                    status = "on";
137.                else status = "off";
138.
139.                TextToSpeech.Say("Markers toggled " + status);
140.            }
141.
142.            //Makes user able to play and stop video
143.            void PlayVideo()
144.            {
145.                GameObject plane = GameObject.Find("Plane");
146.                Renderer planerend = plane.GetComponent<Renderer>();
147.                planerend.enabled = true;
148.                var videoPlayer = plane.AddComponent<UnityEngine.Video.VideoPlayer>(
    );
149.                videoPlayer.url = "http://clips.vorwaerts-
    gmbh.de/big_buck_bunny.mp4";
150.                videoPlayer.Play();
151.                TextToSpeech.Say("Video playing");
152.            }
153.
154.            void StopVideo()
155.            {
156.                GameObject plane = GameObject.Find("Plane");
157.                Renderer planerend = plane.GetComponent<Renderer>();
158.                planerend.enabled = false;
159.                var videoPlayer = plane.GetComponent<UnityEngine.Video.VideoPlayer>(
    );
160.                Destroy(videoPlayer);
161.            }
162.        }
```

Vedlegg L: UpdateText C# script

```
// 13.11.2019 Sjøkrigsskolen
// Bachelor, Augmented Reality
// Written by Christer Algrøy, Julian Tobias Venstad and Markus Øvstedal

// Questions: julian.venstad@gmail.com


using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
using UnityEngine.UI;
using System.Threading.Tasks;

public class UpdateText : MonoBehaviour
{

    public MyTcpClient MyTcpClient;
    public TextToSpeech TextToSpeech;


    // Start is called before the first frame update
    void Start()
    {
        InvokeRepeating("WriteText", 3.0f, 0.5f);
    }


    // Tick the correct element in Unity
    public bool currentPosition;
    public bool currentWaypoint;
    public bool nextWaypoint;
    public bool speedCog;
    public bool tts;
    public bool statictext;
    public bool realtimetext;


    // Returns true if under treshhold, false is over
    bool checktime()
    {
        // Example time = "14m36"
        string time = MyTcpClient.giveMe("Timetowaypoint");
        string[] minutesandseconds = time.Split('m');
        // [0] = 14 and [1] = 36
        currentminutes = int.Parse(minutesandseconds[0]);
        currentseconds = int.Parse(minutesandseconds[1]);

        // Check if remaining time is under the given treshhold.
        int totalseconds = currentminutes * 60 + currentseconds;
        if (totalseconds <= 180)
            return true;
        else
            return false;
    }

    private string lastwaypointstring = "";
    private string textToScreen;

    private bool startup = true;
```

```
62.     private bool message = false;
63.     private int currentminutes;
64.     private int currentseconds;
65.     private string newnextbearing = "";
66.
67.     private int nextminutes;
68.     private int nextseconds;
69.
70.     // Formats the information to display on screen to the variable 'textToScreen'
    and displays it
71.     async void WriteText()
72.     {
73.         // Defines the textbox of the object which the script is attached to
74.         TextMeshPro mText = gameObject.GetComponent<TextMeshPro>();
75.
76.         // RGBA stands for red green blue alpha.
77.         // While it is sometimes described as a color space,
78.         // it is actually the three-
    channel RGB color model supplemented with a 4th alpha channel.
79.         byte red, green, blue, alpha;
80.
81.         // Get the value of each component in the range [0-255]
82.         red = byte.Parse(MyTcpClient.giveMe("Red"));
83.         green = byte.Parse(MyTcpClient.giveMe("Green"));
84.         blue = byte.Parse(MyTcpClient.giveMe("Blue"));
85.         alpha = byte.Parse(MyTcpClient.giveMe("Alpha"));
86.
87.         // Sets the color and font size of the text
88.         mText.faceColor = new Color32(red, green, blue, alpha);
89.         mText.fontSize = float.Parse(MyTcpClient.giveMe("Fontsize"));
90.
91.         if (currentPosition)
92.             // Formats text to display
93.             textToScreen = "Current position Lat: " + MyTcpClient.giveMe("Lat") + "
     // Lon: " + MyTcpClient.giveMe("Lon");
94.
95.         else if (currentWaypoint)
96.         {
97.             string currentwaypointstring = MyTcpClient.giveMe("Prevwaypointname");

98.             // Fixes symbols from TCP-communication
99.             // In the TCP-
    communication some symbols gets interpeted wrong and needs to be corrected.
100.                    // These are the symbols that we found needed to be replaced
101.                    currentwaypointstring = currentwaypointstring.Replace("&gt;", ">
    ");
102.                    currentwaypointstring = currentwaypointstring.Replace("&lt;", "<
    ");
103.                    currentwaypointstring = currentwaypointstring.Replace("/", "/");

104.
105.                    // Reads the string after '>' or '<'
106.                    if (currentwaypointstring.Contains(">"))
107.                        currentwaypointstring = ">" + currentwaypointstring.Substrin
    g(currentwaypointstring.LastIndexOf('>') + 1);
108.                    else if (currentwaypointstring.Contains("<"))
109.                        currentwaypointstring = "<" + currentwaypointstring.Substrin
    g(currentwaypointstring.LastIndexOf('<') + 1);
110.                    // Formats text to display
111.                    textToScreen = currentwaypointstring + "\n" +
112.                    MyTcpClient.giveMe("Bearing") + "°\n" +
113.                    MyTcpClient.giveMe("Distancetowaypoint") + "nm\n";
114.                }
115.                else if (speedCog)
116.                {
117.                    // Formats text to display
```

```
118.                     textToScreen = MyTcpClient.giveMe("Knot") + "kt" + "     " + MyT
    cpClient.giveMe("Cog") + "°";
119.                 }
120.
121.
122.                 else if (nextWaypoint)
123.                 {
124.
125.                     string nextwaypointstring = MyTcpClient.giveMe("Waypointname");

126.                     // Fixes symbols from TCP-communication
127.                     nextwaypointstring = nextwaypointstring.Replace("&gt;", ">");
128.                     nextwaypointstring = nextwaypointstring.Replace("&lt;", "<");
129.                     nextwaypointstring = nextwaypointstring.Replace("/", "/");
130.                     string nextwaypointstring2 = nextwaypointstring;
131.
132.                     // Checks if it's a new waypoint or "just the same as last time"

133.                     if (lastwaypointstring != nextwaypointstring)
134.                     {
135.                         message = true;
136.                         lastwaypointstring = nextwaypointstring;
137.
138.
139.                         // Example time = "14m36"
140.                         string nexttime = MyTcpClient.giveMe("Timetonextwaypoint");

141.                         string[] nextminutesandseconds = nexttime.Split('m');
142.                         nextminutes = int.Parse(nextminutesandseconds[0]);
143.                         nextseconds = int.Parse(nextminutesandseconds[1]);
144.
145.                         string nextbearing = MyTcpClient.giveMe("Nextbearing");
146.                         // Adds a space between the numbers of the bearing for later
     use in TextToSpeech(TTS)
147.                         // Instead of saying "Onehundred and thirtyfour" it should s
    ay "one three four"
148.                         foreach (char c in nextbearing)
149.                         {
150.                             newnextbearing += c;
151.                             newnextbearing += ' ';
152.                         }
153.
154.
155.             }
156.                         if (nextwaypointstring.Contains(">"))
157.                             nextwaypointstring = nextwaypointstring.Substring(0, nex
    twaypointstring.IndexOf('>'));
158.                         else if (nextwaypointstring.Contains("<"))
159.                             nextwaypointstring = nextwaypointstring.Substring(0, nex
    twaypointstring.IndexOf('<'));
160.
161.                         if (nextwaypointstring2.Contains(">"))
162.                             nextwaypointstring2 = ">" + nextwaypointstring2.Substrin
    g(nextwaypointstring2.LastIndexOf('>') + 1);
163.                         else if (nextwaypointstring.Contains("<"))
164.                             nextwaypointstring2 = "<" + nextwaypointstring2.Substrin
    g(nextwaypointstring2.LastIndexOf('<') + 1);
165.
166.                         // We have an edge case where the vessel is standing still,
    and therefore has no speed
167.                         // This messes with some calculations and especially parsing
     of the variables to numbers
168.                         // This is solved by having the server give the time "99m99"
     when the time is in reality "infinite"
```

```
169.                     // Having 99 seconds should be impossible and only achievabl
    e with it being forced like this
170.                 if (currentminutes == 99 && currentseconds == 99)
171.                     // Formats text to display
172.                     textToScreen = "NO SPEED";
173.                 else
174.                 textToScreen = currentminutes +"m"+currentseconds+"s" + "\n";
175.                 if (checktime())
176.                 {
177.                     // Formats text to display
178.                     textToScreen = textToScreen + nextwaypointstring + "\n" +
179.                         MyTcpClient.giveMe("Nextbearing") + "°\n" +
180.                         MyTcpClient.giveMe("Distancetonextwaypoint") + "nm\n" +

181.                         nextwaypointstring2 + "\n" +
182.                         nextminutes + "m" + nextseconds + "s";
183.                 }
184.             }
185.
186.             if (tts)
187.             {
188.                 // Formats text to display
189.                 textToScreen = "Latest message: " + "\n" +
190.                     MyTcpClient.giveMe("Texttospeech");
191.             }
192.
193.             if (statictext)
194.             {
195.                 // Formats text to display
196.                 textToScreen = "Callsign: 1D4E" + "\n" +
197.                     "Draft: 3.2m" + "\n" +
198.                     "Length: 14.2m" + "\n" +
199.                     "Max speed: 30kt";
200.             }
201.
202.             if (realtimetext)
203.             {
204.                 // Formats text to display
205.                 textToScreen = "Load: 78 % // 15237rpm" + "\n" +
206.                     "Fuel 53 % // 14h12m // 14203m²" + "\n" +
207.                     "XTE: " +MyTcpClient.giveMe("Xte") + "nm"+"\n" +
208.                     "Voiceassistance:" + MyTcpClient.giveMe("Voiceassistance");

209.             }
210.
211.             mText.text = textToScreen;
212.
213.             // After a new waypoint and it is not in startup, it will give a "na
    vigation system" like message with TTS.
214.             // Example: Turn in 2 minutes, next course is 1 3 4 degrees, for 4.2
     nautical miles
215.             if (!startup && message && MyTcpClient.giveMe("Voiceassistance") ==
    "True")
216.             {
217.                 if (currentminutes == 99 && currentseconds == 99)
218.                     TextToSpeech.Say("Next course is " + newnextbearing + "degre
    es, for " + MyTcpClient.giveMe("Distancetonextwaypoint") +
219.                         " nautical miles, ");
220.                 else if (currentminutes == 0)
221.                     TextToSpeech.Say("Next course is " + newnextbearing + "degre
    es, for " + MyTcpClient.giveMe("Distancetonextwaypoint") +
222.                         " nautical miles, " + "it will take approximately " + ne
    xtminutes + " minutes");
223.                 else
224.                     TextToSpeech.Say("Turn in " + (currentminutes) + " minutes,
    next course is " + newnextbearing + "degrees, for " +
```

```
225.                         MyTcpClient.giveMe("Distancetonextwaypoint") + " nautica
    l miles, " + "it will take approximately " + nextminutes + " minutes");
226.
227.                     // "Deletes" the bearing and prepares the string for a new value
     to be added
228.                     // If you delete this the courses will be appended after eachoth
    er
229.                     newnextbearing = "";
230.                     message = false;
231.                 }
232.                     startup = false;
233.             }
234.         }
```

Vedlegg M: TextToSpeech C# script

```
1.  // 13.11.2019 Sjøkrigsskolen
2.  // Bachelor, Augmented Reality
3.  // Written by Christer Algrøy, Julian Tobias Venstad and Markus Øvstedal
4.
5.  // Questions: julian.venstad@gmail.com
6.
7.  // NOTE: REMEMBER TO UPDATE SUBSCRIPTIONKEY
8.  // Renew subscription at https://azure.microsoft.com/en-us/
9.  // Current subscription will expire at 13.12.2019
10.
11. // With help from:https://docs.microsoft.com/en-us/azure/cognitive-services/speech-
    service/quickstart-text-to-speech-csharp-unity
12.
13. using UnityEngine;
14. using UnityEngine.UI;
15. using Microsoft.CognitiveServices.Speech;
16. using System;
17. using System.Collections;
18. using System.Collections.Generic;
19. using System.Linq;
20. using UnityEngine.Windows.Speech;
21. using System.Threading.Tasks;
22.
23. public class TextToSpeech : MonoBehaviour
24. {
25.     public MyTcpClient MyTcpClient;
26.     public AudioSource audioSource;
27.
28.     // Vars for TextToSpeech(TTS)
29.     private object threadLocker = new object();
30.     private bool waitingForSpeak;
31.     private string message;
32.
33.     private SpeechConfig speechConfig;
34.     private SpeechSynthesizer synthesizer;
35.
36.     private bool isTalking = false;
37.
38.     void Start()
39.     {
40.         // Creates an instance of a speech config with specified subscription key a
    nd service region.
41.         // Replace with your own subscription key and service region (e.g., "westus
    ").
42.         speechConfig = SpeechConfig.FromSubscription("6ed5e2912bc6432a9f311229bd9eb
    d6d", "northeurope");
43.
44.         // The default format is Riff16Khz16BitMonoPcm.
45.         // We are playing the audio in memory as audio clip, which doesn't require
    riff header.
46.         // So we need to set the format to Raw16Khz16BitMonoPcm.
47.         speechConfig.SetSpeechSynthesisOutputFormat(SpeechSynthesisOutputFormat.Raw
    16Khz16BitMonoPcm);
48.
49.         // Creates a speech synthesizer.
50.         // Make sure to dispose the synthesizer after use!
51.         synthesizer = new SpeechSynthesizer(speechConfig, null);
52.         InvokeRepeating("ToldToSay", 2.0f, 1f);
53.         Say("Hololens ready");
```

```
54.     }
55.
56.     string oldtts ="";
57.     string currenttts="";
58.
59.     // Function that reads (TTS) a given string from the server
60.     public async void ToldToSay()
61.     {
62.         currenttts = MyTcpClient.giveMe("Texttospeech");
63.         if (currenttts!=oldtts && currenttts != "" && currenttts != "Texttospeech"
    && !isTalking)
64.         {
65.             isTalking = true;
66.             Say("Message incoming.");
67.             oldtts = currenttts;
68.             Debug.Log("Told to say: " + currenttts);
69.
70.             // This delay is set by the server, we have had success with ~1 second
    and longer delay.
71. #if !UNITY_EDITOR
72.             double delay = double.Parse(MyTcpClient.giveMe("Delay"));
73.             await Task.Delay(TimeSpan.FromSeconds(delay));
74. #endif
75.             Say(currenttts);
76.             isTalking = false;
77.         }
78.     }
79.
80.     // We start at waypoint 1
81.     private int Waypointid = 1;
82.
83.     //Gives the user feedback if the waypoint was automatically switched.
84.     public async void AutomaticNext()
85.     {
86.         if (MyTcpClient.giveMe("Automaticwaypoint") == "True" &&
87.             (Waypointid != int.Parse(MyTcpClient.giveMe("Waypointid"))) && !isTalki
    ng)
88.         {
89.             isTalking = true;
90.             Waypointid = int.Parse(MyTcpClient.giveMe("Waypointid"));
91.             MyTcpClient.forceSend("Automaticwaypoint:False");
92.             Say("Waypoint reached, advancing to waypoint number" + Waypointid);
93.             isTalking = false;
94.         }
95.     }
96.
97.     public void Say(string input)
98.     {
99.         Debug.Log("Text to speech: " + input);
100.             lock (threadLocker)
101.             {
102.                 waitingForSpeak = true;
103.             }
104.
105.             string newMessage = string.Empty;
106.
107.             // Starts speech synthesis, and returns after a single utterance is
    synthesized.
108.             using (var result = synthesizer.SpeakTextAsync(input).Result)
109.             {
110.                 // Checks result.
111.                 if (result.Reason == ResultReason.SynthesizingAudioCompleted)
112.                 {
113.                     // Native playback is not supported on Unity yet (currently
    only supported on Windows/Linux Desktop).
```

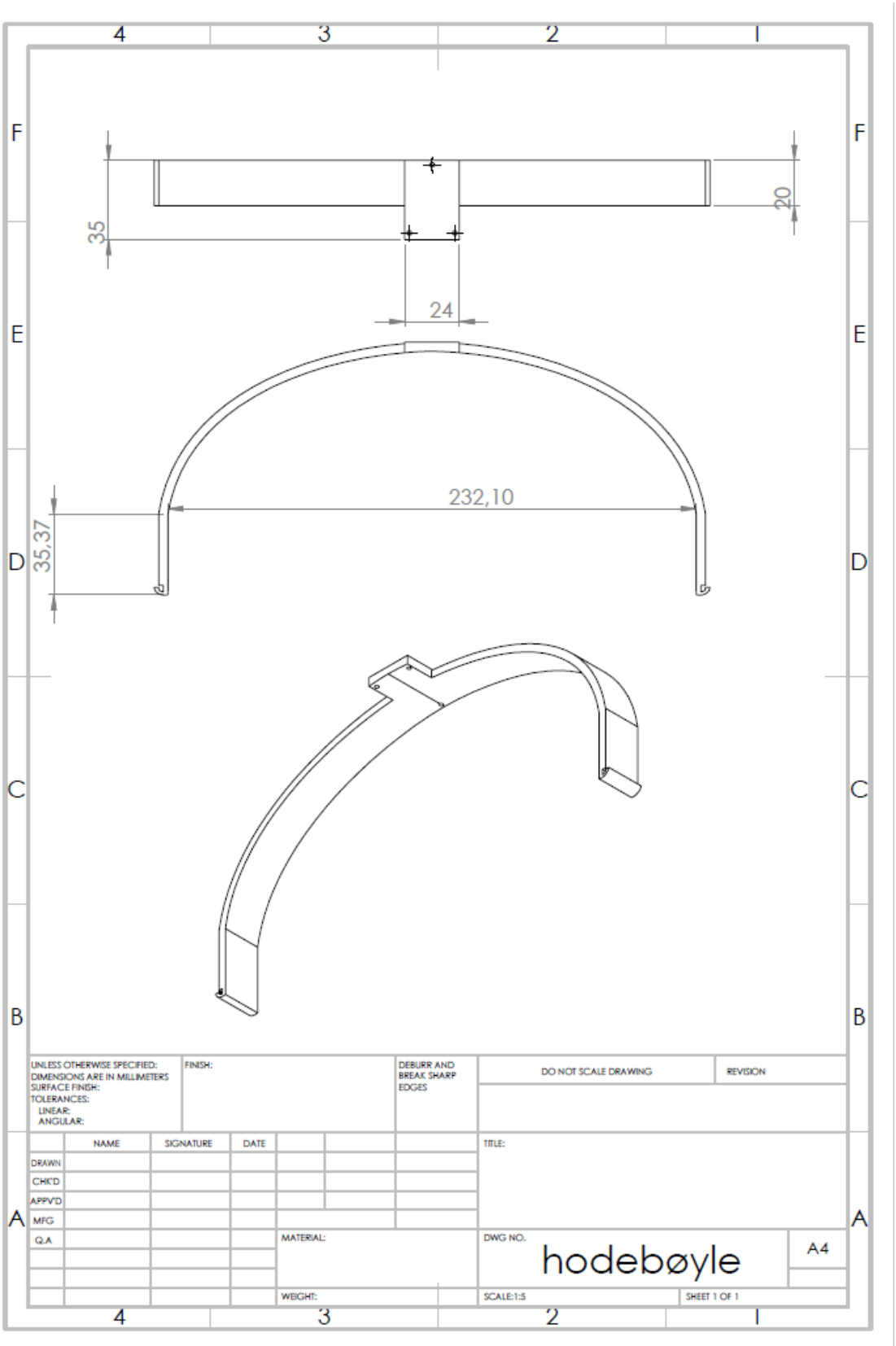
```
114.                         // Use the Unity API to play audio here as a short term solu
    tion.
115.                         // Native playback support will be added in the future relea
    se.
116.                         var sampleCount = result.AudioData.Length / 2;
117.                         var audioData = new float[sampleCount];
118.                         for (var i = 0; i < sampleCount; ++i)
119.                         {
120.                             audioData[i] = (short)(result.AudioData[i * 2 + 1] << 8
    | result.AudioData[i * 2]) / 32768.0F;
121.                         }
122.
123.                         // The output audio format is 16K 16bit mono
124.                         var audioClip = AudioClip.Create("SynthesizedAudio", sampleC
    ount, 1, 16000, false);
125.                         audioClip.SetData(audioData, 0);
126.                         audioSource.clip = audioClip;
127.                         audioSource.Play();
128.
129.                         newMessage = "Speech synthesis succeeded!";
130.                     }
131.                     else if (result.Reason == ResultReason.Canceled)
132.                     {
133.                         var cancellation = SpeechSynthesisCancellationDetails.FromRe
    sult(result);
134.                         newMessage = $"CANCELED:\nReason=[{cancellation.Reason}]\nEr
    rorDetails=[{cancellation.ErrorDetails}]\nDid you update the subscription info?";
135.                     }
136.                 }
137.
138.                 lock (threadLocker)
139.                 {
140.                     message = newMessage;
141.                     waitingForSpeak = false;
142.                 }
143.             }
144.
145.             void OnDestroy()
146.             {
147.                 synthesizer.Dispose();
148.             }
149.         }
```

Vedlegg N: PlaySound C# script

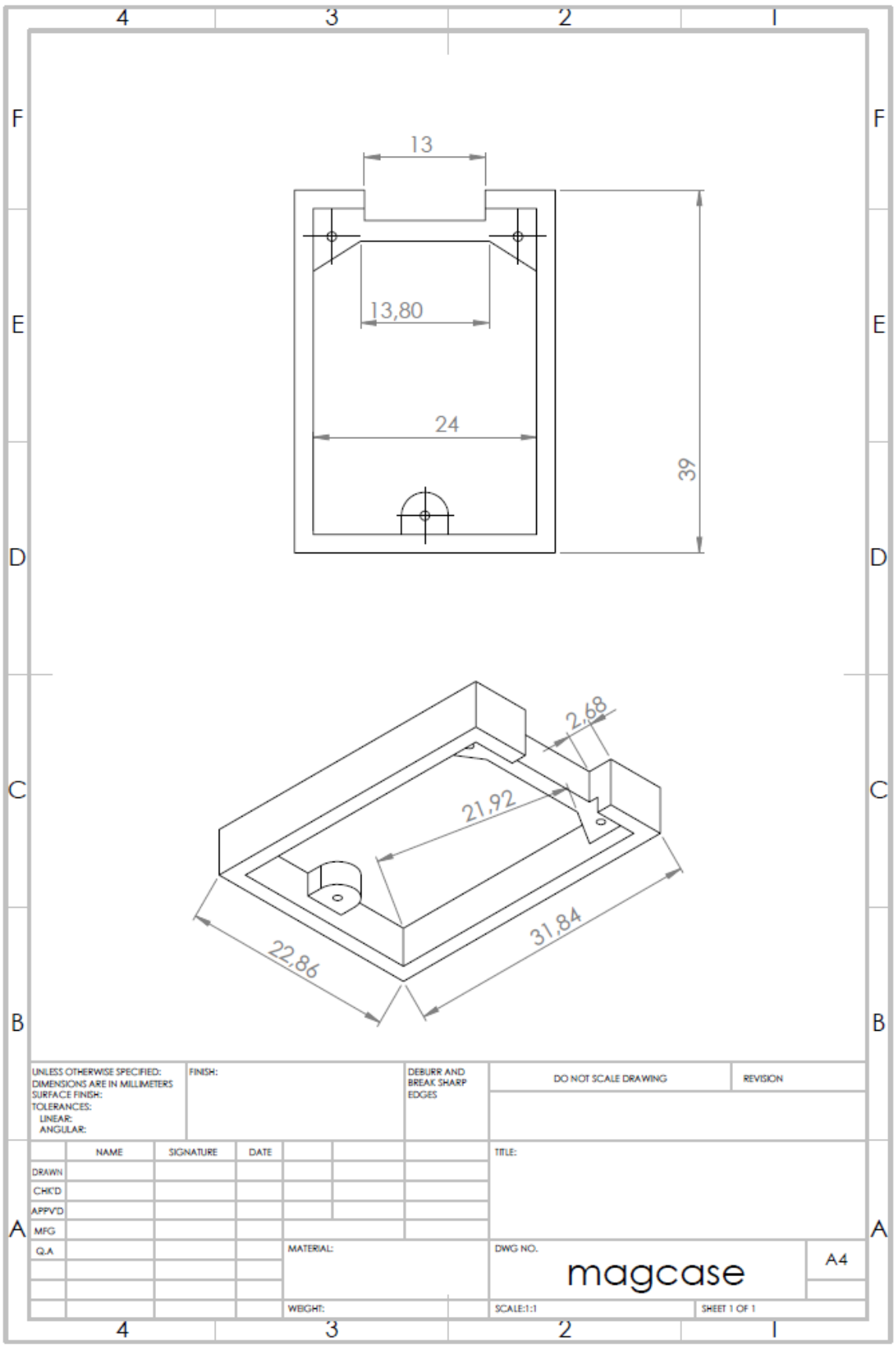
```
1.  // 13.11.2019 Sjøkrigsskolen
2.  // Bachelor, Augmented Reality
3.  // Written by Christer Algrøy, Julian Tobias Venstad and Markus Øvstedal
4.
5.  // Questions: julian.venstad@gmail.com
6.
7.  using System;
8.  using System.Collections;
9.  using System.Collections.Generic;
10. using System.Linq;
11. using UnityEngine;
12. using UnityEngine.Windows.Speech;
13. using UnityEngine.SceneManagement;
14.
15. public class PlaySound : MonoBehaviour
16. {
17.     public MyTcpClient MyTcpClient;
18.     public TextToSpeech TextToSpeech;
19.
20.     private Dictionary<string, Action> keyActs = new Dictionary<string, Action>();

21.     private KeywordRecognizer recognizer;
22.     public AudioClip AlarmSet, SettingNextWaypoint, SettingPreviousWaypoint, MyPlea
    sure;
23.     public AudioClip XteAlarmSound;
24.     public AudioClip[] numbers;
25.     AudioSource audioData;
26.
27.     public async void Start()
28.     {
29.         InvokeRepeating("XteAlarm", 2.0f, 10f);
30.     }
31.
32.     void Set()
33.     {
34.         audioData = GetComponent<AudioSource>();
35.         audioData.Play(0); // Set delay here
36.     }
37.
38.     public void XteAlarm()
39.     {
40.         Debug.Log(MyTcpClient.giveMe("Waypointxte"));
41.         Debug.Log(MyTcpClient.giveMe("Xte"));
42.         if (MyTcpClient.giveMe("Alarm") == "True")
43.         {
44.             if (int.Parse(MyTcpClient.giveMe("Waypointxte")) < Math.Abs(int.Parse(M
    yTcpClient.giveMe("Xte"))))
45.             {
46.                 audioData = GetComponent<AudioSource>();
47.                 audioData.clip = XteAlarmSound;
48.                 audioData.Play(0);
49.
50.             }
51.         }
52.     }
53.     public void Feedback(string input)
54.     {
55.         audioData = GetComponent<AudioSource>();
56.
57.         if (input == "AlarmSet")
58.             audioData.clip = AlarmSet;
```

```
59.         if (input == "SettingNextWaypoint")
60.             audioData.clip = SettingNextWaypoint;
61.         if (input == "SettingPreviousWaypoint")
62.             audioData.clip = SettingPreviousWaypoint;
63.         if (input == "MyPleasure")
64.             audioData.clip = MyPleasure;
65.     }
66. }
```

Vedlegg O: 3D printing



Figur O.1 Arbeidstegning av hodebøylen.



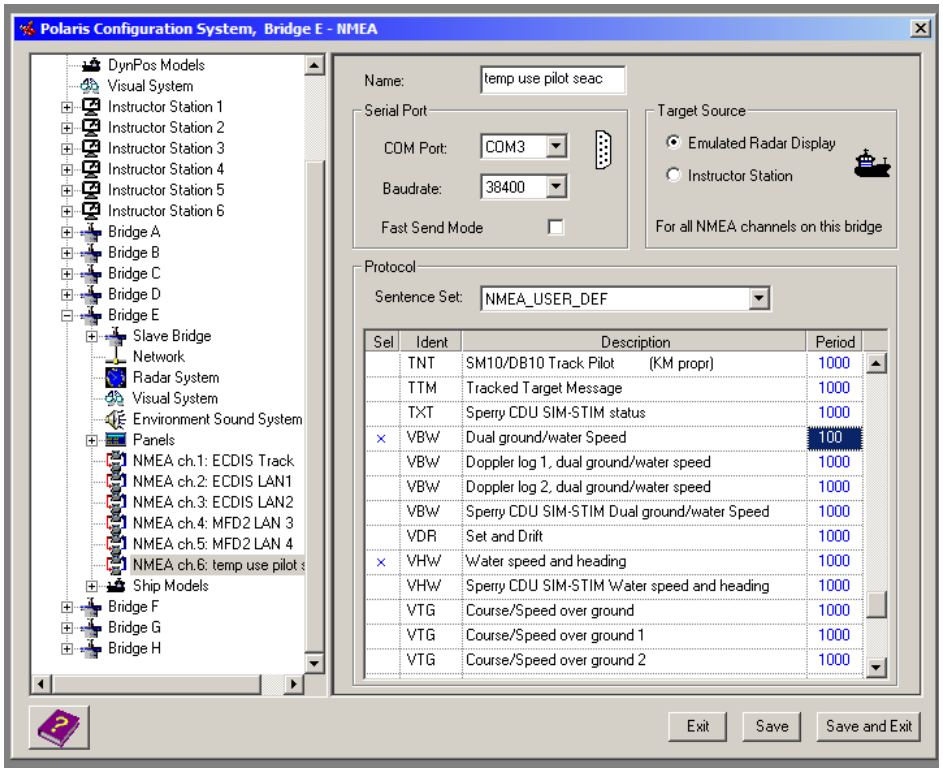
Figur O.2 Arbeidstegning av sensorholder.

Vedlegg P: Programvareversjoner

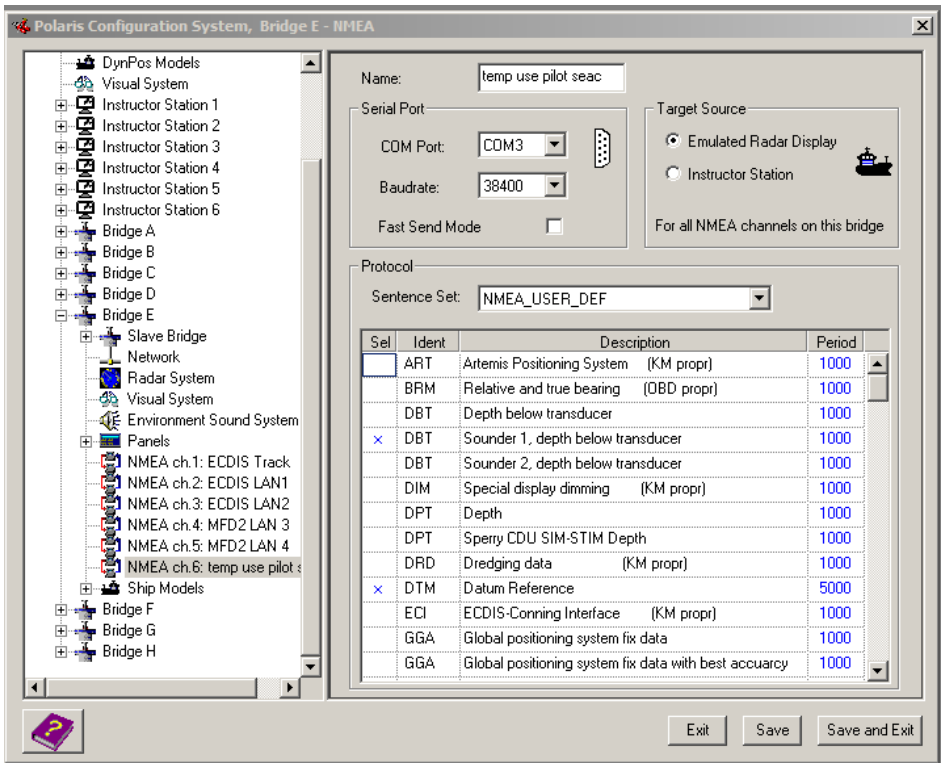
Program	Versjon
Unity	2018.4.2f1 Personal
Node-RED	V0.20.5
Vuforia	8.1.11
Raspbian	Stretch (9.9)
Azure Cognitive Services	Q4 2019
HoloLens OS build:	10.0.17763.865
HoloLens Firmware Revision number:	0.0005.05.S.1801091036R
XML	1.0

Vedlegg Q: Polaris konfigurasjon

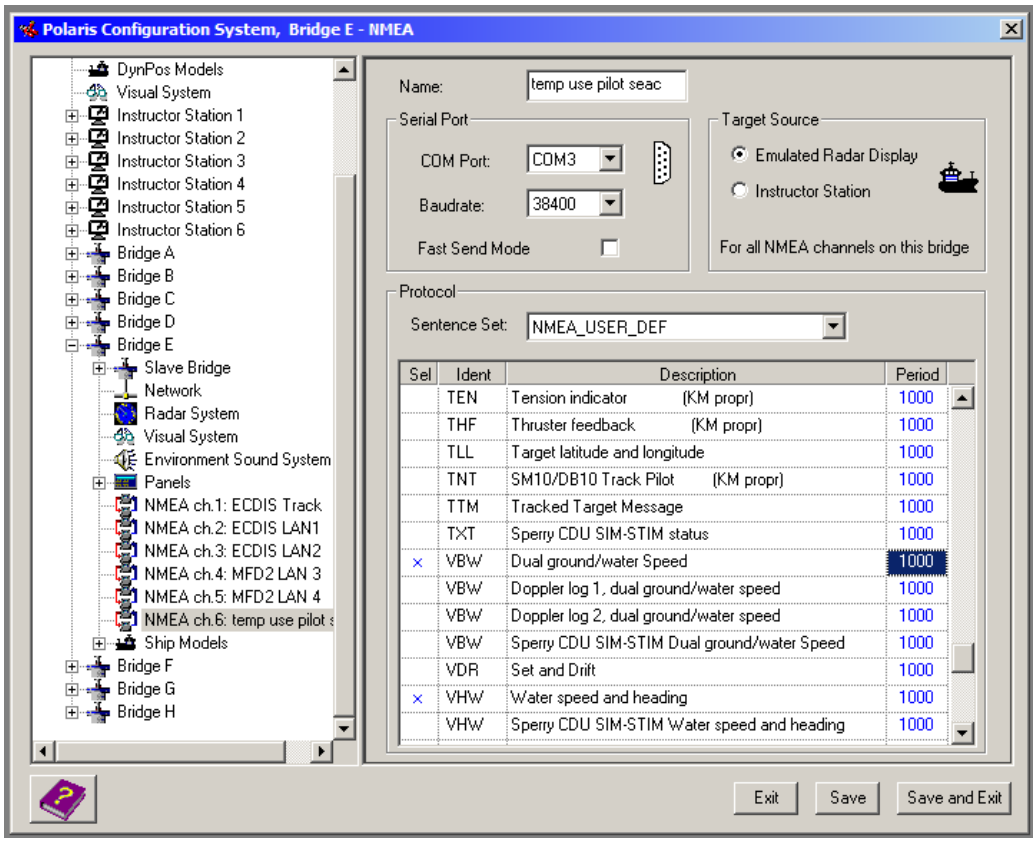
Vedlagt kan man se de valgte NMEA-strengene og hvor ofte de sendes.



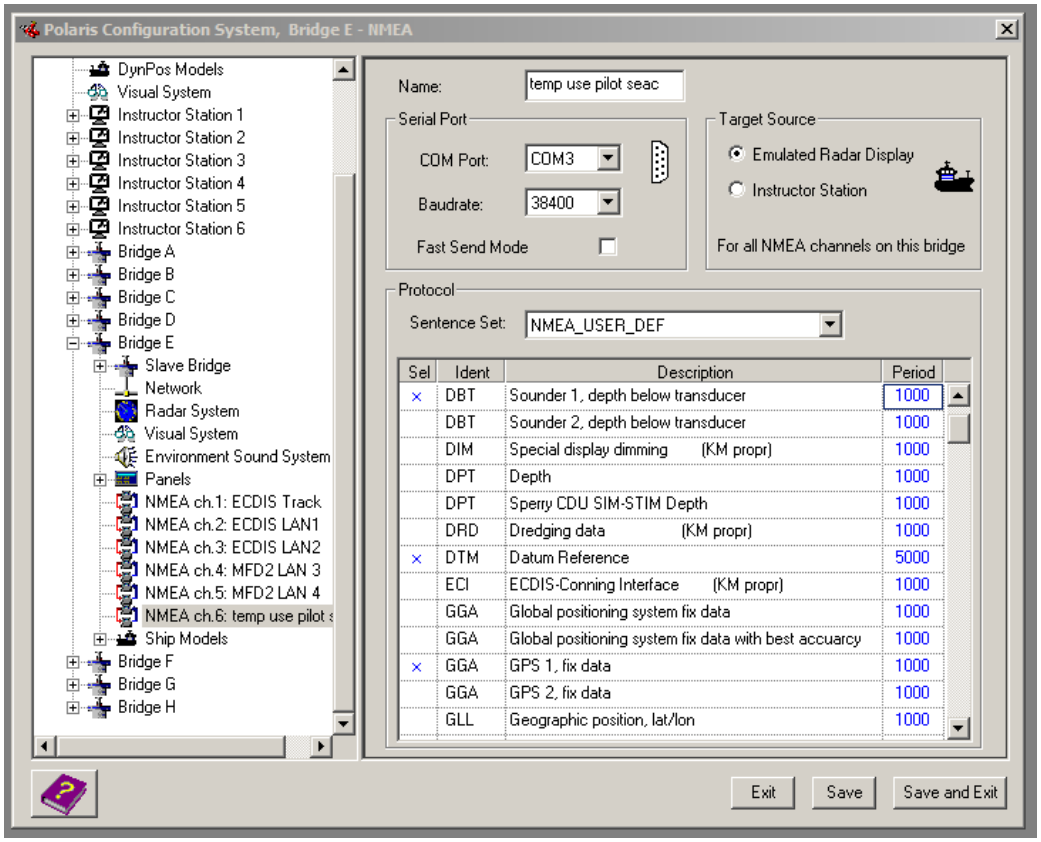
Figur Q.1 Polaris konfigurasjon 1/4.



Figur Q.2 Polaris konfigurasjon 2/4.

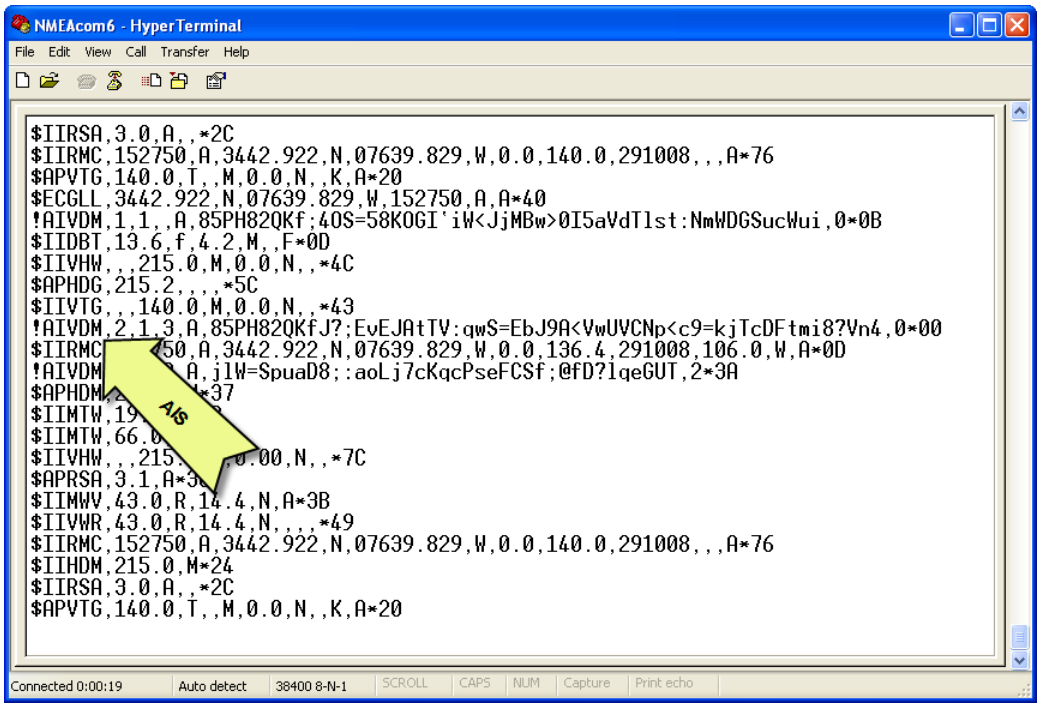


Figur Q.3 Polaris konfigurasjon 3/4.



Figur Q.4 Polaris konfigurasjon 4/4.

Figur Q.5 Datauthenting for første gang med RS232 adapter og HyperTerminal (PC).



Figur Q.6 HyperTerminal med AIS-strenger.

Vedlegg R: Node-RED Eksportert

Vedlagt ligger koden fra Node-RED eksportert. Denne kan limes inn i Node-RED og brukes.

[{"id":"b873dccc.a9864","type":"tab","label":"NMEA","disabled":false,"info":""},{"id":"607c7664.d3f748","type":"tab","label":"Waypoint","disabled":false,"info":""},{"id":"7a85299b.1bc6e8","type":"tab","label":"TCP","disabled":false,"info":""},{"id":"fd97e6d8.2cb9a8","type":"tab","label":"UI editor","disabled":false,"info":""},{"id":"19d59705.d99769","type":"tab","label":"Compass","disabled":true,"info":""},{"id":"e7285da5.dfd63","type":"ui_group","z":"","name":"Waypoint","tab":"722dfb92.be72f4","order":2,"disp":true,"width":"10","collapse":true},{"id":"2e8896f5.e9421a","type":"ui_tab","z":"","name":"AR-data","icon":"dashboard","order":2,"disabled":false,"hidden":false},{"id":"51abf36f.08514c","type":"ui_base","z":0,"theme":{"name":"theme-dark","lightTheme":{"default":"#0094CE","baseColor":"#0094CE","baseFont":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif","edited":true,"reset":false},"darkTheme":{"default":"#097479","baseColor":"#699ee0","baseFont":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif","edited":true,"reset":false},"customTheme":{"name":"Untitled Theme 1","default":"#4B7930","baseColor":"#4B7930","baseFont":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"},"themeState":{"base-color":{"default":"#097479","value":"#699ee0","edited":true},"page-titlebar-backgroundColor":{"value":"#699ee0","edited":false},"page-backgroundColor":{"value":"#111111","edited":false},"page-sidebar-backgroundColor":{"value":"#ffffff","edited":false},"group-textColor":{"value":"#a8c7ed","edited":false},"group-borderColor":{"value":"#555555","edited":false},"group-backgroundColor":{"value":"#333333","edited":false},"widget-textColor":{"value":"#eeeeee","edited":false},"widget-backgroundColor":{"value":"#699ee0","edited":false},"widget-borderColor":{"value":"#333333","edited":false},"base-font":{"value":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"}},"angularTheme":{"primary":"indigo","accents":"blue","warn":"red","background":"grey"}},"site":{"name":"Node-RED Dashboard","hideToolbar":"false","allowSwipe":"true","lockMenu":"false","allowTempTheme":"true","dateFormat":"DD/MM/YYYY","sizes":{"sx":48,"sy":48,"gx":6,"gy":6,"cx":6,"cy":6,"px":0,"py":0}}},{"id":"90db4fbd.7c6ee","type":"ui_group","z":"","name":"NAVIGASJONSDATA AIS","tab":"3d9faefc.9dca62","order":2,"disp":true,"width":"11","collapse":false},{"id":"9d9e4e1e.b57cc","type":"serial-

port","z":"","serialport":"/dev/ttyAMA0","serialbaud":"115200","databits":"8","parity":"none","stopb
its":"1","waitfor":"","newline":"\\n","bin":"false","out":"char","addchar":"","responsetimeout":"1000"
},{"id":"73d3c90d.8aa4b8","type":"lftp-
config","z":0,"host":"192.168.119.158","protocol":"ftp","port":"21","escape":true,"retries":"2","timeo
ut":"10","retryInterval":"5","retryMultiplier":"1","requiresPassword":true,"additionalLftpCommands"
:""},{"id":"722dfb92.be72f4","type":"ui_tab","z":"","name":"WayPointData","icon":"dashboard","or
der":1,"disabled":false,"hidden":false},{"id":"1a4a8be8.65daf4","type":"ui_group","z":"","name":"Ga
uges","tab":"2e8896f5.e9421a","order":1,"disp":true,"width":"6","collapse":false},{"id":"930b4a6d.9
e9de8","type":"serial-
port","z":"","serialport":"/dev/ttyUSB0","serialbaud":"38400","databits":"8","parity":"none","stopbits
":"1","waitfor":"","newline":"\\n","bin":"false","out":"char","addchar":"","responsetimeout":"10000"}
,{"id":"e8a812ff.2917","type":"ui_group","z":"","name":"DASHBOARD","tab":"d1c1342.9c486c8","
disp":true,"width":"12","collapse":false},{"id":"baa364a9.840768","type":"json-db-
collection","z":"","name":"","collection":"myDatas","save":true},{"id":"b378ad25.2dddf","type":"ui_
group","z":"","name":"Position and
time","tab":"2e8896f5.e9421a","order":3,"disp":true,"width":"15","collapse":false},{"id":"7ee01867.
548638","type":"ui_group","z":"","name":"Acc/Mag","tab":"","disp":true,"width":"12","collapse":fals
e},{"id":"3d9faefc.9dca62","type":"ui_tab","z":"","name":"AIS","icon":"dashboard","order":4,"disabl
ed":false,"hidden":false},{"id":"e30ab183.2f1ed","type":"ui_tab","z":"","name":"UI
editor","icon":"dashboard","order":3,"disabled":false,"hidden":false},{"id":"50eda971.220498","type
":"ui_group","z":"","name":"Move","tab":"e30ab183.2f1ed","order":2,"disp":true,"width":"6","collap
se":false},{"id":"13958700.8d0729","type":"ui_group","z":"","name":"Choose
element","tab":"e30ab183.2f1ed","order":1,"disp":true,"width":"6","collapse":false},{"id":"554f65bf.
b478dc","type":"ui_group","z":"","name":"Gauges","tab":"2e8896f5.e9421a","order":2,"disp":true,"w
idth":"6","collapse":false},{"id":"3c8b9d79.0630d2","type":"ui_group","z":"","name":"Text","tab":"e
30ab183.2f1ed","order":3,"disp":true,"width":"6","collapse":false},{"id":"edf83ebe.a41ba","type":"ui
_group","z":"","name":"Text to
speech","tab":"e30ab183.2f1ed","order":4,"disp":true,"width":"6","collapse":false},{"id":"a37e40c.7c
a0cc","type":"ui_spacer","name":"spacer","group":"3c8b9d79.0630d2","order":3,"width":1,"height":1
},{"id":"bfe0f484.9cbb58","type":"serial-
port","serialport":"/dev/ttyAMA0","serialbaud":"9600","databits":8,"parity":"none","stopbits":1,"new
line":"\\n","addchar":"false"},{"id":"80858abc.d026f8","type":"ui_group","z":"","name":"Calculated
Values","tab":"722dfb92.be72f4","order":1,"disp":true,"width":"6","collapse":true},{"id":"529cc127.
d804e","type":"ui_tab","z":"","name":"HoloLens
Heading","icon":"dashboard","order":5,"disabled":false,"hidden":false},{"id":"4301e921.cca2c8","ty
pe":"ui_group","z":"","name":"Heading","tab":"529cc127.d804e","order":1,"disp":true,"width":"6","c
ollapse":false},{"id":"d1c1342.9c486c8","type":"ui_tab","z":"","name":"Tablet","icon":"dashboard","

order":6,"disabled":false,"hidden":false},{"id":"e23e9370.09773","type":"ui_tab","z":"","name":"Tab 7","icon":"dashboard","order":7,"disabled":false,"hidden":false},{"id":"2f586ea4.142ee2","type":"ui_group","z":"","name":"NAVIGASJONSDATA
AIS","tab":"3d9faefc.9dca62","order":1,"disp":true,"width":"6","collapse":false},{"id":"3e7bcb5e.75c294","type":"ui_gauge","z":"b873dccc.a9864","name":"Speedometer","group":"2f586ea4.142ee2","order":1,"width":0,"height":0,"gtype":"gage","title":"Speedometer","label":"Knots","format":"{{value}}","min":0,"max":"60","colors":["#00b500","#e6e600","#ca3838"],"seg1":"15","seg2":"45","x":1280,"y":60,"wires":[]},{"id":"1c9841db.5b356e","type":"ui_gauge","z":"b873dccc.a9864","name":"COG","group":"2f586ea4.142ee2","order":2,"width":0,"height":0,"gtype":"compass","title":"COG","label":"degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":1250,"y":100,"wires":[]},{"id":"6e896ec8.28fd","type":"ui_text","z":"b873dccc.a9864","group":"90db4fbd.7c6ee","order":1,"width":0,"height":0,"name":"longitude","label":"Longitude:","format":"{{msg.payload}}","layout":"row-
spread","x":1260,"y":220,"wires":[]},{"id":"436c6a52.7793c4","type":"ui_text","z":"b873dccc.a9864","group":"90db4fbd.7c6ee","order":2,"width":0,"height":0,"name":"latitude","label":"Latitude:","format":"{{msg.payload}}","layout":"row-
spread","x":1260,"y":260,"wires":[]},{"id":"acd70102.b74fe","type":"file","z":"b873dccc.a9864","name":"","filename":"/home/pi/share/aisdata/current.txt","appendNewline":true,"createDir":true,"overwriteFile":"true","encoding":"none","x":500,"y":240,"wires":[[]]},{"id":"b44846b8.326bd8","type":"file","z":"b873dccc.a9864","name":"","filename":"/home/pi/share/aisdata/log.txt","appendNewline":true,"createDir":true,"overwriteFile":"false","encoding":"none","x":480,"y":200,"wires":[[]]},{"id":"e7ab5518.9b4a48","type":"ui_text","z":"b873dccc.a9864","group":"90db4fbd.7c6ee","order":3,"width":0,"height":0,"name":"Source:","label":"Source:","format":"{{msg.payload}}","layout":"row-left","x":1260,"y":300,"wires":[]},{"id":"d71c81.1841c38","type":"ui_text","z":"b873dccc.a9864","group":"90db4fbd.7c6ee","order":4,"width":0,"height":0,"name":"Navigationstatus:","label":"Navigationstatus:","format":"{{msg.payload}}","layout":"row-left","x":1290,"y":340,"wires":[]},{"id":"ba3ea6ba.cafa88","type":"ui_gauge","z":"b873dccc.a9864","name":"True
heading","group":"2f586ea4.142ee2","order":3,"width":0,"height":0,"gtype":"compass","title":"True heading","label":"degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":1270,"y":180,"wires":[]},{"id":"ad37b61.6a64848","type":"ui_gauge","z":"b873dccc.a9864","name":"Rate of
turn","group":"2f586ea4.142ee2","order":4,"width":0,"height":0,"gtype":"gage","title":"Rate of turn","label":"degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":1270,"y":140,"wires":[]},{"id":"e90c3060.71c3","type":"ais","z":"b873dccc.a9864","name":"","x":410,"y":160,"wires":[["acd70102.b74fe","b44846b8.326bd8","88a095

f1.b984e8"]]},{"id":"da3ff016.960df","type":"serial in","z":"b873dccc.a9864","name":"NMEA data USB : BRO
E","serial":"930b4a6d.9e9de8","x":130,"y":80,"wires":[["c764cd15.035b3","30f1d35c.935cdc"]]},{"id":"88a095f1.b984e8","type":"function","z":"b873dccc.a9864","name":"SOG/COG/rateOfTurn/trueHeading/longitude/latitude/source/navigationStatus","func":"var msg1 = { payload: msg.payload.speedOverGround };\nvar msg2 = { payload: msg.payload.courseOverGround };\nvar msg3 = { payload: msg.payload.rateOfTurn};\nvar msg4 = { payload: msg.payload.trueHeading};\nvar msg5 = { payload: msg.payload.longitude};\nvar msg6 = { payload: msg.payload.latitude};\nvar msg7 = { payload: msg.payload.source};\nvar msg8 = { payload: msg.payload.navigationStatus};\n\nreturn [msg1,msg2,msg3,msg4,msg5,msg6,msg7,msg8]","outputs":8,"noerr":0,"x":890,"y":160,"wires":[["3e7bcb5e.75c294"],["1c9841db.5b356e"],["ad37b61.6a64848"],["ba3ea6ba.cafa88"],["6e896ec8.28fd"],["436c6a52.7793c4"],["e7ab5518.9b4a48"],["d71c81.1841c38"]]},{"id":"c764cd15.035b3","type":"function","z":"b873dccc.a9864","name":"Includes('!AIVDO1,1,')","func":"if (msg.payload.includes('!AIVDO,1,1,'))\n   return msg;","outputs":1,"noerr":0,"x":440,"y":80,"wires":[["e90c3060.71c3"]]},{"id":"1d308e86.2182f1","type":"comment","z":"b873dccc.a9864","name":"Leser data fra AIS Pilot Plug","info":"","x":140,"y":40,"wires":[]},{"id":"2988095d.09fe26","type":"comment","z":"b873dccc.a9864","name":"Sender kun videre ønsket AIS-data","info":"","x":420,"y":40,"wires":[]},{"id":"cfa344fc.9ec698","type":"comment","z":"b873dccc.a9864","name":"Filtrerer meldingen til flere outputs","info":"","x":1000,"y":60,"wires":[]},{"id":"c360579c.fc1248","type":"comment","z":"b873dccc.a9864","name":"Skriver til dashbordelementer","info":"","x":1240,"y":20,"wires":[]},{"id":"17c85a72.53ddf6","type":"comment","z":"b873dccc.a9864","name":"Tolker AIS-data til lesbar form","info":"","x":440,"y":120,"wires":[]},{"id":"c042823c.7505a","type":"ui_gauge","z":"b873dccc.a9864","name":"Speedometer","group":"554f65bf.b478dc","order":1,"width":0,"height":0,"gtype":"gage","title":"Speedometer","label":"Knots","format":"{{value}}","min":0,"max":"60","colors":["#00b500","#e6e600","#ca3838"],"seg1":"15","seg2":"45","x":800,"y":440,"wires":[]},{"id":"51b09ce4.9a6194","type":"ui_gauge","z":"b873dccc.a9864","name":"COG","group":"1a4a8be8.65daf4","order":2,"width":0,"height":0,"gtype":"compass","title":"COG","label":"degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":770,"y":480,"wires":[]},{"id":"a279b627.f871e8","type":"ui_text","z":"b873dccc.a9864","group":"b378ad25.2dddf","order":1,"width":0,"height":0,"name":"longitude","label":"Longitude:","format":"{{msg.payload}}","layout":"row-spread","x":780,"y":400,"wires":[]},{"id":"7ac69b12.a4c994","type":"ui_text","z":"b873dccc.a9864","group":"b378ad25.2dddf","order":2,"width":0,"height":0,"name":"latitude","label":"Latitude:","form

at":"{{msg.payload}}","layout":"row-
spread","x":780,"y":360,"wires":[]},{"id":"494f75a7.07a3cc","type":"ui_gauge","z":"b873dccc.a9864
","name":"True
heading","group":"1a4a8be8.65daf4","order":1,"width":0,"height":0,"gtype":"compass","title":"True
heading","label":"degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600
","#ca3838"],"seg1":"","seg2":"","x":790,"y":520,"wires":[]},{"id":"fcb7d710.361e88","type":"ui_ga
uge","z":"b873dccc.a9864","name":"Rate of
turn","group":"1a4a8be8.65daf4","order":3,"width":0,"height":0,"gtype":"gage","title":"Rate of
turn","label":"degrees","format":"{{value}}","min":"-
30","max":"30","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":790,"y":560,"wir
es":[]},{"id":"257aeb91.2701c4","type":"ui_text","z":"b873dccc.a9864","group":"b378ad25.2dddf","
order":3,"width":0,"height":0,"name":"Time","label":"Time:","format":"{{msg.payload}}","layout":"
row-
spread","x":770,"y":240,"wires":[]},{"id":"79aaec3c.bd9e54","type":"ui_gauge","z":"b873dccc.a9864
","name":"Wind
Speed","group":"554f65bf.b478dc","order":3,"width":0,"height":0,"gtype":"gage","title":"Wind
Speed","label":"m/s","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600","#c
a3838"],"seg1":"","seg2":"","x":790,"y":280,"wires":[]},{"id":"277e0239.1748ce","type":"ui_gauge",
"z":"b873dccc.a9864","name":"Wind
Angle","group":"554f65bf.b478dc","order":2,"width":0,"height":0,"gtype":"compass","title":"Wind
Angle","label":"Degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600"
,"#ca3838"],"seg1":"","seg2":"","x":790,"y":320,"wires":[]},{"id":"6b83dd48.018484","type":"tcp
out","z":"7a85299b.1bc6e8","host":"","port":"","beserver":"reply","base64":false,"end":false,"name":
"reply: all open connections will be
messaged","x":930,"y":100,"wires":[]},{"id":"f14eea0d.8eb868","type":"debug","z":"7a85299b.1bc6e
8","name":"show incomming text at the debug
tab","active":false,"console":"false","complete":"payload","x":190,"y":220,"wires":[]},{"id":"5fcc74a
0.d7aeec","type":"template","z":"7a85299b.1bc6e8","name":"\"SERVER GOT:\"
+","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"SER
VER
GOT:\n{{payload}}","x":130,"y":180,"wires":[["f14eea0d.8eb868"]]},{"id":"57dd9502.61470c","typ
e":"function","z":"7a85299b.1bc6e8","name":".toString()","func":"msg.payload =
msg.payload.toString();\nreturn
msg;","outputs":1,"noerr":0,"x":380,"y":100,"wires":[["5fcc74a0.d7aeec","df4ab9d5.40f3f8"]]},{"id":
"6207297a.cd9038","type":"tcp in","z":"7a85299b.1bc6e8","name":"Server holding connections at
:1025","server":"server","host":"","port":"1025","datamode":"stream","datatype":"buffer","newline":"

\\n","topic":"text","base64":false,"x":160,"y":100,"wires":[["57dd9502.61470c"]]},{"id":"df4ab9d5.40f3f8","type":"function","z":"7a85299b.1bc6e8","name":"Check Request","func":"//Available variables\nvar keyword=[\"Time\", \"Lat\", \"Lon\", \"Knot\", \"Cog\", \"TrueHeading\", \"Rot\", \"Waypointid\", \"Waypointname\", \"Waypointlat\", \"Waypointlon\", \n\"Waypointturnradius\", \"Waypointlegtype\", \"Waypointxte\", \"Waypointspeed\", \"Waypointdanger\", \"Waypointmessage\", \"Waypointrtime\", \"Bearing\", \n\"Nextbearing\", \"Nextwaypointname\", \"Nextwaypointmessage\", \"Distancetowaypoint\", \"Distancetonextwaypoint\", \"Timetowaypoint\", \"Timetonextwaypoint\", \"Voicewaypoint\",\n\"Element\", \"Moveby\", \"Direction\", \"Movenow\", \"Magheading\", \"Red\", \"Green\", \"Blue\", \"Alpha\", \"Texttospeech\", \"Fontsize\", \"Delay\", \"Automaticwaypoint\", \"Voiceassistance\" , \"Xte\", \"Prevwaypointname\"];\n\n//Writes data to variables with given value after ':', sent from the Hololens\nif(msg.payload.includes(\":\")){\nvar splitstring = msg.payload.split(':');\n\nfor (i = 0; i < keyword.length; i++) {\nif (splitstring[0]==keyword[i]){\n global.set(splitstring[0],splitstring[1]); //Example: \"Voiceassistance:True\" --> Sets the global variable Voiceassistance to True\n }\n\n}\n}\n\n//Formats the output string with seperators as '|' and ':'\nmsg.payload=\"\";\nfor (i = 0; i < keyword.length; i++) {\n if(i===0){\n msg.payload= msg.payload + keyword[i]+\":\"+global.get(keyword[i]);\n i++;\n \n }\n msg.payload= msg.payload + \"|\"+ keyword[i]+\":\"+global.get(keyword[i]);\n}\nreturn msg;\n\n\n\n\n","outputs":1,"noerr":0,"x":540,"y":100,"wires":[["1ee01fd6.b9d17"]]},{"id":"8f638264.cd353","type":"debug","z":"7a85299b.1bc6e8","name":"show incomming text at the debug tab","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":910,"y":220,"wires":[]},{"id":"9cd6871d.0f87e8","type":"template","z":"7a85299b.1bc6e8","name":"\"SERVER SENT:\"
+","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"SERVER
SENT:\n{{payload}}","x":850,"y":180,"wires":[["8f638264.cd353"]]},{"id":"1ee01fd6.b9d17","type":"template","z":"7a85299b.1bc6e8","name":"/n","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"\n{{payload}}","x":690,"y":100,"wires":[["6b83dd48.018484","9cd6871d.0f87e8"]]},{"id":"13770103.9d4c6f","type":"i2c
in","z":"19d59705.d99769","name":"","address":"30","command":"","count":"1","x":790,"y":180,"wires":[[]]},{"id":"e44473b.a56089","type":"function","z":"19d59705.d99769","name":"name commands","func":"var topics = [\"TEMP_OUT_L\", \"TEMP_OUT_H\", \"STATUS_REG_M\", \"OUT_X_L_M\", \"OUT_X_H_M\", \"OUT_Y_L_M\", \"OUT_Y_H_M\", \"OUT_Z_L_M\", \"OUT_Z_H_M\", \"WHO_AM_I\", \"INT_CTRL_M\", \"INT_SRC_M\", \"INT_THS_L_M\", \"INT_THS_H_M\", \"OFFSET_X_L_M\", \"OFFSET_X_H_M\", \"OFFSET_Y_L_M\", \"OFFSET_Y_H_M\", \"OFFSET_Z_L_M\", \"OFFSET_Z_H_M\", \"REFERENCE_X\", \"REFERENCE_Y\", \"REFERENCE_Z\", \"CTRL_REG0\", \"CTRL_REG1\", \"CTRL_REG2\",

\"CTRL_REG3\", \"CTRL_REG4\", \"CTRL_REG5\", \"CTRL_REG6\", \"CTRL_REG7\", \"STATUS_REG_A\", \"OUT_X_L_A\", \"OUT_X_H_A\", \"OUT_Y_L_A\", \"OUT_Y_H_A\", \"OUT_Z_L_A\", \"OUT_Z_H_A\", \"FIFO_CTRL\", \"FIFO_SRC\", \"IG_CFG1\", \"IG_SRC1\", \"IG_THS1\", \"IG_DUR1\", \"IG_CFG2\", \"IG_SRC2\", \"IG_THS2\", \"IG_DUR2\", \"CLICK_CFG\", \"CLICK_SRC\", \"CLICK_THS\", \"TIME_LIMIT\", \"TIME_LATENCY\", \"TIME_WINDOW\", \"ACT_THS\", \"ACT_DUR\", \"MAG_SCALE_2\", \"MAG_SCALE_4\", \"MAG_SCALE_8\", \"MAG_SCALE_12\"];\nvar commands = [0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0F, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1A, 0x1B, 0x1C, 0x1D, 0x1E, 0x1F, 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2A, 0x2B, 0x2C, 0x2D, 0x2E, 0x2F, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3A, 0x3B, 0x3C, 0x3D, 0x3E, 0x3F, 0x00, 0x20, 0x40, 0x60];\nvar command = {};\nfor (var i = 0; i < commands.length; i++){\n command.merge({[topics[i]] : commands[i]});\n}\ndelete command.merge;\nmsg.commands = command;\ncommand = undefined;\n\nreturn msg;","outputs":1,"noerr":0,"x":390,"y":180,"wires":[["6facdde4.1df0c4"]]},{"id":"902e1b1a.aa7958","type":"inject","z":"19d59705.d99769","name":"START COMPASS","topic":"","payload":"","payloadType":"date","repeat":"0.5","crontab":"","once":true,"onceDelay":0.1,"x":140,"y":180,"wires":[["e44473b.a56089"]]},{"id":"6facdde4.1df0c4","type":"function","z":"19d59705.d99769","name":"I2C getMag","func":"msg.command = msg.commands.OUT_X_H_M;\nnode.send(msg);\nmsg.command = msg.commands.OUT_X_L_M;\nnode.send(msg);\nmsg.command = msg.commands.OUT_Y_H_M;\nnode.send(msg);\nmsg.command = msg.commands.OUT_Y_L_M;\nnode.send(msg);\nmsg.command = msg.commands.OUT_Z_H_M;\nnode.send(msg);\nmsg.command = msg.commands.OUT_Z_L_M;\nnode.send(msg);","outputs":1,"noerr":0,"x":610,"y":180,"wires":[["13770103.9d4c6f","652cb735.f6ef68"]]},{"id":"652cb735.f6ef68","type":"function","z":"19d59705.d99769","name":"Check","func":"var MagStatus = context.get(\"MagStatus\")||[false,false,false,false,false,false];\nvar MagValues = context.get(\"MagValues\")||[0,0,0,0,0,0];\nvar MagXY = context.get(\"MagXY\")||[0,0];\nswitch (msg.command) {\n case msg.commands.OUT_X_H_M: MagStatus[0] = true; //First arrived\n MagValues[0] = msg.payload;\n break;\n case msg.commands.OUT_X_L_M: MagStatus[1] = MagStatus[0]; //If first has arrived -> Second accepted\n MagValues[1] = msg.payload;\n break;\n case msg.commands.OUT_Y_H_M: MagStatus[2] = true; //First arrived\n MagValues[2] = msg.payload;\n break;\n case msg.commands.OUT_Y_L_M: MagStatus[3] = MagStatus[2]; //If first has arrived -> Second accepted\n MagValues[3] = msg.payload;\n break;\n case msg.commands.OUT_Z_H_M: MagStatus[4] = true; //First arrived\n

MagValues[4] = msg.payload;\n break;\n case msg.commands.OUT_Z_L_M: MagStatus[5] = MagStatus[4]; //If first has arrived -> Second accepted\n MagValues[5] = msg.payload;\n break;\n}\nif (MagStatus[0] && MagStatus[1]) {\n msg.topic = \"OUT_X_M\";\n msg.payload = (MagValues[0]<<8)|MagValues[0];\n if (msg.payload > 32767) {\n msg.payload=msg.payload-65536;\n }\n MagXY[0] = msg.payload;\n node.send(msg);\n MagStatus[0] = false; //Waiting for new values\n MagStatus[1] = false; //Waiting for new values\n}\nif (MagStatus[2] && MagStatus[3]) {\n msg.topic = \"OUT_Y_M\";\n msg.payload = (MagValues[2]<<8)|MagValues[3];\n if (msg.payload > 32767) {\n msg.payload=msg.payload-65536;\n }\n MagXY[1] = msg.payload;\n node.send(msg);\n MagStatus[2] = false; //Waiting for new values\n MagStatus[3] = false; //Waiting for new values\n\n\n}\nif (MagStatus[4] && MagStatus[5]) {\n msg.topic = \"OUT_Z_M\";\n msg.payload = (MagValues[4]<<8)|MagValues[5];\n if (msg.payload > 32767) {\n msg.payload=msg.payload-65536;\n }\n node.send(msg);\n MagStatus[4] = false; //Waiting for new values\n MagStatus[5] = false; //Waiting for new values\n}\ncontext.set(\"MagStatus\", MagStatus);\ncontext.set(\"MagValues\", MagValues);\ncontext.set(\"MagXY\", MagXY);\n","outputs":1,"noerr":0,"x":210,"y":220,"wires":[["a39c51d.21522b"]]},{"id":"be38e210.3de35","type":"i2c out","z":"19d59705.d99769","name":"","address":"30","command":"","payload":"payload","payloadType":"msg","count":"2","x":800,"y":60,"wires":[]},{"id":"d7a70f0e.5cc8c","type":"function","z":"19d59705.d99769","name":"I2C INIT Mag","func":"//https://github.com/DexterInd/GrovePi/blob/master/Software/Python/grove_6axis_acc_compass/lsm303d.py\n// self.write_reg(0x57, self.CTRL_REG1)\t\t\t\t# 0x57 = ODR=50hz, all accel axes on\n// self.write_reg((3<<6)|(0<<3), self.CTRL_REG2) \t# set full-scale\n// self.write_reg(0x00, self.CTRL_REG3) \t\t\t# no interrupt\n// self.write_reg(0x00, self.CTRL_REG4) \t\t\t# no interrupt\n// self.write_reg((4<<2), self.CTRL_REG5) \t\t\t# 0x10 = mag 50Hz output rate\n// self.write_reg(self.MAG_SCALE_2, self.CTRL_REG6)\t# magnetic scale = +/-1.3Gauss\n// self.write_reg(0x00, self.CTRL_REG7) \t\t\t# 0x00 = continouous conversion mode\n\n//Source: https://github.com/Seeed-Studio/Grove_6Axis_Accelerometer_And_Compass_v2/blob/master/LSM303D.cpp\n\nmsg.command = msg.commands.CTRL_REG5;\nmsg.payload = (4<<2);\nnode.send(msg);\nmsg.command = msg.commands.CTRL_REG6;\nmsg.payload = msg.commands.MAG_SCALE_2;\nnode.send(msg);\nmsg.command = msg.commands.CTRL_REG7;\nmsg.payload = 0x00;\nnode.send(msg);\n\n\n\n","outputs":1,"noerr":0,"x":610,"y":60,"wires":[["be38e210.3de35"]]},{"id":"e56b7f1a.03e24","type":"function","z":"19d59705.d99769","name":"name commands","func":"\n\nvar topics = [\"TEMP_OUT_L\", \"TEMP_OUT_H\", \"STATUS_REG_M\",

\"OUT_X_L_M\", \"OUT_X_H_M\", \"OUT_Y_L_M\", \"OUT_Y_H_M\", \"OUT_Z_L_M\", \"OUT_Z_H_M\", \"WHO_AM_I\", \"INT_CTRL_M\", \"INT_SRC_M\", \"INT_THS_L_M\", \"INT_THS_H_M\", \"OFFSET_X_L_M\", \"OFFSET_X_H_M\", \"OFFSET_Y_L_M\", \"OFFSET_Y_H_M\", \"OFFSET_Z_L_M\", \"OFFSET_Z_H_M\", \"REFERENCE_X\", \"REFERENCE_Y\", \"REFERENCE_Z\", \"CTRL_REG0\", \"CTRL_REG1\", \"CTRL_REG2\", \"CTRL_REG3\", \"CTRL_REG4\", \"CTRL_REG5\", \"CTRL_REG6\", \"CTRL_REG7\", \"STATUS_REG_A\", \"OUT_X_L_A\", \"OUT_X_H_A\", \"OUT_Y_L_A\", \"OUT_Y_H_A\", \"OUT_Z_L_A\", \"OUT_Z_H_A\", \"FIFO_CTRL\", \"FIFO_SRC\", \"IG_CFG1\", \"IG_SRC1\", \"IG_THS1\", \"IG_DUR1\", \"IG_CFG2\", \"IG_SRC2\", \"IG_THS2\", \"IG_DUR2\", \"CLICK_CFG\", \"CLICK_SRC\", \"CLICK_THS\", \"TIME_LIMIT\", \"TIME_LATENCY\", \"TIME_WINDOW\", \"ACT_THS\", \"ACT_DUR\", \"MAG_SCALE_2\", \"MAG_SCALE_4\", \"MAG_SCALE_8\", \"MAG_SCALE_12\"];\nvar commands = [0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0F, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1A, 0x1B, 0x1C, 0x1D, 0x1E, 0x1F, 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2A, 0x2B, 0x2C, 0x2D, 0x2E, 0x2F, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3A, 0x3B, 0x3C, 0x3D, 0x3E, 0x3F, 0x00, 0x20, 0x40, 0x60];\nvar command = {};\nfor (var i = 0; i < commands.length; i++){\n command.merge({[topics[i]] : commands[i]});\n}\ndelete command.merge;\nmsg.commands = command;\ncommand = undefined;\n\nreturn msg;\n\n//Source: https://github.com/Seeed-Studio/Grove_6Axis_Accelerometer_And_Compass_v2/blob/master/LSM303D.cpp\n\n\n\n","outputs":1,"noerr":0,"x":390,"y":60,"wires":[["d7a70f0e.5cc8c"]]},{"id":"a241035d.3a4dd","type":"i2c scan","z":"19d59705.d99769","name":"","x":800,"y":100,"wires":[[],[]]},{"id":"6acb8b96.c84434","type":"i2c in","z":"19d59705.d99769","name":"","address":"30","command":"","count":"1","x":790,"y":140,"wires":[[]]},{"id":"2803532.52415ac","type":"function","z":"19d59705.d99769","name":"name commands","func":"Object.prototype.merge = function(obj2) {\n for (var attrname in obj2) {\n this[attrname] = obj2[attrname];\n }\n //Returning this is optional and certainly up to your implementation. \n //It allows for nice method chaining.\n return this;\n};\n\nvar topics = [\"TEMP_OUT_L\", \"TEMP_OUT_H\", \"STATUS_REG_M\", \"OUT_X_L_M\", \n\"OUT_X_H_M\", \"OUT_Y_L_M\", \"OUT_Y_H_M\", \"OUT_Z_L_M\", \"OUT_Z_H_M\", \"WHO_AM_I\", \n\"INT_CTRL_M\", \"INT_SRC_M\", \"INT_THS_L_M\", \"INT_THS_H_M\", \"OFFSET_X_L_M\", \n\"OFFSET_X_H_M\", \"OFFSET_Y_L_M\", \"OFFSET_Y_H_M\", \"OFFSET_Z_L_M\", \"OFFSET_Z_H_M\", \n\"REFERENCE_X\", \"REFERENCE_Y\", \"REFERENCE_Z\", \"CTRL_REG0\", \"CTRL_REG1\", \"CTRL_REG2\", \n\"CTRL_REG3\", \"CTRL_REG4\", \"CTRL_REG5\", \"CTRL_REG6\", \"CTRL_REG7\", \"STATUS_REG_A\", \n\"OUT_X_L_A\", \"OUT_X_H_A\", \"OUT_Y_L_A\", \"OUT_Y_H_A\", \"OUT_Z_L_A\",

\"OUT_Z_H_A\", \n\"FIFO_CTRL\", \"FIFO_SRC\", \"IG_CFG1\", \"IG_SRC1\", \"IG_THS1\", \"IG_DUR1\", \"IG_CFG2\", \n\"IG_SRC2\", \"IG_THS2\", \"IG_DUR2\", \"CLICK_CFG\", \"CLICK_SRC\", \"CLICK_THS\", \"TIME_LIMIT\", \n\"TIME_LATENCY\", \"TIME_WINDOW\", \"ACT_THS\", \"ACT_DUR\", \"MAG_SCALE_2\", \"MAG_SCALE_4\", \n\"MAG_SCALE_8\", \"MAG_SCALE_12\"];\nvar commands = [0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0F, 0x12, 0x13,\n0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1A, 0x1B, 0x1C, 0x1D, 0x1E, 0x1F, 0x20, 0x21, 0x22,\n0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2A, 0x2B, 0x2C, 0x2D, 0x2E, 0x2F, 0x30, 0x31,\n0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3A, 0x3B, 0x3C, 0x3D, 0x3E, 0x3F, 0x00,\n0x20, 0x40, 0x60];\nvar command = {};\nfor (var i = 0; i < commands.length; i++){\n command.merge({[topics[i]] : commands[i]});\n}\ndelete command.merge;\nmsg.commands = command;\ncommand = undefined;\n\nreturn msg;\n\n\n\n\n","outputs":1,"noerr":0,"x":390,"y":140,"wires":[["a7080e12.0646c"]]},{"id":"a7080e12.0646c","type":"function","z":"19d59705.d99769","name":"I2C WHO_AM_I","func":"msg.command = msg.commands.WHO_AM_I;\nnode.send(msg);","outputs":1,"noerr":0,"x":620,"y":140,"wires":[["6acb8b96.c84434"]]},{"id":"a39c51d.21522b","type":"join","z":"19d59705.d99769","name":"","mode":"custom","build":"object","property":"payload","propertyType":"msg","key":"topic","joiner":"\\n","joinerType":"str","accumulate":false,"timeout":"","count":"3","reduceRight":false,"reduceExp":"","reduceInit":"","reduceInitType":"","reduceFixup":"","x":350,"y":220,"wires":[["73a55fd6.2c55a"]]},{"id":"73a55fd6.2c55a","type":"function","z":"19d59705.d99769","name":"MAG XY -> HEADING DEGREE","func":"//Calculate the real Gauss value for the X and Y \n//axes from the amount of LSBs returned where \n//the LSB value by default is 0.48828125 mG\n//, resulting in 2048 LSBs per Gauss.\n//xGaussData = xDataLSB∗0.48828125 mG\n//yGaussData = yDataLSB∗0.48828125 mG\n\n//https://blog.digilentinc.com/how-to-convert-magnetometer-data-into-compass-heading/\n\n\nvar value;\nvalue =Math.atan2(msg.payload.OUT_Y_M, msg.payload.OUT_X_M) * 180 / Math.PI;\n\nif(value < 0)\n{\nvalue = (value + 360);\n}\nmsg.payload=value.toFixed(2);\n\nglobal.set('Magheading',msg.payload);\nreturn msg;","outputs":1,"noerr":0,"x":550,"y":220,"wires":[["232b5494.f07e8c"]]},{"id":"30f1d35c.935cdc","type":"function","z":"b873dccc.a9864","name":"NMEA -> Different Payloads","func":"\n//Different variables. One msg for each output.\nvar msg1;\nvar msg2;\nvar msg3;\nvar msg4;\nvar msg5;\nvar msg6;\nvar msg7;\nvar msg8;\nvar msg9;\n\n//If right NMEA string\nif (msg.payload.includes('GPZDA,'))\n{\n//Split string into right size. The characters from 7-16 represents the time value\nlet TIME = msg.payload.slice(7,16);\n//The parseFloat() function parses a string and returns a floating point number.\nvar numberValue = parseFloat(TIME);\ncontext.set(TIME);\n\n//The payload will be set to the value of numberValue (Time). Topic will just be time. \nmsg1= {payload: numberValue, topic: \"Time\"};\n//Will set the global variabel time to the value of numbervalue\nglobal.set('Time',numberValue);\n\n}\n\n//All of

the different else if statement contain the same structure. \n//The string gets cut to size, then passed along as global variable\n\n//Windspeed & Wind Angle\nelse if (msg.payload.includes('$WIMWV,'))\n{\nlet SPEED = msg.payload.slice(15,20);\nvar numberValue1 = parseFloat(SPEED);\nmsg2= {payload: numberValue1, topic: \"WindSpeed\"};\nglobal.set('WindSpeed',numberValue1);\n\nlet ANGLE = msg.payload.slice(7,12);\nvar numberValue2 = parseFloat(ANGLE);\nmsg3= {payload: numberValue2, topic: \"WindAngle\"};\nglobal.set('WindAngle',numberValue2);\n}\n\n//Latitude & Longitude\nelse if (msg.payload.includes('$GPGGA,'))\n{\nlet Lat = msg.payload.slice(17, 27);\nlet Lat1 = Lat.slice(0,2);\nlet Lat2 = Lat.slice(2,9);\nLat1 = parseFloat(Lat1);\nLat2 = parseFloat(Lat2);\nLat = Lat1 + Lat2/60;\n\nvar numberValue3 = parseFloat(Lat);\nmsg4= {payload:numberValue3 , topic: \"Lat\"};\nglobal.set('Lat',numberValue3);\n\nlet Lon = msg.payload.slice(29, 39);\nlet Lon1 = Lon.slice(0,3);\nlet Lon2 = Lon.slice(3,10);\nLon1 = parseFloat(Lon1);\nLon2 = parseFloat(Lon2);\nLon = Lon1 + Lon2/60;\n\nvar numberValue4 = parseFloat(Lon);\nmsg5= {payload: numberValue4, topic: \"Lon\"};\nglobal.set('Lon',numberValue4);\n}\n\n//Speed & course over ground\nelse if (msg.payload.includes('$GPVTG,'))\n{\nlet Knot = msg.payload.slice(24, 29);\nvar numberValue5 = parseFloat(Knot);\nmsg6= {payload:numberValue5, topic: \"Knots\"};\nglobal.set('Knot',numberValue5);\n\nlet COG = msg.payload.slice(7, 13);\nvar numberValue6 = parseFloat(COG);\nmsg7= {payload: numberValue6, topic: \"COG\"};\nglobal.set('Cog',numberValue6);\n}\n\n//Trueheading\nelse if (msg.payload.includes('$HETHS,'))\n{\nlet TRHD = msg.payload.slice(7, 13);\nvar numberValue7 = parseFloat(TRHD);\nmsg8= {payload: numberValue7, topic: \"TRUEHEADING\"};\nglobal.set('TrueHeading',numberValue7);\n}\n\n//Rate of Turn\nelse if (msg.payload.includes('$HEROT,'))\n{\nlet ROT = msg.payload.slice(7, 13);\nvar numberValue8 = parseFloat(ROT);\nmsg9= {payload: numberValue8, topic: \"Rot\"};\nglobal.set('Rot',numberValue8);\n\n}\n//Each message with different payload and topic is given as an output\nreturn [msg1,msg2,msg3,msg4,msg5,msg6,msg7,msg8,msg9]","outputs":9,"noerr":0,"x":480,"y":340,"wires":[["257aeb91.2701c4"],["79aaec3c.bd9e54"],["277e0239.1748ce"],["7ac69b12.a4c994"],["a279b627.f871e8"],["c042823c.7505a"],["51b09ce4.9a6194"],["494f75a7.07a3cc"],["fcb7d710.361e88"]]},{"id":"84c812e5.7ab61","type":"comment","z":"b873dccc.a9864","name":"Deler de ulike NMEA-telegrammene & Setter de ulike variablene til globale variabler","info":"","x":330,"y":440,"wires":[]},{"id":"21b514d2.7f353c","type":"comment","z":"b873dccc.a9864","name":"Leser av NMEA-strenger fra bro","info":"","x":770,"y":20,"wires":[]},{"id":"25c25e9e.f5f7e2","type":"file in","z":"607c7664.d3f748","name":"","filename":"/home/pi/share/example","format":"lines","chunk":

false,"sendError":true,"encoding":"none","x":610,"y":120,"wires":[["bf5eb2cd.9006b"]]},{"id":"bf5eb2cd.9006b","type":"xml-converter","z":"607c7664.d3f748","name":"","ignorenamespace":false,"autoinline":false,"multithread":false,"x":820,"y":120,"wires":[["8c860c1b.a1c24","af90e4e5.730cd8","673499f0.252938","d3483b60.90f508","1b359195.0235be"]]},{"id":"bfd2144a.dd98d8","type":"function","z":"607c7664.d3f748","name":"Waypoint ->Global variables","func":"\nvar msg1 = { payload: msg.payload.WayPoint.Id };\nvar msg2 = { payload: msg.payload.WayPoint.WPName };\nvar msg3 = { payload: msg.payload.WayPoint.Lat };\nvar msg4 = { payload: msg.payload.WayPoint.Lon };\nvar msg5 = { payload: msg.payload.WayPoint.TurnRadius };\nvar msg6 = { payload: msg.payload.WayPoint.LegType };\nvar msg7 = { payload: msg.payload.WayPoint.XTE };\nvar msg8 = { payload: msg.payload.WayPoint.Speed };\nvar msg9 = { payload: msg.payload.WayPoint.Danger};\nvar msg10 = { payload: msg.payload.WayPoint.Message };\nvar msg11 = { payload: msg.payload.WayPoint.RTime };\n\n//Relevant information -> global variables with function\nglobal.set\nglobal.set('Waypointid',msg.payload.WayPoint.Id);\nglobal.set('Waypointname',msg.payload.WayPoint.WPName);\nglobal.set('Waypointlat',msg.payload.WayPoint.Lat);\nglobal.set('Waypointlon',msg.payload.WayPoint.Lon);\nglobal.set('Waypointturnradius',msg.payload.WayPoint.TurnRadius);\nglobal.set('Waypointlegtype',msg.payload.WayPoint.LegType);\nglobal.set('Waypointxte',msg.payload.WayPoint.XTE);\nglobal.set('Waypointspeed',msg.payload.WayPoint.Speed);\nglobal.set('Waypointdanger',msg.payload.WayPoint.Danger);\nglobal.set('Waypointmessage',msg.payload.WayPoint.Message);\nglobal.set('Waypointrtime',msg.payload.WayPoint.RTime);\n\n//Different outputs\nreturn [msg1,msg2,msg3,msg4,msg5,msg6,msg7,msg8,msg9,msg10,msg11]","outputs":11,"noerr":0,"x":1220,"y":120,"wires":[[],["1866d832.bc96d8"],["c72fa6c1.6f4318"],["69747b6d.115f94"],["b891b02d.dbe3b"],["e19ce261.10e71"],["d45f651a.cfd868"],["68a0c0bb.5e196"],["9dca1442.ce14e8"],["9cd804d3.c87d68"],[]]},{"id":"1866d832.bc96d8","type":"ui_text","z":"607c7664.d3f748","group":"e7285da5.dfd63","order":1,"width":0,"height":0,"name":"","label":"WPName","format":"{{msg.payload}}","layout":"row-spread","x":1460,"y":60,"wires":[]},{"id":"c72fa6c1.6f4318","type":"ui_text","z":"607c7664.d3f748","group":"e7285da5.dfd63","order":2,"width":0,"height":0,"name":"","label":"Lat","format":"{{msg.payload}}","layout":"row-spread","x":1450,"y":100,"wires":[]},{"id":"69747b6d.115f94","type":"ui_text","z":"607c7664.d3f748","group":"e7285da5.dfd63","order":3,"width":0,"height":0,"name":"","label":"Lon","format":"{{msg.payload}}","layout":"row-spread","x":1450,"y":140,"wires":[]},{"id":"b891b02d.dbe3b","type":"ui_text","z":"607c7664.d3f748","group":"e7285da5.dfd63","order":4,"width":0,"height":0,"name":"","label":"TurnRadius","format":"{{msg.payload}}","layout":"row-

spread","x":1470,"y":180,"wires":[]},{"id":"e19ce261.10e71","type":"ui_text","z":"607c7664.d3f748
","group":"e7285da5.dfd63","order":5,"width":0,"height":0,"name":"","label":"LegType","format":"{
{msg.payload}}","layout":"row-
spread","x":1460,"y":220,"wires":[]},{"id":"d45f651a.cfd868","type":"ui_text","z":"607c7664.d3f748
","group":"e7285da5.dfd63","order":6,"width":0,"height":0,"name":"","label":"XTE","format":"{{ms
g.payload}}","layout":"row-
spread","x":1450,"y":260,"wires":[]},{"id":"68a0c0bb.5e196","type":"ui_text","z":"607c7664.d3f748
","group":"e7285da5.dfd63","order":7,"width":0,"height":0,"name":"","label":"Speed","format":"{{m
sg.payload}}","layout":"row-
spread","x":1450,"y":300,"wires":[]},{"id":"9dca1442.ce14e8","type":"ui_text","z":"607c7664.d3f74
8","group":"e7285da5.dfd63","order":8,"width":0,"height":0,"name":"","label":"Danger","format":"{
{msg.payload}}","layout":"row-
spread","x":1460,"y":340,"wires":[]},{"id":"9cd804d3.c87d68","type":"ui_text","z":"607c7664.d3f74
8","group":"e7285da5.dfd63","order":9,"width":0,"height":0,"name":"","label":"Message","format":"
{{msg.payload}}","layout":"row-
spread","x":1460,"y":380,"wires":[]},{"id":"9342084a.c559d8","type":"ui_button","z":"607c7664.d3f
748","name":"Start
seilas","group":"e7285da5.dfd63","order":11,"width":0,"height":0,"passthru":false,"label":"Start
seilas","tooltip":"Viser et waypoint per
sekund","color":"","bgcolor":"","icon":"","payload":"true","payloadType":"bool","topic":"","x":130,"
y":80,"wires":[["25c25e9e.f5f7e2","bba04bce.c0d188"]]},{"id":"ed8ef8a9.07ddd8","type":"comment
","z":"607c7664.d3f748","name":"Knapp i dashbord som starter fremvisningen av
waypoints","info":"","x":250,"y":20,"wires":[]},{"id":"12d1c4d9.2744fb","type":"comment","z":"607
c7664.d3f748","name":"Henter waypoint-data fra
fil","info":"","x":390,"y":60,"wires":[]},{"id":"a22db0bb.43b91","type":"comment","z":"607c7664.d3
f748","name":"Tolker filen som
XML","info":"","x":630,"y":60,"wires":[]},{"id":"2a75af35.e8905","type":"comment","z":"607c7664
.d3f748","name":"Filtrerer meldingen til flere
outputs","info":"","x":1360,"y":17,"wires":[]},{"id":"b756f47e.6608a8","type":"function","z":"607c7
664.d3f748","name":"increment","func":"msg.increment = 1;\nreturn
msg;","outputs":1,"noerr":0,"x":400,"y":200,"wires":[["a04cda7b.3f5238"]]},{"id":"a04cda7b.3f5238
","type":"counter","z":"607c7664.d3f748","name":"","init":"2","step":1,"lower":"","upper":"","mode"
:"increment","outputs":"1","x":560,"y":200,"wires":[["f4491e42.c767b","32239496.a7517c","ccda90.
7032a57"]]},{"id":"755876c5.0f4e48","type":"ui_button","z":"607c7664.d3f748","name":"","group":
"e7285da5.dfd63","order":12,"width":0,"height":0,"passthru":false,"label":"Neste","tooltip":"","color
":"","bgcolor":"","icon":"","payload":"","payloadType":"date","topic":"","x":150,"y":120,"wires":[["b

756f47e.6608a8","25c25e9e.f5f7e2"]]},{"id":"f4491e42.c767b","type":"function","z":"607c7664.d3f748","name":"global.set('count',msg.count);","func":"global.set('count',msg.count);","outputs":1,"noerr":0,"x":780,"y":200,"wires":[[]]},{"id":"8c860c1b.a1c24","type":"function","z":"607c7664.d3f748","name":"wp==selected","func":"//Selected waypoint. Uses 'counter' function to increment\nvar selected = global.get('count') || 0;\n\n//Gets the waypointnumber from the RUX file \nvar wp = Number(msg.payload.WayPoint.Id);\n\n//checks if the waypointnumber is the same as the counted waypoint\nif (wp==selected)\nreturn msg;\n\n\n\n\n\n\n\n\n\n\n\n\n","outputs":1,"noerr":0,"x":1000,"y":120,"wires":[["bfd2144a.dd98d8"]]},{"id":"32239496.a7517c","type":"ui_text","z":"607c7664.d3f748","group":"e7285da5.dfd63","order":10,"width":0,"height":0,"name":"","label":"Selected Waypoint","format":"{{msg.count}}","layout":"row-spread","x":750,"y":240,"wires":[]},{"id":"5c06dfc5.ea294","type":"function","z":"607c7664.d3f748","name":"decrement","func":"msg.decrement = 1;\nreturn msg;","outputs":1,"noerr":0,"x":410,"y":240,"wires":[["a04cda7b.3f5238"]]},{"id":"526a5079.4bee","type":"ui_button","z":"607c7664.d3f748","name":"","group":"e7285da5.dfd63","order":13,"width":0,"height":0,"passthru":false,"label":"Forrige","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"date","topic":"","x":140,"y":160,"wires":[["25c25e9e.f5f7e2","5c06dfc5.ea294"]]},{"id":"bba04bce.c0d188","type":"function","z":"607c7664.d3f748","name":"reset","func":"msg.reset = true\nreturn msg;","outputs":1,"noerr":0,"x":390,"y":160,"wires":[["a04cda7b.3f5238"]]},{"id":"b461eebd.0d6b3","type":"comment","z":"607c7664.d3f748","name":"Sender kun videre waypoint med valgt Id","info":"","x":920,"y":80,"wires":[]},{"id":"680eaed0.360a3","type":"function","z":"607c7664.d3f748","name":"Voice increment/decrement","func":"// This function reads the variable which enables\n// the user to control the waypoints with voice\n\nif (global.get(\"Voicewaypoint\")==\"Next\"){\n   global.set(\"Voicewaypoint\",\"Done\");\n   node.warn(\"Voice next\");\n   msg.increment = 1;\n   return msg;\n}\n\nif (global.get(\"Voicewaypoint\")==\"Previous\"){\n   global.set(\"Voicewaypoint\",\"Done\");\n   node.warn(\"Voice previous\");\n   msg.decrement = 1;\n   return msg;\n}\n\n\n","outputs":1,"noerr":0,"x":420,"y":360,"wires":[["a04cda7b.3f5238","25c25e9e.f5f7e2"]]},{"id":"b6a1c6d5.e72f48","type":"inject","z":"607c7664.d3f748","name":"Update","topic":"","payload":"","payloadType":"date","repeat":"0.1","crontab":"","once":true,"onceDelay":"","x":160,"y":460,"wires":[["680eaed0.360a3","a0de2d8c.6d367","4fbafbcb.f777b4","4fc60696.2737f8","a0f1969b.7e37a8","b61167d4.c17668","db501928.4f9468"]]},{"id":"7b3dc34b.aa4a8c","type":"comment","z":"607c7664.d3f748","name":"Leser av variabel for stemmestyring av waypoint","info":"","x":320,"y":320,"wires":[]},{"id":"c6f12417.139d88","type":"comment","z":"607c7664.d3f748","name":"Styrer waypoint fra UI / Hololens","info":"","x":830,"y":20,"wires":[]},{"id":"ad5ecb8a.7d2348","type":"inject","z":"19d5970

5.d99769","name":"WhoAmI","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":true,"onceDelay":0.1,"x":100,"y":140,"wires":[["2803532.52415ac"]]},{"id":"5133cfc5.ff4f7","type":"inject","z":"19d59705.d99769","name":"SCAN I2C","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":true,"onceDelay":0.1,"x":110,"y":100,"wires":[["a241035d.3a4dd"]]},{"id":"f450858d.4588f8","type":"inject","z":"19d59705.d99769","name":"INJECT COMMANDS","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":true,"onceDelay":0.1,"x":140,"y":60,"wires":[["e56b7f1a.03e24"]]},{"id":"a0de2d8c.6d367","type":"function","z":"607c7664.d3f748","name":"DistanceToWaypoint // TimeToWaypoint","func":"//Function to convert value into radians\nNumber.prototype.toRad = function() {\n return this * Math.PI / 180;\n}\n\n\n//Distance between waypoint and the vessel\n//based on the Haversine Formula: https://www.movable-type.co.uk/scripts/latlong.html\nvar lat1 = parseFloat(global.get(\"Lat\")); //Vessel Latitude\nvar lon1 = parseFloat(global.get(\"Lon\")); //Vessel Longitude\nvar lat2 = parseFloat(global.get(\"Waypointlat\")); // Waypoint Latitude\nvar lon2 = parseFloat(global.get(\"Waypointlon\")); // Waypoint Longitude\n\nvar R = 6371; // Earth Radius in km \nvar x1 = lat2-lat1;\nvar dLat = x1.toRad(); \nvar x2 = lon2-lon1;\nvar dLon = x2.toRad(); \nvar a = Math.sin(dLat/2) * Math.sin(dLat/2) + \n Math.cos(lat1.toRad()) * Math.cos(lat2.toRad()) * \n Math.sin(dLon/2) * Math.sin(dLon/2); \nvar c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a)); \n//The distance between the vessel and the waypoint\n//divided by 1.852 to get it in nautical miles\nvar d = (R * c/1.852); \n\n//var offset = global.get(\"Turnacc\")*0.000539957 + parseFloat(global.get(\"Waypointturnradius\"))*Math.abs(Math.tan((parseFloat((global.get(\"Nextbearing\"))-parseFloat(global.get(\"Bearing\")).toRad()))*0.5));\n//offset = offset.toFixed(3);\nvar offset = global.get(\"Turnacc\");\nd = (d - offset).toFixed(2);\nglobal.set(\"Distancetowaypointtwodecimal\",d);\nd = (d - offset).toFixed(2);\nglobal.set(\"Distancetowaypoint\",d);\n\nvar speed = global.get(\"Knot\");\nvar decimalTimeString = (d/speed);\nvar decimalTime = parseFloat(decimalTimeString);\ndecimalTime = decimalTime * 60 * 60;\n//var hours = Math.floor((decimalTime / (60 * 60)));\n//decimalTime = decimalTime - (hours * 60 * 60);\nvar minutes = Math.floor((decimalTime / 60));\ndecimalTime = decimalTime - (minutes * 60);\nvar seconds = Math.round(decimalTime);\n/*\nif(hours < 10)\n{\n\thours = \"0\" + hours;\n}*/\nif(minutes < 10)\n{\n\tminutes = \"0\" + minutes;\n}\nif(seconds < 10)\n{\n\tseconds = \"0\" + seconds;\n}\nvar Timetowaypoint = (\"\"+ minutes + \"m\" + seconds+\"\");\nif (Timetowaypoint == \"InfinitymNaN\")\nglobal.set(\"Timetowaypoint\",\"99m99\");\nelse\nglobal.set(\"Timetowaypoint\", Timetowaypoint);\n\nvar msg1 = { payload: d};\nvar msg2 = { payload: Timetowaypoint};\nvar msg3 = { payload: offset};\n\n\nreturn

[msg1,msg2,msg3];","outputs":3,"noerr":0,"x":460,"y":480,"wires":[["ea0ef1ee.3ee43"],["cd916d89.72229"],["47976610.092858"]]},{"id":"e441a38c.919f1","type":"function","z":"607c7664.d3f748","name":"Waypoint ->Global variables","func":"var msg1 = { payload: msg.payload.WayPoint.Id };\nvar msg2 = { payload: msg.payload.WayPoint.WPName };\nvar msg3 = { payload: msg.payload.WayPoint.Lat };\nvar msg4 = { payload: msg.payload.WayPoint.Lon };\nvar msg5 = { payload: msg.payload.WayPoint.TurnRadius };\nvar msg6 = { payload: msg.payload.WayPoint.LegType };\nvar msg7 = { payload: msg.payload.WayPoint.XTE };\nvar msg8 = { payload: msg.payload.WayPoint.Speed };\nvar msg9 = { payload: msg.payload.WayPoint.Danger};\nvar msg10 = { payload: msg.payload.WayPoint.Message };\nvar msg11 = { payload: msg.payload.WayPoint.RTime };\n\nglobal.set('Nextwaypointid',msg.payload.WayPoint.Id);\nglobal.set('Nextwaypointname',msg.payload.WayPoint.WPName);\nglobal.set('Nextwaypointlat',msg.payload.WayPoint.Lat);\nglobal.set('Nextwaypointlon',msg.payload.WayPoint.Lon);\nglobal.set('Nextwaypointturnradius',msg.payload.WayPoint.TurnRadius);\nglobal.set('Nextwaypointlegtype',msg.payload.WayPoint.LegType);\nglobal.set('Nextwaypointxte',msg.payload.WayPoint.XTE);\nglobal.set('Nextwaypointspeed',msg.payload.WayPoint.Speed);\nglobal.set('Nextwaypointdanger',msg.payload.WayPoint.Danger);\nglobal.set('Nextwaypointmessage',msg.payload.WayPoint.Message);\nglobal.set('Nextwaypointrtime',msg.payload.WayPoint.RTime);\n\n","outputs":1,"noerr":0,"x":1220,"y":240,"wires":[[]]},{"id":"af90e4e5.730cd8","type":"function","z":"607c7664.d3f748","name":"wp==selected+1","func":"var selected = global.get('count') || 0;\nvar wp = Number(msg.payload.WayPoint.Id);\nif (wp==(selected+1))\nreturn msg;","outputs":1,"noerr":0,"x":1000,"y":240,"wires":[["e441a38c.919f1"]]},{"id":"4fbafbcb.f777b4","type":"function","z":"607c7664.d3f748","name":"Bearing","func":"start_latitude = parseFloat(global.get(\"Prevwaypointlat\"))*Math.PI/180;\nstart_longitude = parseFloat(global.get(\"Prevwaypointlon\"))*Math.PI/180;\nstop_latitude   = parseFloat(global.get(\"Waypointlat\"))*Math.PI/180;\nstop_longitude = parseFloat(global.get(\"Waypointlon\"))*Math.PI/180;\n\nvar y = Math.sin(stop_longitude-start_longitude) * Math.cos(stop_latitude);\nvar x = Math.cos(start_latitude)*Math.sin(stop_latitude)-Math.sin(start_latitude)*Math.cos(stop_latitude)*Math.cos(stop_longitude-start_longitude);\nvar Bearing = Math.atan2(y, x) * 180 / Math.PI;\nBearing = (Bearing + 360) % 360\nBearing = Bearing.toFixed(0);\nglobal.set(\"Bearing\",Bearing);\nmsg.payload = Bearing;\nreturn msg;","outputs":1,"noerr":0,"x":360,"y":520,"wires":[["cfdd3adf.c64408"]]},{"id":"4fc60696.2737f8","type":"function","z":"607c7664.d3f748","name":"XTE","func":"Number.prototype.toRad = function() {\n   return this * Math.PI / 180;\n}\n\nstart_latitude = parseFloat(global.get(\"Prevwaypointlat\"));\nstart_longitude = parseFloat(global.get(\"Prevwaypointlon\"));\nstop_latitude   = parseFloat(global.get(\"Waypointlat\"));\nstop_longitude = parseFloat(global.get(\"Waypointlon\"));\ncurrentlat = parseFloat(global.get(\"Lat\"));\ncurrentlon =

parseFloat(global.get(\"Lon\"));\n\nbrng = global.get(\"Bearing\");\n\n\n\n\nvar y = Math.sin(currentlon-start_longitude) * Math.cos(stop_latitude);\nvar x = Math.cos(start_latitude)*Math.sin(currentlat) -\n Math.sin(start_latitude)*Math.cos(currentlat)*Math.cos(currentlon-start_longitude);\nvar crs_AD = Math.atan2(y, x) * 180 / Math.PI;\ncrs_AD = (crs_AD + 360) % 360\n\nvar z = Math.sin(stop_longitude-start_longitude) * Math.cos(stop_latitude);\nvar w = Math.cos(start_latitude)*Math.sin(stop_latitude) -\n Math.sin(start_latitude)*Math.cos(stop_latitude)*Math.cos(stop_longitude-start_longitude);\nvar crs_AB = Math.atan2(z, w) * 180 / Math.PI;\ncrs_AB = (crs_AB + 360) % 360\n\n\n\n\n\n\n\n//(positive XTE means right of course, negative means left)\n//Der A er der man seiler fra, B er dit man skal, D er der man er\n\nvar R = 6371; // km \n\nvar x1 = currentlon-start_latitude;\nvar dLat = x1.toRad(); \nvar x2 = currentlon-start_longitude;\nvar dLon = x2.toRad(); \nvar a = Math.sin(dLat/2) * Math.sin(dLat/2) + \n Math.cos(start_latitude.toRad()) * Math.cos(currentlon.toRad()) * \n Math.sin(dLon/2) * Math.sin(dLon/2); \nvar c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a)); \n\ndist_AD = (R * c/1.852).toFixed(2); \n\n\nXTE =Math.asin(Math.sin(dist_AD)*Math.sin(crs_AD-crs_AB)).toFixed(2);\nglobal.set(\"Xte\",XTE);\nmsg.payload=XTE;\nreturn msg;","outputs":1,"noerr":0,"x":350,"y":640,"wires":[["db70a55.d5be358"]]},{"id":"9864c5.252b5b38","type":"ui_button","z":"fd97e6d8.2cb9a8","name":"","group":"50eda971.220498","order":7,"width":0,"height":0,"passthru":false,"label":"Towards","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Towards","payloadType":"str","topic":"","x":140,"y":520,"wires":[["45184f20.e3f3f"]]},{"id":"9caaee95.440e9","type":"ui_button","z":"fd97e6d8.2cb9a8","name":"","group":"50eda971.220498","order":8,"width":0,"height":0,"passthru":false,"label":"Away","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Away","payloadType":"str","topic":"","x":150,"y":560,"wires":[["45184f20.e3f3f"]]},{"id":"36e3f3f.b68860c","type":"ui_button","z":"fd97e6d8.2cb9a8","name":"","group":"50eda971.220498","order":4,"width":0,"height":0,"passthru":false,"label":"Down","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Down","payloadType":"str","topic":"","x":150,"y":400,"wires":[["45184f20.e3f3f"]]},{"id":"98ab2c0f.46b95","type":"ui_button","z":"fd97e6d8.2cb9a8","name":"","group":"50eda971.220498","order":3,"width":0,"height":0,"passthru":false,"label":"Up","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Up","payloadType":"str","topic":"","x":150,"y":360,"wires":[["45184f20.e3f3f"]]},{"id":"df43b190.3b761","type":"function","z":"fd97e6d8.2cb9a8","name":"Choose element","func":"str = msg.payload.replace(\" \",\"\");\nglobal.set(\"Element\",str);\nreturn msg;\n","outputs":1,"noerr":0,"x":320,"y":160,"wires":[["dad7bc14.b217f"]]},{"id":"43300961.c8a7f8","type":"ui_slider","z":"fd97e6d8.2cb9a8","name":"Move by","label":"Move by","tooltip":"","group":"50eda971.220498","order":2,"width":0,"height":0,"passthru":true,"outs":"end","topic":"Moveby","min":"0","max":"15","step":"0.1","x":840,"y":560,"wires":[["4b131305.b01d9c

","e10a3d0d.f852b"]]},{"id":"605268a3.a61218","type":"ui_button","z":"fd97e6d8.2cb9a8","name":"
","group":"13958700.8d0729","order":16,"width":0,"height":0,"passthru":false,"label":"Current
waypoint","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Current
waypoint","payloadType":"str","topic":"","x":110,"y":120,"wires":[["df43b190.3b761"]]},{"id":"afaf
e22a.0326d","type":"ui_button","z":"fd97e6d8.2cb9a8","name":"","group":"13958700.8d0729","orde
r":16,"width":0,"height":0,"passthru":false,"label":"Next
waypoint","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Next
waypoint","payloadType":"str","topic":"","x":120,"y":160,"wires":[["df43b190.3b761"]]},{"id":"3b9
45fe9.97d1","type":"ui_button","z":"fd97e6d8.2cb9a8","name":"","group":"13958700.8d0729","order
":16,"width":0,"height":0,"passthru":false,"label":"Cog","tooltip":"","color":"","bgcolor":"","icon":"",
"payload":"Cog","payloadType":"str","topic":"","x":150,"y":200,"wires":[["df43b190.3b761"]]},{"id"
:"dad7bc14.b217f","type":"ui_text","z":"fd97e6d8.2cb9a8","group":"13958700.8d0729","order":3,"w
idth":0,"height":0,"name":"Chosen element","label":"Chosen
element","format":"{{msg.payload}}","layout":"row-
spread","x":500,"y":160,"wires":[]},{"id":"45184f20.e3f3f","type":"function","z":"fd97e6d8.2cb9a8",
"name":"Choose direction","func":"global.set(\"Direction\",msg.payload);\nreturn
msg;\n","outputs":1,"noerr":0,"x":350,"y":360,"wires":[["f601f440.d7dc38"]]},{"id":"f601f440.d7dc3
8","type":"function","z":"fd97e6d8.2cb9a8","name":"Move
now","func":"global.set(\"Movenow\",\"True\");\nreturn
msg;","outputs":1,"noerr":0,"x":530,"y":360,"wires":[[]]},{"id":"32167af9.5a1e76","type":"ui_button
","z":"fd97e6d8.2cb9a8","name":"","group":"50eda971.220498","order":6,"width":0,"height":0,"passt
hru":false,"label":"Right","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Right","payloadTy
pe":"str","topic":"","x":150,"y":480,"wires":[["45184f20.e3f3f"]]},{"id":"89bafa79.067958","type":"u
i_button","z":"fd97e6d8.2cb9a8","name":"","group":"50eda971.220498","order":5,"width":0,"height"
:0,"passthru":false,"label":"Left","tooltip":"","color":"","bgcolor":"","icon":"","payload":"Left","payl
oadType":"str","topic":"","x":150,"y":440,"wires":[["45184f20.e3f3f"]]},{"id":"b20e798d.cdafb8","ty
pe":"ui_button","z":"fd97e6d8.2cb9a8","name":"","group":"13958700.8d0729","order":16,"width":0,
"height":0,"passthru":false,"label":"All
elements","tooltip":"","color":"","bgcolor":"","icon":"","payload":"All
elements","payloadType":"str","topic":"","x":130,"y":240,"wires":[["df43b190.3b761"]]},{"id":"a0f1
969b.7e37a8","type":"function","z":"607c7664.d3f748","name":"Next bearing","func":"start_latitude
= parseFloat(global.get(\"Waypointlat\"))*Math.PI/180;\nstart_longitude =
parseFloat(global.get(\"Waypointlon\"))*Math.PI/180;\nstop_latitude =
parseFloat(global.get(\"Nextwaypointlat\"))*Math.PI/180;\nstop_longitude =
parseFloat(global.get(\"Nextwaypointlon\"))*Math.PI/180;\n\nvar y = Math.sin(stop_longitude-
start_longitude) * Math.cos(stop_latitude);\nvar x = Math.cos(start_latitude)*Math.sin(stop_latitude)
-\n Math.sin(start_latitude)*Math.cos(stop_latitude)*Math.cos(stop_longitude-

start_longitude);\nvar Bearing = Math.atan2(y, x) * 180 / Math.PI;\nBearing = (Bearing + 360) % 360\nBearing = Bearing.toFixed(0);\nglobal.set(\"Nextbearing\",Bearing);\nmsg.payload = Bearing;\nreturn msg;","outputs":1,"noerr":0,"x":370,"y":560,"wires":[["4d81439c.261c3c"]]},{"id":"32e711b3.70d3be","type":"function","z":"607c7664.d3f748","name":"Waypoint ->Global variables","func":"var msg1 = { payload: msg.payload.WayPoint.Id };\nvar msg2 = { payload: msg.payload.WayPoint.WPName };\nvar msg3 = { payload: msg.payload.WayPoint.Lat };\nvar msg4 = { payload: msg.payload.WayPoint.Lon };\nvar msg5 = { payload: msg.payload.WayPoint.TurnRadius };\nvar msg6 = { payload: msg.payload.WayPoint.LegType };\nvar msg7 = { payload: msg.payload.WayPoint.XTE };\nvar msg8 = { payload: msg.payload.WayPoint.Speed };\nvar msg9 = { payload: msg.payload.WayPoint.Danger};\nvar msg10 = { payload: msg.payload.WayPoint.Message };\nvar msg11 = { payload: msg.payload.WayPoint.RTime };\n\nglobal.set('Nextnextwaypointid',msg.payload.WayPoint.Id);\nglobal.set('Nextnextwaypointname',msg.payload.WayPoint.WPName);\nglobal.set('Nextnextwaypointlat',msg.payload.WayPoint.Lat);\nglobal.set('Nextnextwaypointlon',msg.payload.WayPoint.Lon);\nglobal.set('Nextnextwaypointturnradius',msg.payload.WayPoint.TurnRadius);\nglobal.set('Nextnextwaypointlegtype',msg.payload.WayPoint.LegType);\nglobal.set('Nextnextwaypointxte',msg.payload.WayPoint.XTE);\nglobal.set('Nextnextwaypointspeed',msg.payload.WayPoint.Speed);\nglobal.set('Nextnextwaypointdanger',msg.payload.WayPoint.Danger);\nglobal.set('Nextnextwaypointmessage',msg.payload.WayPoint.Message);\nglobal.set('Nextnextwaypointrtime',msg.payload.WayPoint.RTime);\n\nreturn [msg1,msg2,msg3,msg4,msg5,msg6,msg7,msg8,msg9,msg10,msg11]","outputs":1,"noerr":0,"x":1220,"y":280,"wires":[[]]},{"id":"673499f0.252938","type":"function","z":"607c7664.d3f748","name":"wp==selected+2","func":"var selected = global.get('count') || 0;\nvar wp = Number(msg.payload.WayPoint.Id);\nif (wp==(selected+2))\nreturn msg;","outputs":1,"noerr":0,"x":1000,"y":280,"wires":[["32e711b3.70d3be"]]},{"id":"3da1c30a.0f451c","type":"comment","z":"fd97e6d8.2cb9a8","name":"Manipulates the UI-elements in real time","info":"","x":570,"y":40,"wires":[]},{"id":"103f1cea.27bf43","type":"comment","z":"fd97e6d8.2cb9a8","name":"Define how many units to move","info":"","x":770,"y":520,"wires":[]},{"id":"731cb3d.9037c4c","type":"comment","z":"fd97e6d8.2cb9a8","name":"Define direction and execute once","info":"","x":240,"y":320,"wires":[]},{"id":"eca44e3d.8d8d2","type":"comment","z":"fd97e6d8.2cb9a8","name":"Choose which element to move","info":"","x":150,"y":80,"wires":[]},{"id":"22e6c199.0153ce","type":"comment","z":"7a85299b.1bc6e8","name":"Waiting for connections at port 1025","info":"","x":160,"y":20,"wires":[]},{"id":"cd2cd6bb.6519e8","type":"comment","z":"7a85299b.1bc6e8","name":"Parses the buffer to

string","info":"","x":350,"y":60,"wires":[]},{"id":"f03be58c.a47178","type":"comment","z":"7a85299b.1bc6e8","name":"Writes to variables and sends their values","info":"","x":580,"y":20,"wires":[]},{"id":"c7481023.855d7","type":"comment","z":"7a85299b.1bc6e8","name":"For some odd reason we need to add a line","info":"","x":780,"y":140,"wires":[]},{"id":"43d71ff0.bfc69","type":"comment","z":"7a85299b.1bc6e8","name":"Sends the output string to the connected Hololens","info":"","x":920,"y":60,"wires":[]},{"id":"b61167d4.c17668","type":"function","z":"607c7664.d3f748","name":"Distance to nextwaypoint","func":"Number.prototype.toRad = function() {\n return this * Math.PI / 180;\n}\n\nvar lat1 = parseFloat(global.get(\"Waypointlat\"));\nvar lon1 = parseFloat(global.get(\"Waypointlon\"));\nvar lat2 = parseFloat(global.get(\"Nextwaypointlat\"));\nvar lon2 = parseFloat(global.get(\"Nextwaypointlon\"));\n\nvar R = 6371; // Earth Radius in km \nvar x1 = lat2-lat1;\nvar dLat = x1.toRad(); \nvar x2 = lon2-lon1;\nvar dLon = x2.toRad(); \nvar a = Math.sin(dLat/2) * Math.sin(dLat/2) + \n Math.cos(lat1.toRad()) * Math.cos(lat2.toRad()) * \n Math.sin(dLon/2) * Math.sin(dLon/2); \nvar c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a)); \n//The distance between the vessel and the waypoint\n//divided by 1.852 to get it in nautical miles\nvar d = (R * c/1.852).toFixed(2); \n\n// This tfunction does not work as it stands now, the error from the \"ECDIS\"-value is larger than with just the distance instead\n//d = (d - global.get(\"Turnacc\")*0.000539957 - global.get(\"Waypointturnradius\")*Math.abs(Math.tan((global.get(\"Nextbearing\")-global.get(\"Bearing\"))*0.5))).toFixed(1);\n\nglobal.set(\"Distancetonextwaypoint\",d);\nmsg.payload = d;\n\n\nvar speed = global.get(\"Knot\");\nvar decimalTimeString = (d/speed);\nvar decimalTime = parseFloat(decimalTimeString);\ndecimalTime = decimalTime * 60 * 60;\n//var hours = Math.floor((decimalTime / (60 * 60)));\n//decimalTime = decimalTime - (hours * 60 * 60);\nvar minutes = Math.floor((decimalTime / 60));\ndecimalTime = decimalTime - (minutes * 60);\nvar seconds = Math.round(decimalTime);\n/*\nif(hours < 10)\n{\n\thours = \"0\" + hours;\n}*/\nif(minutes < 10)\n{\n\tminutes = \"0\" + minutes;\n}\nif(seconds < 10)\n{\n\tseconds = \"0\" + seconds;\n}\nvar Timetowaypoint = (\"\"+ minutes + \"m\" + seconds+\"\");\nif (Timetowaypoint == \"InfinitymNaN\")\nglobal.set(\"Timetonextwaypoint\",\"99m99\");\nelse\nglobal.set(\"Timetonextwaypoint\",Timetowaypoint);\n\nreturn msg;\n\n","outputs":1,"noerr":0,"x":410,"y":440,"wires":[["17769e36.82f3d2"]]},{"id":"48ca8ae5.4f7194","type":"ui_text","z":"fd97e6d8.2cb9a8","group":"13958700.8d0729","order":3,"width":0,"height":0,"name":"Movenow","label":"Movenow","format":"{{msg.payload}}","layout":"row-spread","x":620,"y":280,"wires":[]},{"id":"3daa542d.ca68ac","type":"function","z":"fd97e6d8.2cb9a8","name":"msg.payload = global.get(\"Movenow\");","func":"msg.payload = global.get(\"Movenow\") || \"Movenow not found!\";\nreturn

msg;\n","outputs":1,"noerr":0,"x":390,"y":280,"wires":[["48ca8ae5.4f7194"]]},{"id":"5675cd29.72f5e4","type":"inject","z":"fd97e6d8.2cb9a8","name":"10Hz","topic":"","payload":"","payloadType":"date","repeat":"0.1","crontab":"","once":true,"onceDelay":0.1,"x":150,"y":280,"wires":[["3daa542d.ca68ac"]]},{"id":"e10a3d0d.f852b","type":"function","z":"fd97e6d8.2cb9a8","name":"Edit variable","func":"global.set(msg.topic,msg.payload);\n","outputs":1,"noerr":0,"x":1030,"y":300,"wires":[[]]},{"id":"f42b7edf.e8987","type":"ui_slider","z":"fd97e6d8.2cb9a8","name":"Fontsize","label":"Fontsize","tooltip":"","group":"3c8b9d79.0630d2","order":2,"width":"0","height":"0","passthru":true,"outs":"all","topic":"Fontsize","min":"0","max":"50","step":"1","x":840,"y":180,"wires":[["e10a3d0d.f852b","acdc87cd.979778"]]},{"id":"acdc87cd.979778","type":"ui_text","z":"fd97e6d8.2cb9a8","group":"3c8b9d79.0630d2","order":1,"width":0,"height":0,"name":"Size","label":"Size","format":"{{msg.payload}}","layout":"row-spread","x":1010,"y":180,"wires":[]},{"id":"b9cfb19a.6f4e","type":"ui_colour_picker","z":"fd97e6d8.2cb9a8","name":"Font color","label":"Font color","group":"3c8b9d79.0630d2","format":"rgb","outformat":"object","showSwatch":true,"showPicker":false,"showValue":false,"showHue":true,"showAlpha":true,"showLightness":true,"dynOutput":"true","order":5,"width":0,"height":0,"passthru":true,"topic":"","x":840,"y":140,"wires":[["cf799446.e044b8","f391fe4a.355bc"]]},{"id":"cf799446.e044b8","type":"function","z":"fd97e6d8.2cb9a8","name":"Colorpicker","func":"global.set(\"Red\",msg.payload.r);\nglobal.set(\"Green\",msg.payload.g);\nglobal.set(\"Blue\",msg.payload.b);\nglobal.set(\"Alpha\",(msg.payload.a*255).toFixed(0));","outputs":1,"noerr":0,"x":1030,"y":100,"wires":[[]]},{"id":"f391fe4a.355bc","type":"ui_text","z":"fd97e6d8.2cb9a8","group":"3c8b9d79.0630d2","order":4,"width":0,"height":0,"name":"Color","label":"Color","format":"{{msg.payload}}","layout":"row-spread","x":1010,"y":140,"wires":[]},{"id":"4b131305.b01d9c","type":"ui_text","z":"fd97e6d8.2cb9a8","group":"50eda971.220498","order":1,"width":0,"height":0,"name":"Move by","label":"Move by","format":"{{msg.payload}}","layout":"row-spread","x":1020,"y":560,"wires":[]},{"id":"16b52b74.f4fdf5","type":"ui_slider","z":"fd97e6d8.2cb9a8","name":"Red","label":"Red","tooltip":"","group":"3c8b9d79.0630d2","order":6,"width":0,"height":0,"passthru":true,"outs":"all","topic":"Red","min":0,"max":"255","step":1,"x":850,"y":220,"wires":[["e10a3d0d.f852b"]]},{"id":"6acc8e31.b5c36","type":"ui_slider","z":"fd97e6d8.2cb9a8","name":"Green","label":"Green","tooltip":"","group":"3c8b9d79.0630d2","order":7,"width":0,"height":0,"passthru":true,"outs":"all","topic":"Green","min":0,"max":"255","step":1,"x":850,"y":260,"wires":[["e10a3d0d.f852b"]]},{"id":"18473bc7.8944b4","type":"ui_slider","z":"fd97e6d8.2cb9a8","name":"Blue","label":"Blue","tooltip":"","group":"3c8b9d79.0630d2","order":8,"width":0,"height":0,"passthru":true,"outs":"all","topic":"Blue","min":0,"max":"255","step":1,"x":850,"y":300,"wires":[["e10a3d0d.f852b"]]},{"id":"4025d794.332498","type":"ui_slider","z":"fd97e6d8.2cb9a8","name":"Alpha","label":"Alpha","tooltip":"","group":"3c8b9d79.0630d2","order":9,"width":0,"height":0,"passthru":true,"outs":"all","topic

":"Alpha","min":0,"max":"255","step":1,"x":850,"y":340,"wires":[["e10a3d0d.f852b"]]},{"id":"540b5da9.312f64","type":"ui_text_input","z":"fd97e6d8.2cb9a8","name":"Text to speech","label":"Text to speech","tooltip":"","group":"edf83ebe.a41ba","order":3,"width":0,"height":0,"passthru":true,"mode":"text","delay":"0","topic":"Texttospeech","x":820,"y":420,"wires":[["e10a3d0d.f852b"]]},{"id":"4316f64e.6dcc68","type":"ui_slider","z":"fd97e6d8.2cb9a8","name":"Delay","label":"Delay","tooltip":"","group":"edf83ebe.a41ba","order":2,"width":"0","height":"0","passthru":true,"outs":"all","topic":"Delay","min":"0","max":"5","step":"0.1","x":850,"y":460,"wires":[["760a5726.686808","e10a3d0d.f852b"]]},{"id":"760a5726.686808","type":"ui_text","z":"fd97e6d8.2cb9a8","group":"edf83ebe.a41ba","order":1,"width":0,"height":0,"name":"Delay","label":"Delay","format":"{{msg.payload}}","layout":"row-spread","x":1010,"y":460,"wires":[]},{"id":"db501928.4f9468","type":"function","z":"607c7664.d3f748","name":"Next next bearing","func":"start_latitude = parseFloat(global.get(\"Nextwaypointlat\"))*Math.PI/180;\nstart_longitude = parseFloat(global.get(\"Nextwaypointlon\"))*Math.PI/180;\nstop_latitude   = parseFloat(global.get(\"Nextnextwaypointlat\"))*Math.PI/180;\nstop_longitude = parseFloat(global.get(\"Nextnextwaypointlon\"))*Math.PI/180;\n\nvar y = Math.sin(stop_longitude-start_longitude) * Math.cos(stop_latitude);\nvar x = Math.cos(start_latitude)*Math.sin(stop_latitude) -\n   Math.sin(start_latitude)*Math.cos(stop_latitude)*Math.cos(stop_longitude-start_longitude);\nvar Bearing = Math.atan2(y, x) * 180 / Math.PI;\nBearing = (Bearing + 360) % 360\nBearing = Bearing.toFixed(3);\nglobal.set(\"Nextnextbearing\",Bearing);\nmsg.payload = Bearing;\nreturn msg;","outputs":1,"noerr":0,"x":390,"y":600,"wires":[[]]},{"id":"d3483b60.90f508","type":"function","z":"607c7664.d3f748","name":"wp==selected+3","func":"var selected = global.get('count') || 0;\nvar wp = Number(msg.payload.WayPoint.Id);\nif (wp==(selected+3))\nreturn msg;","outputs":1,"noerr":0,"x":1000,"y":320,"wires":[["4873553c.ded98c"]]},{"id":"4873553c.ded98c","type":"function","z":"607c7664.d3f748","name":"Waypoint ->Global variables","func":"var msg1 = { payload: msg.payload.WayPoint.Id };\nvar msg2 = { payload: msg.payload.WayPoint.WPName };\nvar msg3 = { payload: msg.payload.WayPoint.Lat };\nvar msg4 = { payload: msg.payload.WayPoint.Lon };\nvar msg5 = { payload: msg.payload.WayPoint.TurnRadius };\nvar msg6 = { payload: msg.payload.WayPoint.LegType };\nvar msg7 = { payload: msg.payload.WayPoint.XTE };\nvar msg8 = { payload: msg.payload.WayPoint.Speed };\nvar msg9 = { payload: msg.payload.WayPoint.Danger};\nvar msg10 = { payload: msg.payload.WayPoint.Message };\nvar msg11 = { payload: msg.payload.WayPoint.RTime };\n\nglobal.set('Nextnextnextwaypointid',msg.payload.WayPoint.Id);\nglobal.set('Nextnextnextwaypointname',msg.payload.WayPoint.WPName);\nglobal.set('Nextnextnextwaypointlat',msg.payload.WayPoint.Lat);\nglobal.set('Nextnextnextwaypointlon',msg.payload.WayPoint.Lon);\nglobal.set('Nextne

xtnextwaypointturnradius',msg.payload.WayPoint.TurnRadius);\nglobal.set('Nextnextnextwaypointlegtype',msg.payload.WayPoint.LegType);\nglobal.set('Nextnextnextwaypointxte',msg.payload.WayPoint.XTE);\nglobal.set('Nextnextnextwaypointspeed',msg.payload.WayPoint.Speed);\nglobal.set('Nextnextnextwaypointdanger',msg.payload.WayPoint.Danger);\nglobal.set('Nextnextnextwaypointmessage',msg.payload.WayPoint.Message);\nglobal.set('Nextnextnextwaypointrtime',msg.payload.WayPoint.RTime);\n\nreturn
[msg1,msg2,msg3,msg4,msg5,msg6,msg7,msg8,msg9,msg10,msg11]","outputs":1,"noerr":0,"x":1220,"y":320,"wires":[[]]},{"id":"cdee3d4c.7b22","type":"function","z":"607c7664.d3f748","name":"DISABLED If Distance --> nextwaypoint","func":"// This fucntion is somewhat bugged\n// In addition, it is desired to change waypoint \n// after you have passed it, not before\n\nif (global.get(\"Distancetowaypointtwodecimal\")<global.get(\"Waypointarrival\")){\n global.set(\"Automaticwaypoint\",\"True\")\n msg.increment = 1;\n return msg;\n}\n\n\n","outputs":1,"noerr":0,"x":420,"y":280,"wires":[["a04cda7b.3f5238","25c25e9e.f5f7e2"]]},{"id":"a416b296.9f3d7","type":"ui_slider","z":"607c7664.d3f748","name":"Auto-advance distance","label":"Auto-advance distance","tooltip":"","group":"e7285da5.dfd63","order":15,"width":0,"height":0,"passthru":true,"outs":"all","topic":"Waypointarrival","min":"0","max":"3","step":"0.01","x":360,"y":740,"wires":[["a4b4c714.231a58","a96d1d2e.515bb"]]},{"id":"b653a312.cce2d","type":"comment","z":"607c7664.d3f748","name":"Auto-advance distance","info":"","x":400,"y":700,"wires":[]},{"id":"a4b4c714.231a58","type":"ui_text","z":"607c7664.d3f748","group":"e7285da5.dfd63","order":14,"width":0,"height":0,"name":"Auto-advance distance","label":"Auto-advance distance","format":"{{msg.payload}}","layout":"row-spread","x":620,"y":740,"wires":[]},{"id":"7fe427cc.606258","type":"ui_slider","z":"607c7664.d3f748","name":"Turnaccelerationlength","label":"Turnaccelerationlength","tooltip":"","group":"e7285da5.dfd63","order":15,"width":0,"height":0,"passthru":true,"outs":"all","topic":"Turnacc","min":"0","max":"0.5","step":"0.001","x":360,"y":840,"wires":[["ccc196f6.fb9228","a96d1d2e.515bb"]]},{"id":"16b8c791.6c9fe8","type":"comment","z":"607c7664.d3f748","name":"Auto-advance distance","info":"","x":400,"y":800,"wires":[]},{"id":"ccc196f6.fb9228","type":"ui_text","z":"607c7664.d3f748","group":"e7285da5.dfd63","order":14,"width":0,"height":0,"name":"Turnaccelerationlength","label":"Turnaccelerationlength","format":"{{msg.payload}}","layout":"row-spread","x":620,"y":840,"wires":[]},{"id":"a96d1d2e.515bb","type":"function","z":"607c7664.d3f748","name":"Edit variable","func":"global.set(msg.topic,msg.payload);\n","outputs":1,"noerr":0,"x":650,"y":780,"wires":[[]]},{"id":"1b359195.0235be","type":"function","z":"607c7664.d3f748","name":"wp==selected-1","func":"var selected = global.get('count') || 0;\nvar wp = Number(msg.payload.WayPoint.Id);\nif (wp==(selected-1))\nreturn msg;

","outputs":1,"noerr":0,"x":1000,"y":360,"wires":[["640ceee3.81cba"]]},{"id":"640ceee3.81cba","type":"function","z":"607c7664.d3f748","name":"Waypoint ->Global variables","func":"\nvar msg1 = { payload: msg.payload.WayPoint.Id };\nvar msg2 = { payload: msg.payload.WayPoint.WPName };\nvar msg3 = { payload: msg.payload.WayPoint.Lat };\nvar msg4 = { payload: msg.payload.WayPoint.Lon };\nvar msg5 = { payload: msg.payload.WayPoint.TurnRadius };\nvar msg6 = { payload: msg.payload.WayPoint.LegType };\nvar msg7 = { payload: msg.payload.WayPoint.XTE };\nvar msg8 = { payload: msg.payload.WayPoint.Speed };\nvar msg9 = { payload: msg.payload.WayPoint.Danger};\nvar msg10 = { payload: msg.payload.WayPoint.Message };\nvar msg11 = { payload: msg.payload.WayPoint.RTime };\n\n//Relevant information -> global variables with function global.set\nglobal.set('Prevwaypointname',msg.payload.WayPoint.WPName);\n\nglobal.set('Prevwaypointlat',msg.payload.WayPoint.Lat);\nglobal.set('Prevwaypointlon',msg.payload.WayPoint.Lon);\n\n//Different outputs\nreturn [msg1,msg2,msg3,msg4,msg5,msg6,msg7,msg8,msg9,msg10,msg11]","outputs":1,"noerr":0,"x":1220,"y":360,"wires":[[]]},{"id":"cfdd3adf.c64408","type":"ui_gauge","z":"607c7664.d3f748","name":"Bearing","group":"80858abc.d026f8","order":3,"width":0,"height":0,"gtype":"compass","title":"Bearing","label":"Degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":720,"y":520,"wires":[]},{"id":"4d81439c.261c3c","type":"ui_gauge","z":"607c7664.d3f748","name":"NextBearing","group":"80858abc.d026f8","order":3,"width":0,"height":0,"gtype":"compass","title":"NextBearing","label":"Degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":730,"y":560,"wires":[]},{"id":"ea0ef1ee.3ee43","type":"ui_text","z":"607c7664.d3f748","group":"80858abc.d026f8","order":2,"width":"0","height":"0","name":"Distance","label":"Distance","format":"{{msg.payload}}","layout":"col-center","x":720,"y":400,"wires":[]},{"id":"232b5494.f07e8c","type":"ui_gauge","z":"19d59705.d99769","name":"Heading Sensor","group":"4301e921.cca2c8","order":0,"width":0,"height":0,"gtype":"compass","title":"Heading","label":"degrees","format":"{{value}}","min":0,"max":"360","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":780,"y":220,"wires":[]},{"id":"cd916d89.72229","type":"ui_text","z":"607c7664.d3f748","group":"80858abc.d026f8","order":2,"width":"0","height":"0","name":"Time to waypoint","label":"Time to waypoint","format":"{{msg.payload}}","layout":"col-center","x":750,"y":440,"wires":[]},{"id":"47976610.092858","type":"ui_text","z":"607c7664.d3f748","group":"80858abc.d026f8","order":2,"width":"0","height":"0","name":"Offset","label":"Offset","format":"{{msg.payload}}","layout":"col-center","x":710,"y":480,"wires":[]},{"id":"db70a55.d5be358","type":"ui_text","z":"607c7664.d3f748","group":"80858abc.d026f8","order":2,"width":"0","height":"0","name":"XTE","label":"XTE","format":"{{msg.payload}}","layout":"col-

center","x":710,"y":640,"wires":[]},{"id":"cb35019.4a5b5","type":"ui_button","z":"607c7664.d3f748","name":"Next waypoint","group":"e8a812ff.2917","order":2,"width":0,"height":0,"passthru":false,"label":"Next waypoint","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"","x":120,"y":200,"wires":[["b756f47e.6608a8","25c25e9e.f5f7e2"]]},{"id":"d523a31e.23777","type":"ui_button","z":"607c7664.d3f748","name":"Previous Waypoint","group":"e8a812ff.2917","order":1,"width":0,"height":0,"passthru":false,"label":"Previous Waypoint","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"","x":110,"y":240,"wires":[["5c06dfc5.ea294","25c25e9e.f5f7e2"]]},{"id":"9c9ad84b.ca5688","type":"ui_button","z":"607c7664.d3f748","name":"Reset Seilas","group":"e8a812ff.2917","order":0,"width":0,"height":0,"passthru":false,"label":"Reset Seilas","tooltip":"","color":"","bgcolor":"","icon":"","payload":"","payloadType":"str","topic":"","x":130,"y":280,"wires":[["bba04bce.c0d188","25c25e9e.f5f7e2"]]},{"id":"ccda90.7032a57","type":"ui_text","z":"607c7664.d3f748","group":"e8a812ff.2917","order":10,"width":0,"height":0,"name":"","label":"Selected Waypoint","format":"{{msg.count}}","layout":"row-spread","x":750,"y":280,"wires":[]},{"id":"17769e36.82f3d2","type":"ui_text","z":"607c7664.d3f748","group":"80858abc.d026f8","order":2,"width":"0","height":"0","name":"NextDistance","label":"NEXTDistance","format":"{{msg.payload}}","layout":"col-center","x":730,"y":360,"wires":[]},{"id":"24b75295.974d8e","type":"comment","z":"607c7664.d3f748","name":"Regner ut en rekke variabler ","info":"","x":420,"y":400,"wires":[]},{"id":"e94b450c.1ac828","type":"comment","z":"607c7664.d3f748","name":"10Hz","info":"","x":170,"y":420,"wires":[]},{"id":"73054c77.4e0c54","type":"comment","z":"fd97e6d8.2cb9a8","name":"Send text for TTS with a set delay","info":"","x":770,"y":380,"wires":[]},{"id":"5c7be0fa.8659","type":"comment","z":"fd97e6d8.2cb9a8","name":"Define color of HoloLens UI","info":"","x":780,"y":100,"wires":[]}]