

Sammendrag

VR har i økende grad blitt tatt i bruk i norsk helsevesen. I dag er dette hovedsakelig i opplæring av helsepersonell og behandling av pasienter, da særlig innen psykisk helse, men det jobbes hyppig med å finne flere bruksområder for ny teknologi. Ved å ta i bruk VR kan man simulere nye miljøer som vil gi pasienter og spesialister muligheten til å utforske nye løsninger.

Oppgaven tar utgangspunkt i hjerneslag, fenomenet neglekt som kan oppstå etter hjerneslag og hvordan man i dag evaluerer en pasient med neglekt. Tidligere har det blitt gjort forsøk på å digitalisere tradisjonelle tester og utføre disse testene i VR, men dette krever at brukeren har kompetanse innen utvikling og teknologi.

I denne bacheloroppgaven jobbet gruppa derfor med å gi spesialister muligheten til å kunne lage og designe sine egne løsninger. Gruppa har utviklet et VR-miljø hvor spesialister kan eksperimentere og lage egne tester og lagre disse testene for å videre kunne bruke dem til å evaluere pasienter. I dette miljøet er det mulig å plassere ulike typer objekter, justere størrelse og bevegelse til objektene, spore øyebevegelse og analysere relevante data i ettertid. Ettersom dette produktet er utviklet for medisinsk personell, er det satt stor vekt på enkel og selvforklarende bruk.

Abstract

VR has increasingly been adopted in the Norwegian healthcare system. Currently, it is mainly used in training of personnel and treatment of patients, especially regarding mental health, but there is frequently researched more areas of use for new technology. By adopting VR, new environments can be simulated which will give patients and specialists the opportunity to explore new solutions.

The assignment is based on stroke, a medical condition, the phenomenon of neglect that can occur after a stroke and how to evaluate a patient with neglect today. In the past, attempts have been made to digitize traditional tests and perform these tests in VR, but this requires the user to have expertise in development and technology.

In this bachelor's thesis, the group has therefore strived to give specialists the opportunity to be able to create and design their own solutions. The group has developed a VR environment where specialists can experiment and create their own tests and save these tests so that they can be used for patient evaluation. In this environment, it is possible to place different types of objects, adjust the size and movement of the objects, track eye movements and analyze relevant data afterwards. As this product has been developed for medical personnel, great emphasis has been placed on simple and easy use.

Forord

Foran deg ligger bacheloroppgaven for studieretningen Dataingeniør ved Institutt for data teknologi og informatikk, NTNU. Denne oppgaven er skrevet som et krav for å fullføre faget IDATT2900 Bacheloroppgave og er vår siste oppgave i løpet av vårt 3-årige studieløp.

Med en økende interesse i VR og et ønske om å lage et program som kan bli brukt til å hjelpe mennesker, var dette en oppgave som fanget oppmerksomheten til gruppa ved første øyekast. Implementasjon og utvidelse av teknologi som VR i hverdagen og områder hvor teknologien gjerne ikke virker naturlig er noe alle i gruppa synes er meget spennende. Dette var derfor en perfekt oppgave for gruppa som alle engasjerte seg i.

Vi vil gjerne takke veilederen vår, Alexander Holt, for støtte, engasjement og god veiledning gjennom hele prosessen. Vi vil også takke Tor Ivar Hansen, nevropsykolog fra Institutt for nevromedisin og bevegelsesvitenskap, for innsikt, interessante idéer og gode tilbakemeldinger.

Trondheim 19. mai 2023



Dominykas Mazys



Minh Dan Nguyen



Joel Mattias Tømmerbakk

Oppgavetekst

Oppgaven går ut på implementere en prototype av et VR-miljø, hvor spesialister innen helsesektoren kan designe eksperimenter for å utforske en gitt pasients neglekt. Brukere av systemet vil primært være medisinsk personell, så lett forståelig og enkel bruk er noe som vektlegges. Systemet må likevel være fleksibelt nok når en skal designe eksperimenter, slik at man har reell nytteverdi.

Selv evalueringen av neglekt vil skje gjennom å analysere blikkbruk under og/eller etter eksperimentene. Denne blikk-bruken vil fanges ved hjelp av øyesporing i VR-headsetet, og det er derfor nødvendig med lagring av både øye-data og miljø-data. Disse dataene må være synkrone, slik at man vet hvilke eventer under eksperimentet forårsaket hvilke øyebevegelser. Det er også ønskelig å kunne visualisere blikkbruken.

Se visjonsdokument (se vedlegg 3) og kravdokumentasjon (se vedlegg 4) for liste over spesifikke funksjonelle krav.

Innholdsfortegnelse

Figurer	xi
Tabeller	xi
1. Introduksjon og relevans	1
1.1. Bakgrunn	1
1.2. Akronymer og forkortelser	2
2. Teori og relevant litteratur	3
2.1. Hjerneslag	3
2.1.1. Neglekt	3
2.1.2. Evaluering av neglekt	3
2.1.3. Evaluering av neglekt med VR	5
2.2. Teknologi	5
2.2.1. Virtuell Virkelighet	5
2.2.2. Øyesporing	5
2.2.3. Head Mounted Display	5
2.2.4. Spillmotor	6
2.3. Utvikling	6
2.3.1. Smidig utvikling	6
2.3.2. Versjonskontroll	7
3. Metode	8
3.1. Utviklingsprosess	8
3.1.1. Smidig utvikling med Kanban	8
3.1.2. GitLab	8
3.2. Valg av teknologi	8
3.2.1. HTC Vive Pro Eye	8
3.2.2. Unity	9
3.2.3. OpenXR	9
3.3. Visualisering	10
3.3.1. Spredningsplot	10
3.3.2. Heatmap	11
3.3.3. Plott av objekter	11
4. Resultater	14
4.1. Ingeniørfaglige resultater	14
4.1.1. Funksjonelle krav fra visjonsdokument	14
4.1.2. Funksjonelle krav fra bruker etter demo	15
4.1.3. Ikke-funksjonelle krav	15

4.1.4. Svar fra brukertester	17
4.2. Administrative resultater	20
4.2.1. Kanban-tavle	20
4.2.2. Timeregnskap	21
5. Diskusjon	22
5.1. Diskusjon av ingeniørfaglige resultater	22
5.1.1. Funksjonelle krav	22
5.1.1.1. Hovedmeny	22
5.1.1.2. Oppretting og design av eksperimenter	22
5.1.1.3. Gjennomføring av eksperimenter	26
5.1.1.4. Datalagring	28
5.1.2. Ikke-funksjonelle krav	30
5.1.3. Svar fra brukertester	30
5.2. Diskusjon av administrative resultater	31
5.2.1. Kanban-tavle	31
5.2.2. Timeregnskap	31
5.2.3. Gruppearbeid	32
5.3. Diskusjon av metode	32
5.3.1. Utviklingsprosess	32
5.3.1.1. Smidig utvikling med Kanban	32
5.3.1.2. GitLab	32
5.3.2. Visualisering	32
6. Konklusjon og videre arbeid	34
6.1. Konklusjon	34
6.2. Videre arbeid	34
6.2.1. Replay-funksjon	34
6.2.2. 3D visualisering	34
6.2.3. Bevegelsesbane	34
6.2.4. Objekter av forskjellige form	35
Samfunnspåvirkning	36
Referanser	37
Vedlegg	39

Figurer

Figur 2.1: Et eksempel på et resultat av en test for neglekt [7]	4
Figur 2.2: Et eksempel på epletetesten [8].	4
Figur 2.3: Eksempel på et Head Mounted Display [14]	6
Figur 3.1: Eksempel på et spredningsplott	10
Figur 3.2: Eksempel på et varmekart	11
Figur 3.3: Fullt-objekt, venstre-side-objekt, høyre-side-objekt	12
Figur 3.4: Eksempel på visualisering av objekter	12
Figur 4.1: Eksempel på et tooltip	16
Figur 4.2: WCAG-test for blå knapper	16
Figur 4.3: Bilde av knapp for å illustrere kontrast	16
Figur 4.4: Resultater fra det første spørsmålet i brukertesten	18
Figur 4.5: Resultater fra det andre spørsmålet i brukertesten	18
Figur 4.6: Resultater fra det tredje spørsmålet i brukertesten	19
Figur 4.7: Resultater fra det fjerde spørsmålet i brukertesten	19
Figur 4.8: Resultater fra det femte spørsmålet i brukertesten	20
Figur 4.9: Gruppas Kanban-tavle midt i prosjektet	20
Figur 4.10: Diagram som viser andel tid brukt per aktivitet	21
Figur 5.1: Eksperiment-området	22
Figur 5.2: Vegg som viser hvordan kontrollene brukes	23
Figur 5.3: Meny på venstre kontroll	23
Figur 5.4: Meny av objektvariabler	24
Figur 5.5: Meny for endring av objektets bevegelse	25
Figur 5.6: Meny over lagring og lasting av eksperiment	25
Figur 5.7: Meny ved gjennomføring av spesiallaget eksperiment	26
Figur 5.8: Meny ved gjennomføring av randomisert eksperiment	27
Figur 5.9: Knapp for forhåndsvisning av fullstendig test/eksperiment	30
Figur 5.10: Eksperiment-området med rutenett	31

Tabeller

Tabell 1.1: Tabell over forkortelser og betydning	2
Tabell 4.1: Tabell over resultat av funksjonelle krav fra visjonsdokumentet	14
Tabell 4.2: Tabell over funksjonelle krav fra bruker etter demo	15

1. Introduksjon og relevans

1.1. Bakgrunn

Mange tusen nordmenn blir årlig rammet av hjerneslag og 67% av alle nordmenn som har gjennomgått hjerneslag har funksjonssvikt som følge av slaget [1]. Et av de mest utbredte utfallene av hjerneslag er et fenomen kalt neglekt. En presis og sensitiv evaluering av pasienten er kritisk for å kunne la dem gjennomgå en effektiv rehabilitering. I tillegg er det viktig å få en nøyaktig diagnose av påvirkningsgrad slik at man kan trygt slippe dem tilbake ut i samfunnet uten at de vil være en fare for seg selv eller andre. Oppgavens mål er å utvikle et program som kan være et tilleggsverktøy for spesialister for å hjelpe dem med evaluering av slagpasienter.

I et samarbeid med Klinikk for Fysikalsk Medisin og Rehabilitering Lian St. Olav har NTNU Vizlab opprettet et prosjekt som går ut på evaluering og rehabilitering av slagpasienter. Under dette prosjektet har det blitt lagt flere oppgaver som tar for seg dette temaet, blant annet gruppas bacheloroppgave som innebærer å lage en prototype av et VR-miljø hvor det skal være mulig for en spesialist å designe egne eksperimenter i VR. I tillegg skal en pasient kunne gjennomføre den selvlagde testen mens en spesialist følger med på en egen skjerm, mens relevante data angående øyesporing og blikkfang blir samlet underveis for analyse i ettertid. Videre tar oppgaven utgangspunkt i å utforske en pasients neglekt, et mulig symptom av hjerneslag. En tidligere utført oppgave har hatt som mål å lage en prototype som lar pasienter gjennomgå tester som evaluerer neglekt i VR. Gruppa har brukt denne oppgaven som inspirasjon og har hatt som mål å lage et robust og fullverdig VR-miljø som fyller den forrige prototypens mangler.

Basert på oppgavens bakgrunn har problemstillingen blitt som følger:

"Utvikle et VR-miljø med øyesporing for eksperimentering, med utgangspunkt i å gi spesialister innen helsesektoren et mer fleksibelt miljø for evaluering av pasienter."

Denne rapporten tar for seg oppgavens problemstilling og beskriver prosessen og resultatet til oppgaven. Rapporten er bygget opp av kapitler som dekker blant annet teori og relevant litteratur, prosjektets metodologi, resultat og sluttprodukt, diskusjon og til slutt konklusjon.

1.2. Akronymer og forkortelser

Forkortelse	Forklaring
VR	Virtual Reality, virtuell virkelighet
XR	Extended Reality
HMD	Head Mounted Display
OLED	Organic Light-Emitting Diode
JSON	JavaScript Object Notation
CSV	Comma-Separated Values, kommaseparert fil
UI	User Interface, brukergrensesnitt
WCAG	Web Content Accessibility Guidelines
NTNU	Norges teknisk-naturvitenskapelige universitet
Plastic SCM	Plastic Software Configuration Management system

Tabell 1.1: Tabell over forkortelser og betydning

2. Teori og relevant litteratur

I dette kapitlet blir det gjort rede for teori og litteratur som er relevant for prosjektet.

2.1. Hjerneslag

Hjerneslag er en fellesbetegnelse på sykdomstilstander forårsaket av plutselige forstyrrelser av blodsirkulasjon i hjernen [2]. I Norge blir rundt 12 000 personer rammet årlig av hjerneslag, og selv om man kan redusere risikoen, kan ingen forsikre seg mot å få hjerneslag [3]. Symptomene på hjerneslag oppstår plutselig og varierende, hvor de vanligste symptomene er lammelse og språkforstyrrelser eller talevansker.

Ved hjerneblødninger blir oksygentilførselen til hjernen forhindret som fører til at hjerneceller dør [3]. Blødningen øker gradvis i størrelse og kan fort føre til irreversibel skade, eller i verste fall død. Tiden før pasienten får behandling kan derfor være avgjørende for prognosene og det er anbefalt at man ringer 113 umiddelbart dersom man mistenker at noen har fått hjerneslag [2].

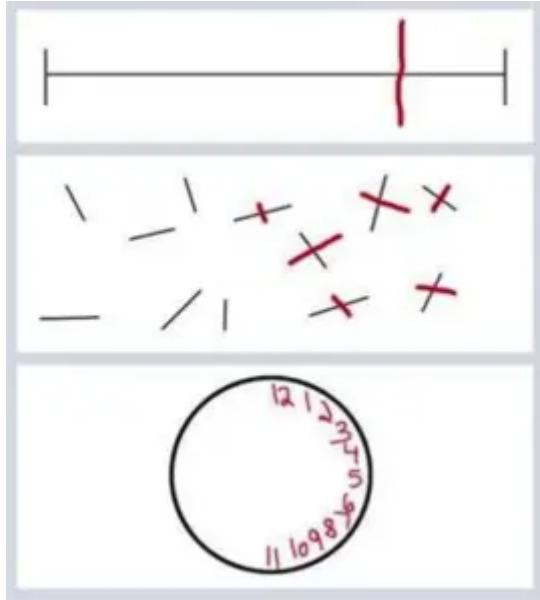
2.1.1. Neglekt

Neglekt er et mulig kognitivt utfall etter hjerneslag som gir redusert oppmerksomhet mot egen kropp eller synsfelt, til tross for upåvirket synsfelt. Neglekt oppstår på motsatt side av den påvirkede delen av hjernen etter skaden. Det vil si, dersom venstresiden av hjernen ble påvirket etter skaden, vil man oppleve neglekt på høyresiden. Neglekt oppstår vanligvis på venstresiden, og det finnes varierende grader av neglekt [4].

Individer med neglekt reagerer ikke på stimuli på den påvirkede siden og oppfører seg som om den siden ikke eksisterer. Man skiller vanligvis mellom kroppsneglekt og visuell neglekt, men begge kan opptre samtidig. Individer med kroppsneglekt vil gjerne overse den ene delen av kroppen sin. Eksempler på dette kan være å kun kle på seg, barbere eller stelle seg på den ene halvdelen av kroppen. Når det gjelder visuell neglekt vil man ofte kun spise fra den ene siden av tallerken eller dunke borti gjenstander og dørkarmer på venstresiden av kroppen [5].

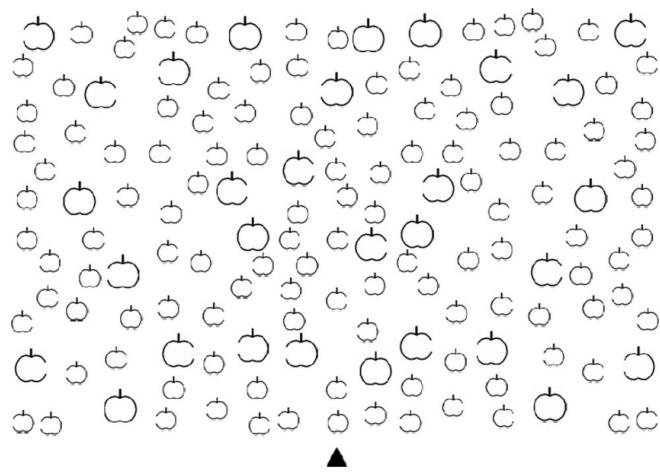
2.1.2. Evaluering av neglekt

Neglekt blir tradisjonelt diagnostisert med tester som tar i bruk penn og papir. Eksempler på noen av disse testene kan være å forsøke å finne midtpunktet på en linje, kopiere og tegne figurer eller krysse ut målrettede objekter i et bilde [6]. Her vil en person med neglekt gjerne ha problemer med å finne midtpunktet på linja eller vil de utelukke den ene halvdelen av tegninger de skal kopiere.



Figur 2.1: Et eksempel på et resultat av en test for neglekt [7]

Slik kan resultatet av en test gjort av en pasient med neglekt på venstresiden se ut. De første to testene har som mål å finne midtpunktet på linjene, mens den siste testen har som mål å tegne en klokke. I den første testen vil pasienten overse venstresiden av testen og linjen kan dermed virke kortere. I likhet med den første testen, har pasienten oversett venstresiden av den andre testen også. Pasienten har derfor bare funnet midtpunktet til linjene på høyresiden. I den siste testen har pasienten gjerne vansker med å se venstresiden av klokka.



Figur 2.2: Et eksempel på eplettesten [8].

Oppgaven tar utgangspunkt i en test som ofte blir kalt eplettesten, eller "Apple Cancellation Test" på engelsk. Testens mål er å krysse ut alle eplene som er fulle. En pasient med neglekt vil muligens i likhet med de forrige testene ignorere den ene halvsiden av testen. I tillegg kan individer med neglekt også ignorere halvdelen av et objekt, uavhengig av hvor objektet ligger. I eplettesten er det epler som mangler en bit av høyre- eller venstresiden. En pasient med neglekt på venstresiden vil da kunne ignorere at eplene har en mangel på venstresiden og det vil se ut som eplene er hele.

2.1.3. Evaluering av neglekt med VR

Det har blitt gjort flere forsøk og studier angående evaluering av neglekt ved bruk av VR. Eksempelvis har noen av de tradisjonelle testene blitt utført i et VR-miljø. I disse testene har kontrollørene ikke hatt problemer med å utføre gitte oppgaver, mens individer med neglekt hadde problemer med å identifisere objekter lokalisert i det påvirkede synsfeltet. Andre VR-tester har også vist at individer med neglekt viser symptomer på neglekt i testene og scorer vanligvis likt eller verre i de samme testene i VR. Dette kan vise til at de samme testene man utfører på papir kan prestere like bra i et VR-miljø [9].

I tillegg til at det er mulig å registrere typisk adferd på neglekt i et VR-miljø, har man også muligheten til å utforske nye måter å evaluere pasienter på og samle inn ytterlige verdifulle data. Eksempelvis har VR blitt brukt til å utforske og studere pasienter med neglekt utføre aktiviteter i et dagligdags miljø. Videre får man også muligheten til å samle inn viktig data som kan brukes til evaluering av pasienten, som for eksempel øyesporingsdata. Dette kan gi observatører et bedre innsyn på hva pasienten fikserer eller fokuserer på i bildet [9]. Til slutt vil tester i VR kunne utforske dybdesyn. Disse testene vil kunne gi en ny akse med data som ikke er mulig å utnytte i tradisjonelle tester.

2.2. Teknologi

2.2.1. Virtuell Virkelighet

Med virtuell virkelighet, vanligvis forkortet til VR, bruker man teknologi og datagrafikk til å simulere et virtuelt tredimensjonalt miljø for brukeren. Brukeren observerer dette miljøet ved bruk av et Head Mounted Display som dekker hele synsfeltet slik at brukeren skal bli mindre bevisst på den faktiske virkeligheten de befinner seg i [10].

Brukeren kan også bruke bevegelseskontroller som virker som virtuelle hender, og gir brukeren mulighet for interaksjon med virtuelle objekter og manipulering av det virtuelle miljøet de befinner seg i [11].

2.2.2. Øyesporing

Øyesporing er en teknologi som kan spore og følge det en person ser på i sanntid. Slik teknologi kan både brukes i separate enheter for øyesporing og som en del av annet utstyr, som for eksempel et VR-headset. For å spore øyebevegelser i et VR-headset brukes det et kamera og infrarødt lys. Det infrarøde lyset pekes mot midtpunktet til pupillen, og avstanden mellom refleksjonen av pupillen og hornhinnen måles av kameraet [12].

Teknologien kan oversette en persons øyebevegelser til data som bland annet posisjonen til pupillene, blikkvektorer for hvert øye, og blikkpunkt [12]. Denne dataen kan videre brukes for å analysere og visualisere øyebevegelsene.

2.2.3. Head Mounted Display

Et Head Mounted Display består av to små stereoskopiske skjermer, ett for hvert øye, som hver viser et virtuelt miljø. Den ene skjermen viser det virtuelle miljøet fra en

annen vinkel enn den andre. Dette fører til at det skapes en illusjon av dybdesyn, siden forskjellen i vinkel gir en indikasjon på avstand.

Bevegelsessensorer plukker opp bevegelsen av brukerens hodet og justerer bildet på skjermene slik at brukeren får opplevelsen av at de beveger seg rundt i et tredimensjonalt miljø [13].



Figur 2.3: Eksempel på et Head Mounted Display [14]

2.2.4. Spillmotor

En spillmotor er et slags rammeverk for kjøring og bygging av spill og andre interaktive programmer som opererer i sanntid. Denne programvaren består av mange forskjellige komponenter, som hver har ansvar for ulik funksjonalitet som er vanlig i spill.

Spillmotorer pleier å ha komponenter som en grafikkmotor som støtter todimensjonal eller tredimensjonal grafikk, en fysikkmotor som kan simulere kollisjoner og tyngdekraft, animasjon, brukergrensesnitt, lyd og mer. Dersom man ønsker stor kontroll over hvordan disse komponentene skal fungere er det mulig å utvikle sin egen spillmotor, men dette kan være en veldig tids- og ressurskrevende jobb. Av denne grunn er det ganske vanlig å benytte seg av spillmotorer som andre selskaper har lagd [15]. Dette kan være spesielt nyttig for mindre prosjekter siden man ikke trenger å sette av store mengder tid til å utvikle en spillmotor, og man kan heller begynne rett på selve utviklingen av produktet.

2.3. Utvikling

2.3.1. Smidig utvikling

Smidig programvareutvikling er en utviklingsmetodologi som går ut på å dele prosessen inn i flere deler. Metodologien legger vekt på kontinuerlig samarbeid og utvikling i motsetning til tradisjonelle metoder som understrekker viktigheten med en ferdig plan ved starten av prosjektet [16]. Manifestet for smidig programvareutvikling var en reaksjon på tradisjonelle utviklingsprosesser som var strengt bundet til en nøye planlagt

timeplan med lite rom for endringer. Manifestets forfattere mente at datidens bedrifter prioriterte planlegging og dokumentering over kundenes tilfredshet. Dette manifestet poengterer sine verdier som er som følgende:

Personer og samspill fremfor prosesser og verktøy

Programvare som virker fremfor omfattende dokumentasjon

Samarbeid med kunden fremfor kontraktsforhandlinger

Å reagere på endringer fremfor å følge en plan [17]

Med disse verdiene dukket det opp en ny utviklingsprosess som jobber iterativt med å levere resultater til kundene deres. I tillegg er det en kontinuerlig planlegging gjennom hele prosessen slik at utviklerne kan reagere effektivt til endringer og tilbakemeldinger [16].

2.3.2. Versjonskontroll

I en gruppe med programvareutviklere er et versjonskontrollsysten et viktig verktøy brukt for å kunne programmere samtidig og spore endringer. Når flere personer redigerer på en kildekode samtidig vil det kunne oppstå konflikter, feil eller lignende som gjør endringene inkompatibel med resten av kildekoden. Disse problemene kan bli løst ved å bruke versjonskontroll. Versjonskontroll omhandler sporing og administrering av programvarekode og kan brukes til å gå tilbake i tid og sammenligne ulike versjoner av samme kode [18].

3. Metode

I dette kapitlet blir metodene brukt for å løse oppgaven beskrevet og hvorfor akkurat disse metodene ble valgt blir begrunnet.

3.1. Utviklingsprosess

3.1.1. Smidig utvikling med Kanban

Oppgaven hadde mange muligheter for egen tolkning og det var satt få krav til hvordan sluttproduktet skulle se ut. Etter det første møte med oppdragsgiver fikk gruppa en oppfatning av at det var gruppa selv som kom til å ta de fleste beslutningene angående produktet. Det ble derfor viktig for gruppa å bruke en utviklingsprosess som var fleksibel og åpen for forandringer i løpet av utviklingen. Metoder som fossefallsmetoden eller scrum krever at gruppen og kunden har en god initiell forståelse for hvordan sluttproduktet skal se ut. Scrum deler sluttmålet i mange små mål og har ofte en streng timeplan med flere seremonier og fossefallsmetoden krever omfattende dokumentasjon ved starten av prosjektet. Med tanke på hvor åpen oppgaven var, var det essensielt å ta i bruk en utviklingsprosess som lot gruppa ha friheten til å endre på krav og funksjoner underveis uten å måtte revidere og forkaste gamle planer. Det ble derfor bestemt at gruppa skulle ta i bruk en smidig utviklingsmetode med Kanban-tavle.

En Kanban-tavle blir ofte brukt som et hjelpemiddel for å visualisere arbeidsflyt ved smidige utviklingsmetoder. En Kanban-tavle er delt inn i kolonner som forklarer status eller aktivitet og kort som beskriver arbeidsoppgavene. Gruppa valgte å ta i bruk dette hjelpebiddelet for å organisere viktige funksjoner og effektivisere arbeidsflyten innad i gruppa.

3.1.2. GitLab

GitLab er et nettbasert Git-programvarelager. Git er et system for versjonskontroll som brukes til å spore endringer i datafiler som beskrevet i kapittel 2.3.2 [19]. Gruppa valgte å bruke Gitlab som versjonskontroll da dette var et system alle på gruppa hadde erfaring med. I tillegg hadde GitLab funksjoner som en digital Kanban-tavle som gjorde valget mer attraktivt for gruppa.

3.2. Valg av teknologi

3.2.1. HTC Vive Pro Eye

Siden øyesporingen er en essensiell del av denne oppgaven brukte gruppa HTC Vive Pro Eye. Dette VR-headsettet har to OLED-skjermer med en total oppløsning på 2880 x 1600 piksler, 110 grader synsfelt, og øyesporing drevet av Tobii. Dataen fra øyesporingen har en utgangsfrekvens på 120Hz med en nøyaktighet på 0.5-1.1 grader (innen et synsfelt på 20 grader). Headsettet har også en G-sensor, gyroskop, nærhetssensor, og en pupillavstandssensor [20].

3.2.2. Unity

Unity er en populær spillmotor som er basert i C# programmeringsspråket. Spillmotoren har støtte for både 3D og 2D platformer og er tilgjengelig på mange forskjellige operativsystemer. Unity har også støtte for VR [21]. Prosjektets struktur følger Unity sin prosjektstruktur. Mer informasjon om dette kan finnes i systemdokumentasjonen (se vedlegg 5).

Det programmeringsspråket gruppa hadde mest erfaring med var Java. Siden Unity bruker C#, som har veldig mange fellestrekke med Java, ville ikke gruppa trenge å bruke mye tid på å lære seg et nytt programmeringsspråk. Gruppa fikk også beskjed fra veileder og oppdragsgiver om å bruke Unity siden det var den spillmotoren datamaskinen og VR-utstyret gruppa fikk tildelt var satt opp til å bruke.

3.2.3. OpenXR

Tidligere prosjekt som tar i bruk øyesporing har som oftest valgt å bruke OpenVR, et programgrensesnitt utviklet av Valve som gir brukere tilgang til maskinvaren til tross for at applikasjonene ikke har alle spesifikasjonene til maskinvaren [22]. Dette gjør det mulig for program, som Unity, å ta i bruk HTC Vive Pro Eye uten å måtte laste ned ytterlige programvarer.

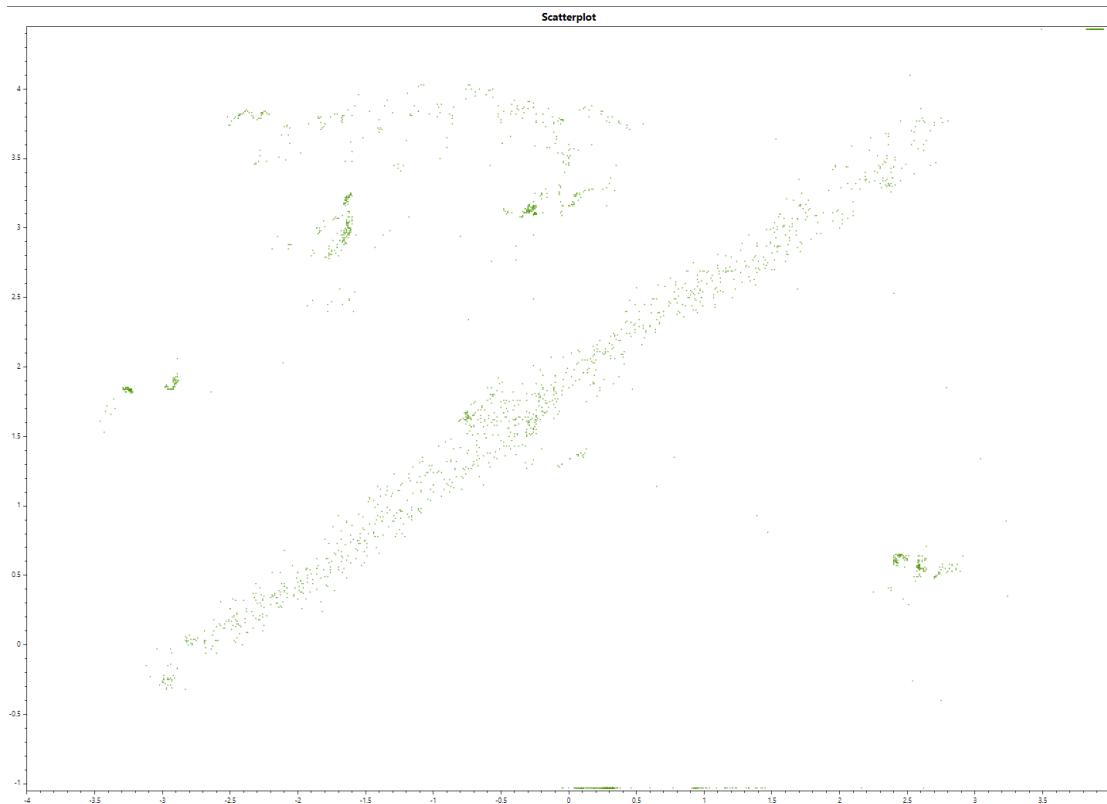
Unity har derimot nylig erstattet OpenVR med et nyere programgrensesnitt, OpenXR, som er utviklet av Khronos [23]. OpenXR skal støtte et større spektrum av blandet virkelighet og har et mål om forenkle VR programvareutvikling slik at flere applikasjoner kan nå ut til flere plattformer uten å måtte skrive om koden flere ganger [24].

Det ble understreket at oppgaven skulle være en prototype og videreutvikles i fremtiden. Som et resultat av dette valgte gruppa å bruke OpenXR da dette er et nyere programgrensesnitt som fortsatt oppdateres i dag av utviklere.

3.3. Visualisering

3.3.1. Spredningsplass

Dataen angående øyesporing ble lagret som en liste med punkter. Disse punktene ble plottet inn i et spredningsplass for å visualisere hvor pasienten så i løpet av testen. Spredningsplassen kan i tillegg bli brukt sammen med et heatmap for å verifisere og analysere hvor i eksperiment-området pasienten hadde hyppigst øyebruk.

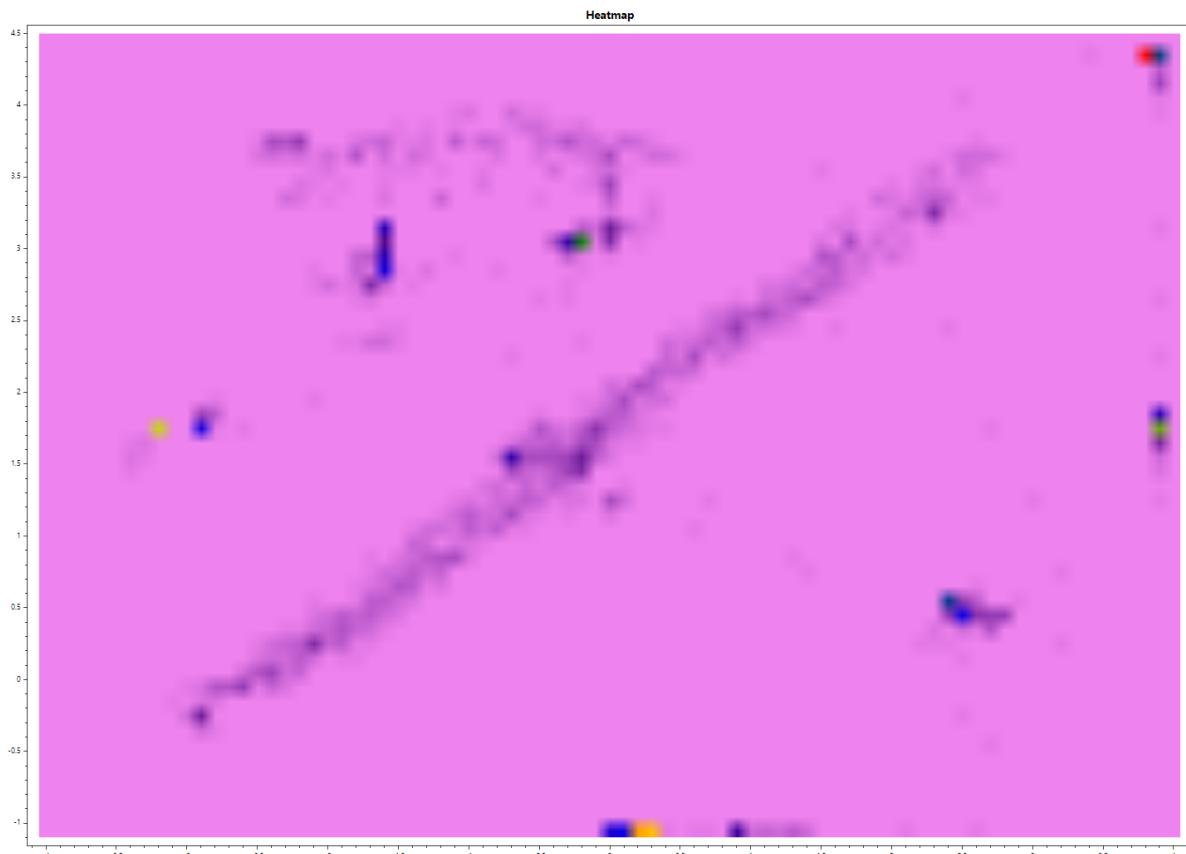


Figur 3.1: Eksempel på et spredningsplass

Figur 3.1 viser et eksempel på hvordan et spredningsplass kan se ut etter at man har kjørt et eksperiment. I likhet med alle grafene, er x- og y-aksen bestemt ut fra størrelsen og plasseringen til eksperiment-området i Unity-miljøet. Dette er fordi aksen skal lettere korrespondere med eksperiment-området, og om området blir endret eller skalert vil man ikke trenge å endre på plotting og visualisering.

3.3.2. Heatmap

Et heatmap er en visualiseringsteknikk hvor felt i et koordinatsystem farges inn etter hvor mange punkter som faller innen det samme området. Heatmap er brukt i oppgaven slik at man kan visualisere hvor i rommet en person har hyppigst øyebruk. Et heatmap har ulike graderinger som viser hvor mange punkter man har i et område; jo sterkere farge det er i grafen, jo flere punkter har man i området.



Figur 3.2: Eksempel på et varmekart

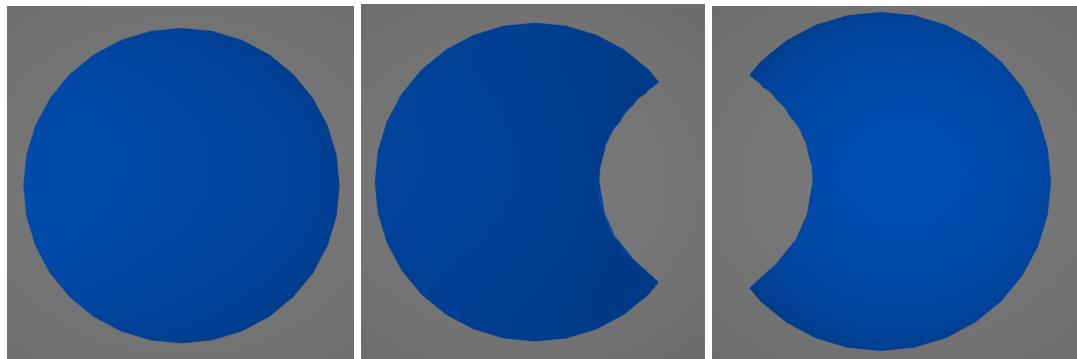
Figur 3.2 er et eksempel på hvordan et varmekart kan se ut etter et eksperiment. I figur 3.2 er x- og y-aksen begrenset likt som i figur 3.1.

3.3.3. Plott av objekter

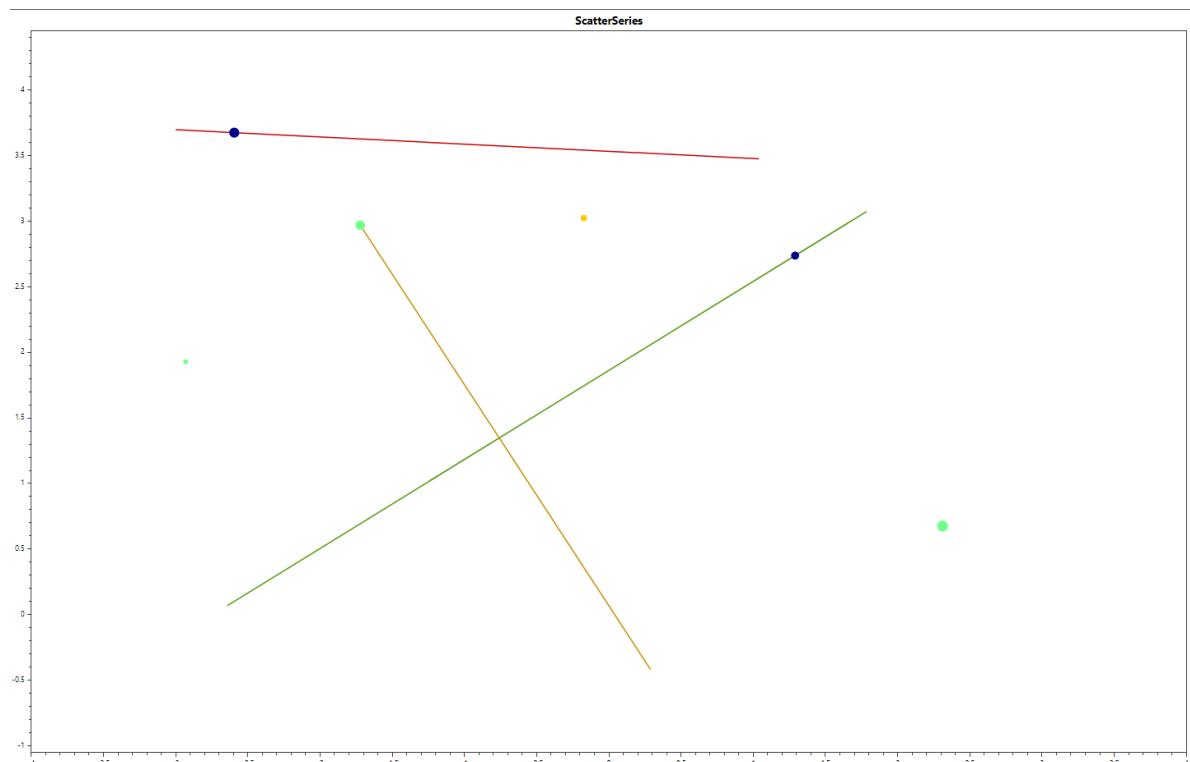
Denne grafen ble opprettet som et hjelpeverktøy for visualisering av objektene i eksperimentet. Dette gjør det mulig å se en visualisering av eksperimentet uten å måtte åpne opp Unity-miljøet etter at man har gjennomført et eksperiment. Her kan man også se objektets bane, hvor på banen objektene ble registrert, og hvilke objekter som ble registrert av pasienten. Ved å visualisere objektene på en graf, kan man også letttere sammenligne data fra heatmap og spredningsplot med det faktiske eksperimentet.

Eksperimentet har 3 typer objekter:

- Fullt-objekt: En full kule
- Venstre-side-objekt: En kule som mangler en bit på høyre side
- Høyre-side-objekt: En kule som mangler en bit på venstre side



Figur 3.3: Fullt-objekt, venstre-side-objekt, høyre-side-objekt



Figur 3.4: Eksempel på visualisering av objekter

Figur 3.3 viser et eksempel på hvordan et plott av objektene i eksperimentet kan se ut. Forskjellige farger på punktene viser til forskjellige egenskaper til objektene som ble brukt i eksperimentet.

- **Linje:** En representasjon av bevegelse fra startpunkt til sluttpunkt. Selv om det er forskjellige farger på linjene, så har det ingen betydning og er et resultat av plugin som ble brukt for visualisering.
- **Grønn kule:** Posisjon av hvor et fullt objekt ble sett.
- **Lyseblått kule:** Posisjon av hvor et høyre-side-objekt ble sett.
- **Mørkeblått kule:** Posisjon av hvor et venstre-side-objekt ble sett.
- **Lyse oransje kule:** Et høyre-side-objekt som har ikke blitt sett.
- **Mørke oransje kule:** Et venstre-side-objekt som har ikke blitt sett.
- **Rød kule:** Et fullt objekt som har ikke blitt sett.

I tillegg har punktene i grafen forskjellig størrelse avhengig av hvor store objektene var i eksperimentet i Unity-miljøet.

4. Resultater

I dette kapitlet blir resultatene til prosjektet fremstilt.

4.1. Ingeniørfaglige resultater

Først og fremst blir de ingeniørfaglige resultatene presentert. Dette innebærer de funksjonelle kravene som ble oppgitt i visjonsdokumentet (se vedlegg 3), samt ytterligere funksjonelle krav som ble satt av oppdragsgiver etter demonstrering. Prosjektets kildekode kan finnes i vedlegg 6.

4.1.1. Funksjonelle krav fra visjonsdokument

Funksjonelle krav	Oppfylt	Ikke oppfylt
Funksjonalitet for datainnsamling	x	
Funksjonalitet for å opprette tester i VR-miljøet.	x	
Et bibliotek av ferdiglagde tester	x	

Tabell 4.1: Tabell over resultat av funksjonelle krav fra visjonsdokumentet

Det ble som nevnt tidligere ikke satt mange funksjonelle krav. Dette er fordi disse funksjonelle kravene innebærer å lage grunnmuren til et større program og involverer mye funksjonalitet, og er dermed ganske omfattende. Eksempelvis vil kravet om funksjonalitet for å opprette tester i VR-miljøet inneholde å blant annet lage et eksperiment-område som en bruker kan designe eksperimenter i, lage et objekt som brukeren kan legge til eksperiment-området og lage menyer og funksjonalitet for å manipulere objektet på ulike måter. Ettersom gruppas utgangspunkt var tradisjonelle og todimensjonale tester, ble det brukt mye tid på å bygge miljøet fra grunnen av og utforske alternativer til hvordan testene kunne bli tolket i et VR-miljø.

Under kravet om et bibliotek av ferdiglagde tester handlet dette hovedsakelig om at brukeren skal kunne designe et eksperiment, lagre dette eksperimentet slik at brukeren kunne hente det frem i et bibliotek i fremtiden. I tillegg skulle biblioteket også inneholde noe som blir kalt en randomisert test. I motsetning til eksperimenter som blir designet av en spesialist og som foregår kronologisk, vil en randomisert test ha flere runder med objekter som dukker opp tilfeldig hvor spesialisten kan justere på en rekke parametere som påvirker testen på en egen skjerm i sanntid.

4.1.2. Funksjonelle krav fra bruker etter demo

Etter alle de initiale funksjonelle kravene ble innfridd hadde gruppa en demonstrering av programmet for oppdragsgivere. Her fikk også oppdragsgiver selv teste programmet og kommentere på hva som burde forbedres og nye funksjoner som de ville at gruppa skulle implementere.

Funksjonelle krav	Oppfylt	Ikke oppfylt
Mulighet til å bytte mellom å låse eksperiment-området til synet*		x
Flytte på pasienten under en test	x	
Kunne endre på størrelse til objekter	x	
Kunne bestemme når et objekt trigges	x	
Tooltips som forklarer knappene	x	

Tabell 4.2: Tabell over funksjonelle krav fra bruker etter demo

*Å låse eksperiment-området til synet førte til problemer med øyesporing. Ettersom hensikten med å låse pasientens syn til testområdet var for å rette deres oppmerksomhet til testen, var det mulig å implementere andre løsninger som ga samme resultat. Det ble i stedet implementert en knapp som nullstilte pasientens syn, slik at synet og oppmerksomheten ble rettet til eksperiment-området uansett hvor mye de hadde flyttet og rotert hodet.

4.1.3. Ikke-funksjonelle krav

Oppgavens ikke-funksjonelle krav kan bli funnet i visjonsdokumentet (se vedlegg 3).

Brukervennlighet (Usability):

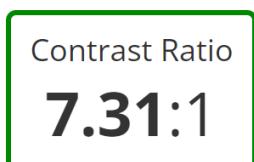
Oppgaven la stor vekt på brukervennlighet og enkel bruk. Gruppa har derfor strevd med å gjøre programmet så lett forståelig som mulig. Et eksempel på dette er implementeringen av tooltips. Ved siden av de fleste knappene og andre grensesnitt er det et lite spørsmålstege som brukeren kan klikke på. Når brukeren klikker på spørsmålsteget vil det dukke opp en boks som forklarer hva slags funksjon knappene har. Videre er det bilder og beskrivelser på veggen i rommet hvor brukeren lager tester som forklarer hvordan kontrollene fungerer.



Figur 4.1: Eksempel på et tooltip

Der brukeren har mulighet til å gjennomføre handlinger de muligens ville kunne angre på, som for eksempel å overskrive et lagret eksperiment eller starte på et nytt eksperiment mens de allerede lager eller redigerer et eksperiment, har gruppa implementert dialogbokser hvor brukeren må bekrefte valget sitt. På denne måten blir det gitt et valg til brukerne om de faktisk mente å gjøre det de trykket på, eller om de vil angre og gå tilbake. Brukerne unngår dermed å havne i situasjoner hvor de har tapt arbeid eller data grunnet et feiltrykk.

Til slutt er det veldig liten bruk av farger i programmet. Der hvor det er blitt brukt farger har gruppa vært bevisste med å bruke farger som har høy kontrast for å vise hensyn til svaksynte og fargeblinde. Noen knapper veksles mellom å være av eller på. Når knappen viser at noe er på, vil den få en blå farge. Kontrasten har blitt sjekket og har bestått WCAG Level AAA testen [25].



Normal Text

WCAG AA: **Pass**
WCAG AAA: **Pass**

The five boxing wizards jump quickly.

Figur 4.2: WCAG-test for blå knapper



Figur 4.3: Bilde av knapp for å illustrere kontrast

Pålitelighet (Reliability):

Et problem med den tidligere prototypen var at programmet ofte krasjet tilfeldig slik at det ikke var mulig å bruke programmet. Gruppa har derfor strevd etter å lage et stabilt og pålitelig produkt med feilhåndtering som unngår at brukeren uheldigvis skal krasje programmet. Dette har blant annet blitt gjort ved å deaktivere og skjule visse UI-elementer i programmet når visse prosesser er aktive for å hindre at brukeren gjør noe som muligens kan føre til problemer. Gruppa har også prøvd å minimere tekst-input

der det gir mening, siden dette kan være en kilde til feil og problemer dersom input for brukeren ikke håndteres riktig. Tekst-input krever en del validering for å sørge for at det er av den riktige datatypen for det dataen skal brukes til. Dersom ikke alle unntakstilfeller håndteres kan programmet ende opp med å motta data som ikke er av riktig format, noe som kan føre til problemer. Gruppa har i stedet valgt å bruke andre UI-elementer som for eksempel knapper, slidere og pluss-minus-kontrollere.

Ytelse (Performance):

Gruppa har prøvd å skrive kode slik at det ikke var mye repetisjon og unngå kode som kan henge opp lett. Gruppa har ikke opplevd at programmet har stoppet opp eller krasjet. Det er også kun brukt objekter med lav kompleksitet og enkel grafikk for å sørge for at programmet kjører problemfritt.

Støttebarhet (Supportability):

Det understrekkes at oppgaven var en prototype og at skalerbarhet var av stor betydning. Koden i prosjektet er derfor grundig dokumentert. Det skulle også være enkelt for andre å kunne fortsette å jobbe på prosjektet. For å oppnå dette, har blant annet gruppa brukt Unity Prefabs der det er passende. Et eksempel på dette er eksperiment-området, som potensielt er noe som andre vil kunne endre på og justere ved videre arbeid. Siden dette objektet er en Unity Prefab, vil det endre seg alle steder der hvor den blir brukt ved å kun endre det på en plass. På denne måten slipper man å endre eksperiment-området alle steder det er brukt dersom man ønsker å justere det. Dette systemet har også blitt brukt på blant annet tooltips, som er naturlig å bruke flere ganger dersom ny funksjonalitet og parametre legges til.

4.1.4. Svar fra brukertester

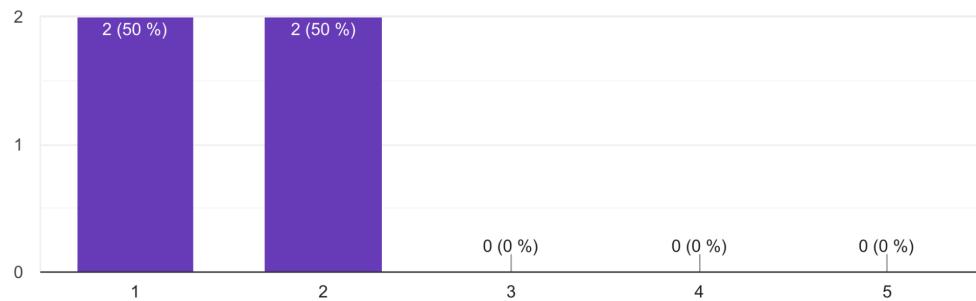
Ettersom brukervennlighet og enkel bruk var noe som var høyt verdsatt i oppgaven, var det naturlig for gruppa å ha noen brukertester. Her ble det lagt størst vekt på hvordan det var å bruke programmet og hvor godt testeren skjønte hvordan man utførte oppgaver. Siden det var hovedsakelig oppdragsgiverens oppgave å be om spesifikk funksjonalitet, unngikk gruppa å stille spørsmål om hva slags funksjoner testerne syntes manglet med programmet, men de hadde likevel muligheten til å komme med forslag og endringer hvor det virket naturlig.

Oppgaven testerne fikk var å sette ut flere objekter, hvor halvparten av disse skulle ha bevegelse med varierende fart og ulike modeller skulle bli brukt. Til slutt skulle testeren lagre testen. Før de skulle løse oppgaven, fikk testerne en rask gjennomgang av hvordan kontrollene fungerte og hvordan programmet fungerte. Det var også muligheter for dem til å stille spørsmål i løpet av testen.

Gruppa inviterte tilfeldige personer og det første naturlige spørsmålet var derfor hvor mye erfaring de hadde med VR, hvor 1 betydd at de aldri har brukt VR, mens 5 betydd at de bruker VR regelmessig. Mesteparten hadde lite eller ingen erfaring med VR. Siden det er antatt at de endelige brukerne ikke har noe tidligere erfaring med VR vil derfor svarene til testerne angående brukervennlighet kunne være representativt med hvor lett de endelige brukerne vil synes programmet er.

Hvor mye erfaring har du i VR?

4 svar

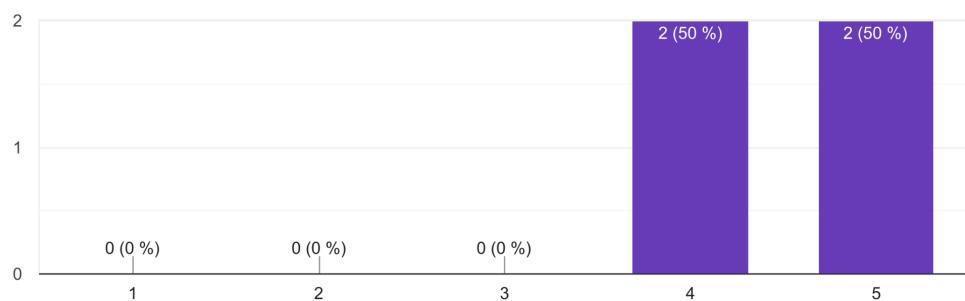


Figur 4.4: Resultater fra det første spørsmålet i brukertesten

Etter de hadde testet programmet og utført oppgaven ble de spurta om hvor lett det var å bruke programmet, hvor 1 betydd at programmet var vanskelig og uforståelig å bruke, mens 5 betydd at programmet var lett og forståelig å bruke. Til tross for at ingen av testerne hadde noe særlig erfaring med å bruke VR syntes de likevel at programmet var lett og forståelig å bruke.

Hvor lett og forståelig var det å bruke programmet?

4 svar



Figur 4.5: Resultater fra det andre spørsmålet i brukertesten

Videre ble testerne spurta om hva som gjorde programmet vanskelig å bruke. Problemer og spørsmål ble også utdypet og nevnt i løpet av testingen. Et problem som gikk igjen var at testerne syntes det var vanskelig å sikte på menyen. Dette ble løst ved å justere på posisjonen til menyen slik at brukerne ikke måtte anstrengte seg like mye for å sikte.

Hva gjorde programmet vanskelig å bruke?

4 svar

Programmet virket litt klønrete. Det var vanskelig å sikte.

Jeg hadde lite erfaring i VR fra før av

Vanskelig å sikte på menyen ved håndleddet

Det var ikke så bra dybdesyn og noen sliders hadde ikke navn så jeg visste ikke hva de gjorde

Figur 4.6: Resultater fra det tredje spørsmålet i brukertesten

Testerne ble også spurta om hva de likte med programmet. Dette hjalp gruppa med å vite hva de har implementert som gjør programmet brukervennlig, slik at fremtidige funksjoner kan bli implementert med lignende løsninger. Her har testerne lagt vekt på at oversiktligheten gjør programmet lettere å bruke og gir en bedre opplevelse.

Hva likte du spesielt godt med programmet?

4 svar

Programmet var brukervennlig og det var lett å finne frem til ulike ting

Det var lett å bruke

Alt føltes oversiktig og brukte ikke mye tid på å lete seg frem

Veldig intuitivt med en meny som redigerte bevegelse og programmet var veldig fleksibelt

Figur 4.7: Resultater fra det fjerde spørsmålet i brukertesten

Til slutt ble testerne gitt en mulighet til å gi ytterligere kommentarer om brukertesten. Her nevnte nesten alle at deres største utfordring var å bli kjent med kontrollene og programmet, men å navigere seg rundt og løse oppgaven etter dette var ingen utfordring. Med tanke på at de endelige brukerne vil få et introduserende kurs i programmet vil man kunne anta at disse problemene vil bli løst.

Hvordan var det ellers å løse oppgaven?

4 svar

Det var lett å løse oppgaven når jeg forsto kontrollene

Jeg brukte litt tid på å bli kjent med kontrollene, men det var ganske rett frem etter det

Gikk fint å løse oppgaven og var ikke noen spesiell utfordring

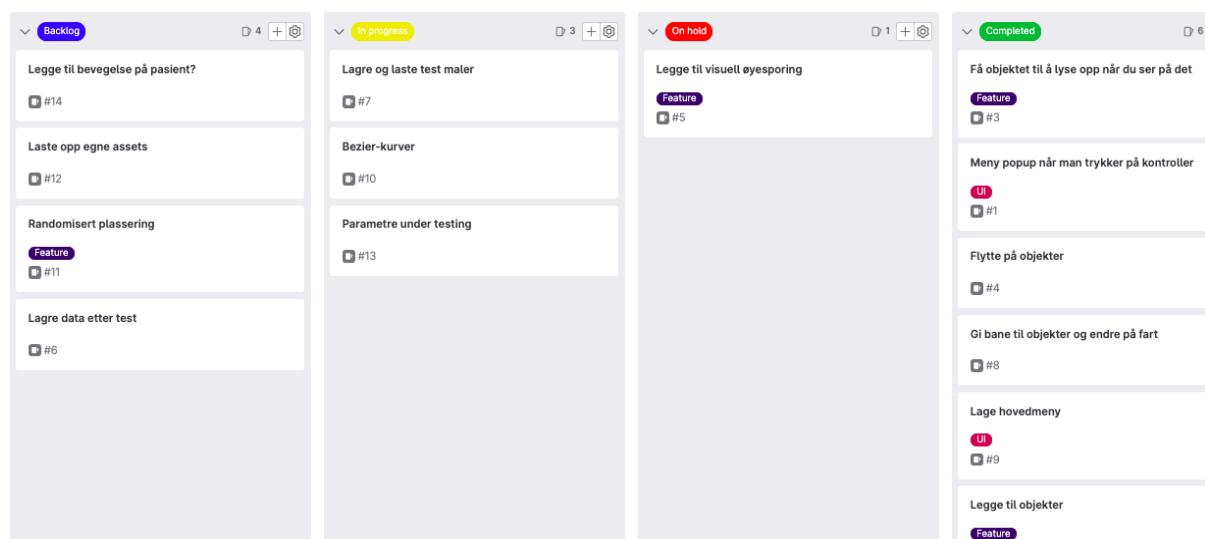
Var veldig lett etter man fikk forklart det

Figur 4.8: Resultater fra det femte spørsmålet i brukertesten

4.2. Administrative resultater

4.2.1. Kanban-tavle

Gruppa brukte Kanban og oppdaterte tavla daglig. Ved starten av prosjektet la gruppa til alle funksjonelle krav og la videre til flere funksjoner som skulle implementeres underveis. Disse oppgavene ble delt inn i "Backlog" som var en liste over tilgjengelige oppgaver, "In progress" som var oppgaver gruppa jobbet med for øyeblikket, "On hold" som var oppgaver gruppa hadde startet med, men måtte sette på vent for å for eksempel prioritere andre oppgaver og til slutt "Completed" som var ferdig implementerte funksjoner.

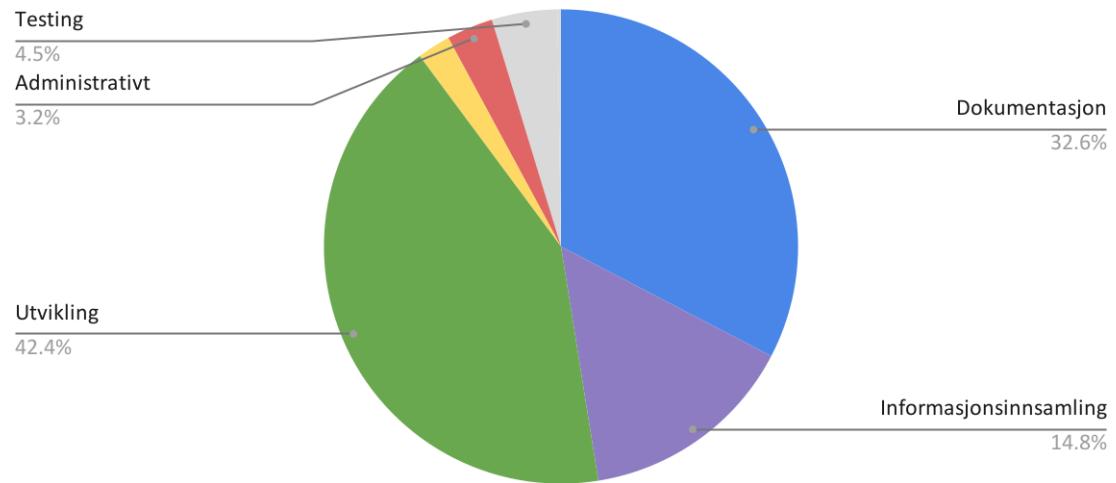


Figur 4.9: Gruppas Kanban-tavle midt i prosjektet

Som en anbefaling fra gruppas veileder ble det ikke lagt et Gantt-diagram. Det ble derimot laget en milepæl liste som var en liste med mål som gruppa hadde satt for seg selv for når de ville bli ferdig med ulike aktiviteter. Denne lista kan man finne i gruppas forprosjektplan (se vedlegg 1).

4.2.2. Timeregnskap

Diagrammet under viser hvor stor andel av tiden gruppa brukte på hver enkel aktivitet samlet sett. Ytterlige timelister, statusrapporter og møtereferat refereres til prosjekthåndboka (se vedlegg 2).



Figur 4.10: Diagram som viser andel tid brukt per aktivitet

5. Diskusjon

I dette kapitlet vil resultatet bli drøftet.

5.1. Diskusjon av ingeniørfaglige resultater

5.1.1. Funksjonelle krav

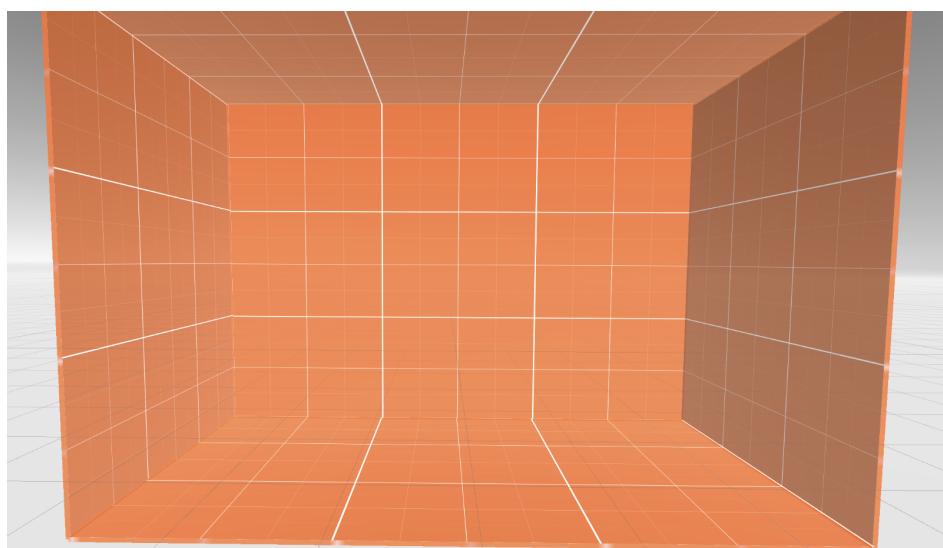
Produktet var ment til å være en prototype og det ble derfor satt få funksjonelle krav utenom de mest grunnleggende funksjonene programmet skulle ha. Dette bidro med å lage et stabilt og pålitelig program som kan effektivt videreutvikles, før gruppa videre implementerte flere funksjoner. Da gruppa hadde implementert alle de funksjonelle kravene i visjonsdokumentet (se vedlegg 3), var det likevel god tid igjen av prosjektet. Gruppa hadde derfor et møte og en demonstrasjon av produktet med oppdragsgivere og forespurte flere krav og funksjoner.

5.1.1.1. Hovedmeny

Når programmet først åpnes kommer man til en hovedmeny. Her kan man navigere videre til VR-miljøene for oppretting og design av eksperimenter, gjennomføring av disse eksperimentene, samt miljøet for gjennomføring av randomiserte eksperimenter. Kalibrering av øyesporingen og åpning av mappen hvor resultatdata lagres er også mulig her.

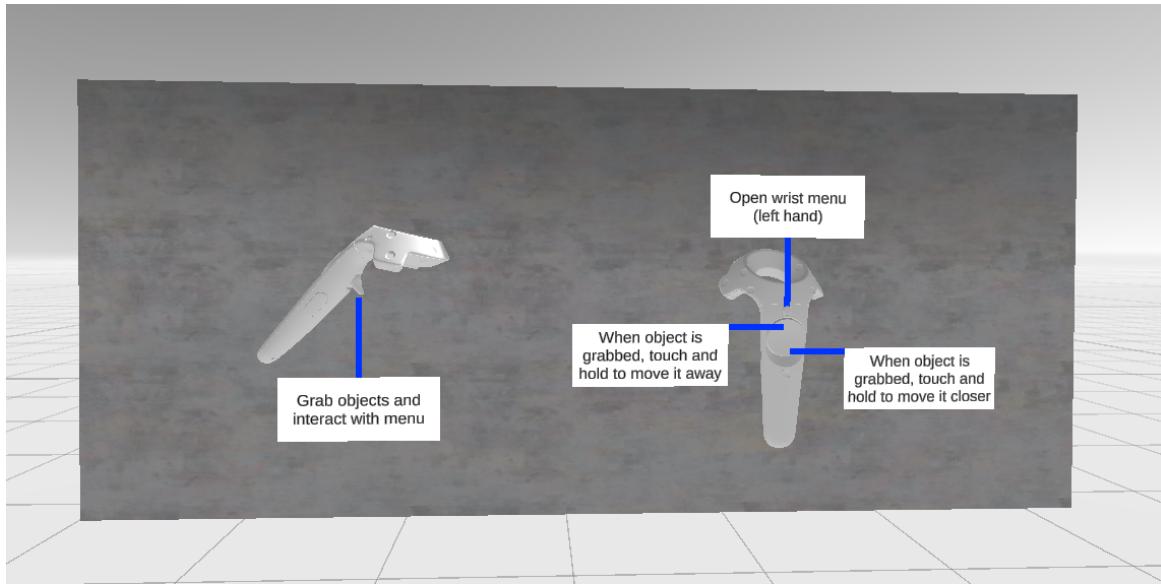
5.1.1.2. Oppretting og design av eksperimenter

VR-miljøet for oppretting og design av eksperimenter består av tre hovedelementer: en meny for lagring og lasting av eksperimenter, eksperiment-området, og en vegg med informasjon om hvordan kontrollene brukes. Eksperiment-området er en stor boks med en åpen side. Denne boksen visualiserer eksperimentets grenser. Selve opprettingen av et eksperiment vil altså foregå inne i denne boksen.



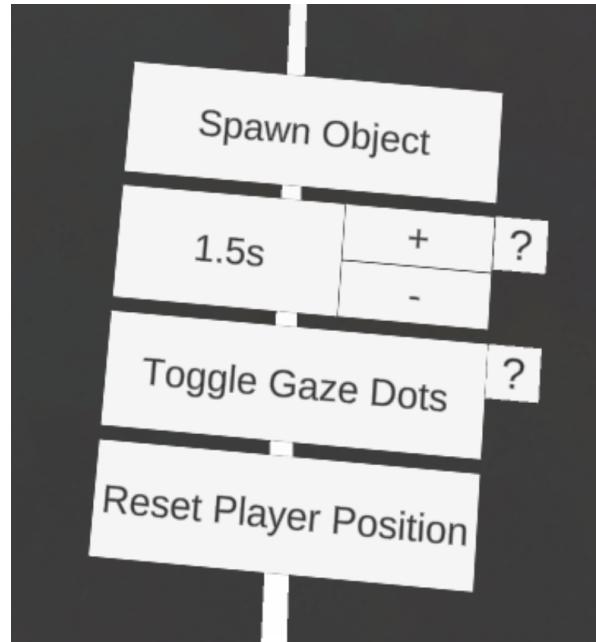
Figur 5.1: Eksperiment-området

Til høyre for eksperiment-området kan man se veggen med informasjon om hvordan bevegelseskontrollene brukes og hva de forskjellige knappene gjør. På veggen er det forklart hvordan man åpner menyen på kontrollen og hvordan man velger objekter og trykker på knapper i menyen.



Figur 5.2: Vegg som viser hvordan kontrollene brukes

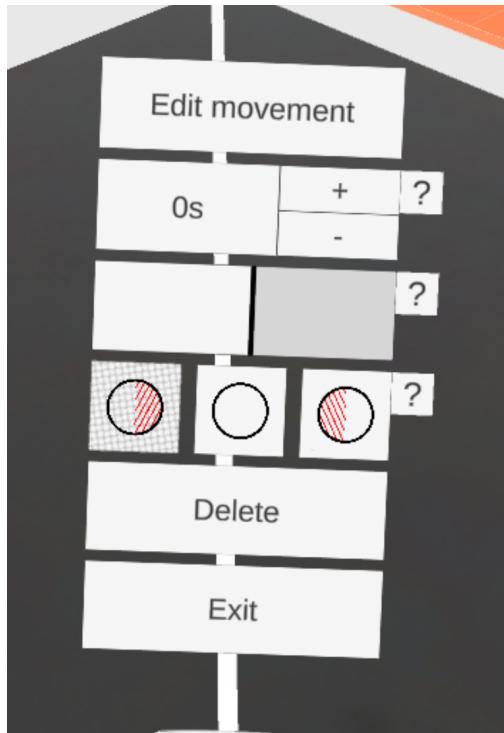
Menyen på venstre kontrollen inneholder alle parameterne som man kan endre til eksperimentet. Det er her man legger til objekter i eksperiment-området, skrur på og av visualisering av øyesporing via blikk-prikker, endrer på hvor lang tid pasienten skal se på objektet før det blir registrert at det har ble sett på, og nullstiller posisjonen til spilleren til midten av plattformen.



Figur 5.3: Meny på venstre kontroll

For å legge til eksperiment-objekter i eksperimentet, kan man trykke på "Spawn Object"-knappen, og et objekt vil da dukke opp midt i eksperiment-området. Dette

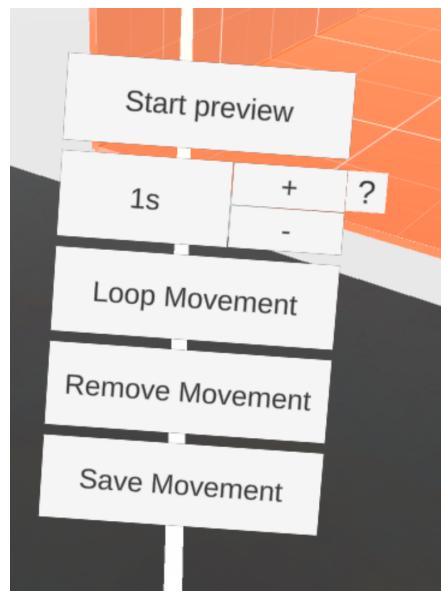
objektet kan flyttes på ved å holde inne velg-knappen (trigger) på en av kontrollene mens man peker på objektet, deretter er det bare bevege på hånda for å flytte objektet. Samtidig vil også en ny meny komme opp over den venstre kontrollen. Denne menyen er brukt til å endre objektets forskjellige parametere og egenskaper.



Figur 5.4: Meny av objektvariabler

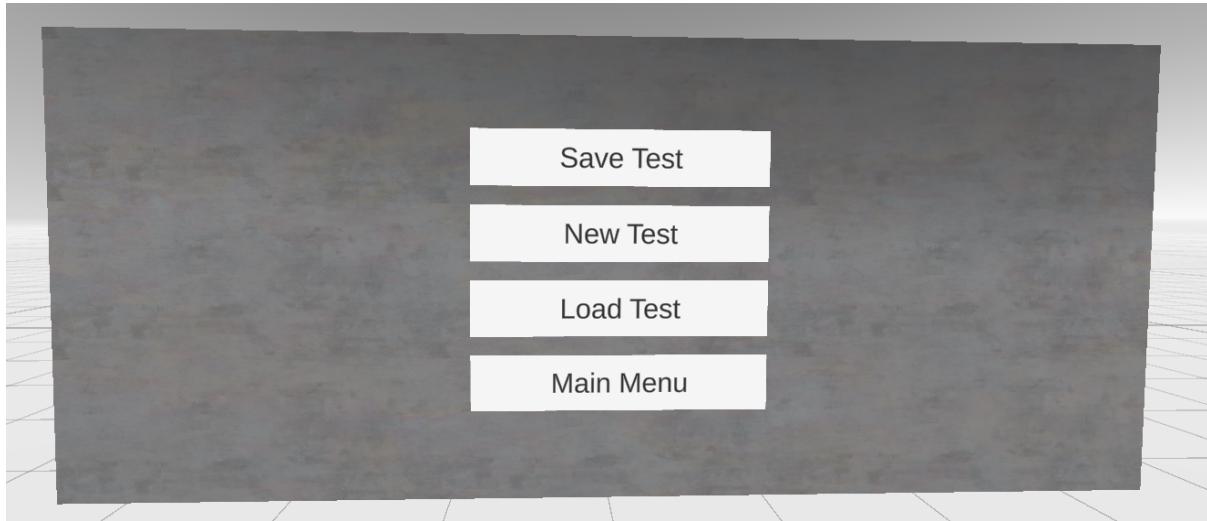
I denne menyen kan man velge å legge til bevegelse, endre på hvor langt tid det skal gå før objektet dukker opp etter eksperimentets start, endre på størrelse, hva slags type objekt det skal være, og eventuelt slette objektet. Ettersom oppgaven tar utgangspunkt i epletesten, vil brukeren ha mulighet til å legge til objekter som mangler en bit på høyre- eller venstresiden i likhet med at noen epler mangler en bit i epletesten. Om man velger å legge til bevegelse til objektet vil en hvit kule dukke opp i eksperiment-området, samtidig som det vises en ny meny over den venstre kontrollen. Denne hvitekulen representerer sluttpunktet i eksperiment-objektets bevegelsesbane, og kan flyttes på samme måte som eksperiment-objektet.

Den nye menyen inneholder innstillinger som gjelder objektets bevegelse. Her kan man forhåndsvise bevegelsesbanen til objektet, endre hvor langt tid objektet skal bruke fra startpunkt til sluttpunkt, om bevegelsen skal være i en løkke, fjerne bevegelsen og lagre bevegelsen.



Figur 5.5: Meny for endring av objektets bevegelse

Til venstre for eksperiment-området er det en vegg med noen knapper. Denne veggen inneholder knapper for å lagre eksperimentet, starte et nytt eksperiment, laste inn eksperiment som var lagret fra før, og en knapp for å gå tilbake til hovedmenyen. Når man velger å lagre et eksperiment, vil man kunne velge et navn for eksperimentet ved hjelp av et virtuelt tastatur. Dersom man trykker på lagre-knappen, vil eksperimentet lagres i en fil. Dersom man redigerer et eksperiment som allerede har blitt lagret, har man også muligheten til å overskrive det gamle eksperimentet med de nye endringene.



Figur 5.6: Meny over lagring og lasting av eksperiment

Menylene som kontrollerer eksperiment-objektene er de menyene i VR-miljøet brukeren vil ha flest interaksjoner med. På grunn av dette er det viktig at disse menyene er posisjonert på en måte som gjør det lett å bruke den. Ved å plassere disse menyene på en av kontrollene kan brukeren selv holde menyene i en posisjon som de føler er naturlig og komfortabel. Dette gjør det også mulig for brukeren å kunne se hvordan endringene de foretar påvirker eksperimentet i sanntid uten å måtte snu på hodet ved å holde menyene nærmere objektet de endrer på. Dersom disse menyene hadde vært plassert på en av sidene av eksperiment-området, ville brukeren muligens måtte snu på hodet for å

se hvordan endringene de nettopp foretok påvirket eksperimentet. En annen mulighet hadde vært å ha det foran eksperiment-området, men det kan føre til at menyen var i veien for å lage eksperimentet, så gruppa valgte ikke å ha det heller.

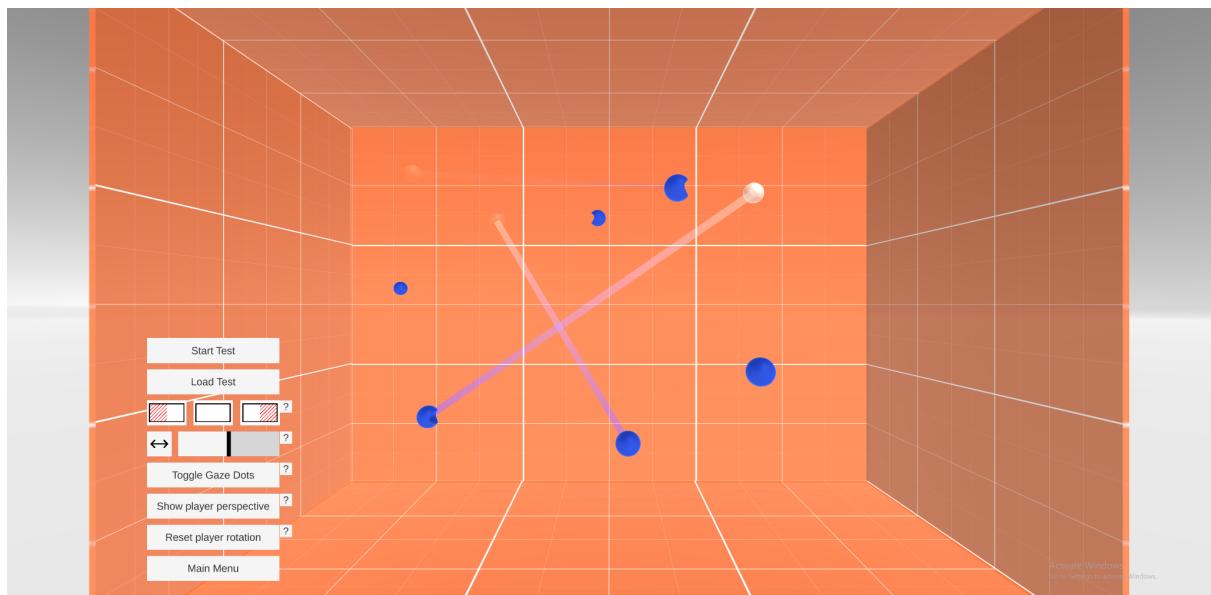
Menyen for lagring og lasting av eksperimenter vil brukeren derimot ha mange færre interaksjoner med. Denne menyen har heller ikke noe funksjonalitet som direkte påvirker eksperimentet, og det er dermed ikke nødvendig å plassere den på en av kontrollene.

5.1.1.3. Gjennomføring av eksperimenter

Det er to typer eksperimenter det er mulig å gjennomføre: spesiallagde eksperimenter og randomiserte eksperimenter. Begge disse eksperimentene gjennomføres ved at pasienten tar på seg VR-headsettet mens eksperimentet kontrolleres fra en eksterne PC-skjerm. Pasienten vil dermed ikke kunne se menyvalgene som er tilgjengelige for personen som kontrollerer gjennomføringen av eksperimentet. Det er gjort på denne måten for å fjerne alle mulige distraksjoner for pasienten slik at de lettere kan fokusere fullstendig på eksperimentet.

Gjennomføring av spesiallagde eksperimenter:

Et spesiallaget eksperiment kan gjennomføres ved å velge det fra en liste på hovedmenyen. VR-miljøet for gjennomføring av eksperimenter vil så åpne seg, og brukeren vil få tilgang til en meny for å kontrollere eksperimentet. På denne menyen er det også mulig å laste inn et nytt eksperiment, slik at man slipper å gå tilbake til hovedmenyen dersom man ønsker å gjennomføre enda et eksperiment.



Figur 5.7: Meny ved gjennomføring av spesiallaget eksperiment

På denne menyen har brukeren tilgang til følgende funksjoner:

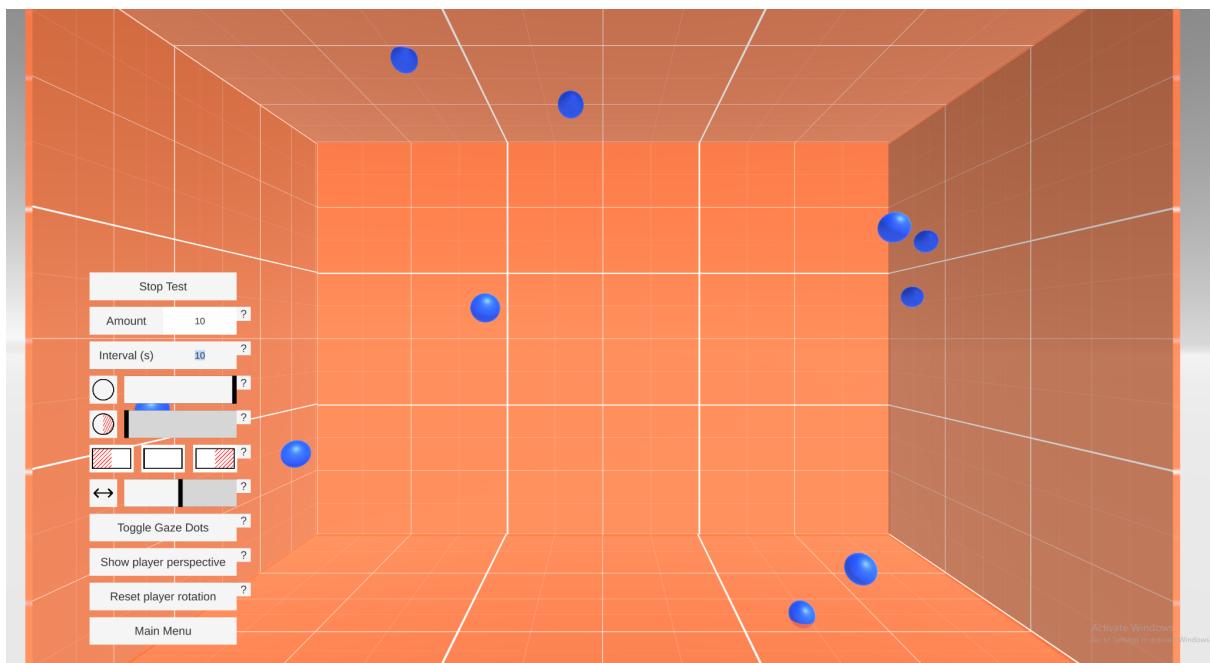
- Starte og stoppe eksperimentet.
- Laste inn et annet eksperiment.
- Skjule alle eksperiment-objektene på høyre eller venstre halvside av eksperiment-området.
- Justering av pasientens avstand til eksperiment-området.

- Visualisering av pasientens øyebevegelser via blikk-prikker (gaze dots)
- Vise pasientens perspektiv.
- Snu pasienten mot eksperiment-området.
- Gå tilbake til hovedmenyen.

Dersom eksperimentet blir startet, vil en nedtelling begynne. Når nedtellingen er fullført vil selve eksperimentet begynne. Eksperimentet vil pågå helt til det blir manuelt stoppet av brukeren, hvor en meny som gir brukeren en mulighet til å lagre resultatene av eksperimentet dukker opp.

Gjennomføring av randomiserte eksperimenter:

I likhet med spesiallagde eksperimenter kan VR-miljøet for gjennomføring av randomiserte eksperimenter åpnes fra hovedmenyen. Et randomisert eksperiment består av flere runder hvor et brukervalgt antall eksperiment-objekter dukker opp på tilfeldige posisjoner inne i eksperiment-området. Etter et bestemt tidsintervall vil objektene forsvinne og en ny runde med slike objekter dukker opp. Dette vil fortsette kontinuerlig helt til eksperimentet stoppes. Også for denne typen eksperiment vil brukeren få tilgang til en meny for å kontrollere eksperimentet.



Figur 5.8: Meny ved gjennomføring av randomisert eksperiment

Denne menyen inneholder det meste av funksjonaliteten tilgjengelig for spesiallagde eksperimenter, men har også noe ekstra funksjonalitet som kontrollerer hver runde av randomiserte objekter. Denne funksjonaliteten er som følger:

- Antall objekter som skal dukke opp på tilfeldige posisjoner i eksperiment-området.
- Hvor lang tid det tar før en ny runde med randomiserte objekter skal dukke opp.
- Hvor stor prosentandel av alle objektene som skal være fulle objekter.
- Hvor stor prosentandel av alle gjenværende objekter som skal være venstre-side-objekter.

Endring av disse parametrerne vil gjelde den neste kommende randomiserte runden og ikke den nåværende runden.

Start og stopp av eksperimentet foregår på samme måte som ved spesiallagde eksperimenter, og brukeren vil få muligheten til å lagre resultatdata når eksperimentet er stoppet og fullført.

5.1.1.4. Datalagring

For å oppfylle de funksjonelle kravene satt i visjonsdokumentet (se vedlegg 3) om datainnsamling og et bibliotek av ferdiglagde tester, ble det implementert et system for datalagring. Dette systemet inkluderer både lagring av opprettede eksperimenter og lagring av innsamlet resultatdata ved gjennomføring av et eksperiment. All data lagres i AppData-mappen på brukerens PC. Denne mappen brukes for å holde innstillinger og data for applikasjoner på en PC [26].

Lagring av eksperimenter:

Etter at et eksperiment har blitt opprettet, kan det lagres slik at det er mulig å fortsette å jobbe på det senere eller slik at det kan gjennomføres på en pasient. For å oppnå dette, må alle parametere og verdier brukeren har muligheten til å endre lagres. Det lagres dermed en liste over alle eksperiment-objektene innen eksperiment-området, hvor hvert element i listen inneholder følgende verdier:

- **startPosition:** En tredimensjonal vektor som viser posisjonen til objektet.
- **endPosition:** En tredimensjonal vektor som viser sluttposisjonen til objektets bevegelsesbane.
- **scale:** En tredimensjonal vektor som representerer objektets størrelse.
- **moveTime:** Tiden det tar (i sekunder) for objektet å bevege seg langs hele bevegelsesbanen.
- **startDelay:** Tiden det tar (i sekunder) før objektet dukker opp i eksperimentet.
- **hasMovement:** Boolsk variabel som bestemmer om objektet har en bevegelsesbane.
- **loopMovement:** Boolsk variabel som bestemmer om objektet skal gjenta bevegelsen langs bevegelsesbanen.
- **objectType:** Tekst som sier hvilket type objekt det er.

I tillegg til disse verdiene som lagres for hvert eneste objekt, lagres det også en verdi som gjelder hele eksperimentet:

- **visionDetectionTime:** Tiden det tar før det registreres at et objekt har blitt sett på.

Disse verdiene er alle verdiene som trengs for å kunne gjenskape eksperimentet slik at det kan lastes inn i VR-miljøet. Denne dataen lagres i JSON-filformatet. Grunnen til at dette filformatet blir brukt er at JSON brukes for å serialisere enkle datastrukturer, som lister og navn/verdipar [27]. Dette egner seg godt for det som trengs å lagres om eksperimentene, siden det kun trengs å lagre en liste som inneholder grunnleggende datatyper som blant annet tekst, boolske variabler, og tall. Dersom det blir ønskelig å lagre flere verdier, er dette lett å legge til da dette kan gjøres ved å legge til flere datafelt i klassen som representerer et eksperiment-objekt.

Lagring av resultatdata:

Når et eksperiment gjennomføres, er det hovedsakelig to datasett som lagres: data om

øyesporingen og data om eksperiment-objektene. Dataen om eksperiment-objektene lagres på samme måte og i samme format som eksperimentene. I tillegg til alle verdiene som lagres om objektene ved lagring av et eksperiment, lagres også disse verdiene:

- **positionWhenSeen:** En tredimensjonal vektor som viser posisjon til objektet når det ble registrert at det ble sett på.
- **hasBeenSeen:** Boolsk variabel som bestemmer om det har blitt registrert at objektet har blitt sett.
- **timePassedBeforeSeen:** Tid (i sekunder) det tok etter starten av eksperimentet før det ble registrert at objektet ble sett på.

Disse tilleggsverdiene lagres slik at det vil være mulig i fremtiden å gjenskape gjennomføringen av eksperimentet i VR-miljøet, som vil åpne opp muligheten for mer grundig analyse av pasienter. Denne dataen brukes også for å lage et plott over alle eksperiment-objektene som ble beskrevet i kapittel 3.3.3.

Data om øyesporingen lagres i en CSV-fil, hvor hvert kollisjonspunkt (en tredimensjonal vektor) mellom blikk vektorene og eksperiment-objekter og eksperiment-området skrives på en ny linje i filen. CSV-filer egner seg godt for lagring av større mengder data som kan hentes ut som en tabell eller liste [28]. Siden kollisjonspunktene skrives til CSV-filen i en rate lik programmets bildefrekvens kan det fort bli veldig mange datapunkter som lagres. Når gjennomføringen av eksperimentet fullføres, og brukeren velger å lagre dataen, blir all data om øyesporingen flyttet over til en ny fil med et brukervalgt navn. Flere grafer blir også opprettet basert på denne dataen, noe som er beskrevet i kapittel 3.3.

Lagring av resultatdata for randomisert eksperiment:

For hver runde med randomiserte eksperiment-objekter lagres det en liste over alle objektene, i likhet med det som gjøres ved lagring av resultatdata for gjennomføring av vanlige eksperimenter. I tillegg til denne listen lagres følgende verdier for hver runde:

- **amount:** Tall som sier hvor mange objekter som var i eksperiment-området.
- **interval:** Tid (i sekunder) den gjeldende randomiserte runden varte.
- **percentFullSpheres:** Prosent av totalt antall objekter som var fulle objekter.
- **percentLeftSpheres:** Prosent av gjenværende antall objekter som var venstre-side-objekter.
- **displayedSide:** Tekst som beskriver om bare objektene på venstresiden, høyresiden, eller om alle objektene ble vist.

Disse verdiene er parametre brukeren kan justere og endre på underveis i et randomisert eksperiment, og vil være nødvendige dersom eksperimentet skal gjenskapes og spilles av på nytt.

For hver randomiserte runde som kjøres, lagres det et tall som representerer denne runden. Etter at en runde er over, lagres dette tallet i CSV-filen med data om øyesporing. Dette gjør det mulig å skille mellom hvilken runde kollisjonspunktene hører til. Dersom man i framtiden også vil kunne gjenskape øyebevegelsene til pasienten, kunne det ha vært aktuelt å lagre tidspunktet til når hvert kollisjonspunkt dukket opp til CSV-filen. Dette vil gjøre det mulig å visualisere de faktiske øyebevegelsene som ble utført under eksperimentet i sanntid.

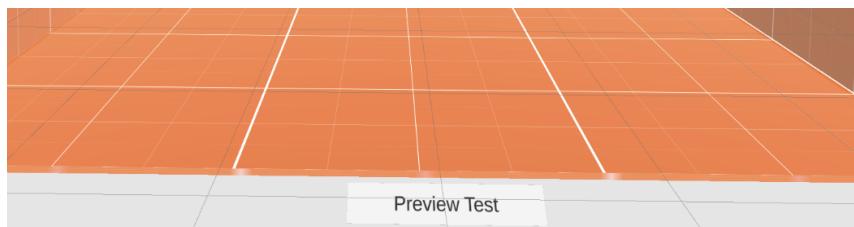
5.1.2. Ikke-funksjonelle krav

De funksjonelle kravene som ble lagt mest vekt på var brukervennlighet, pålitelighet og muligheter for videreutvikling. For å lage et pålitelig og utvidbart program er det viktigste å skrive en god og robust kode. Gruppa har også brukt god tid på å dokumentere koden slik at fremtidige utviklere vil lett kunne utvide eller feilsøke koden. Videre implementerte gruppa flere funksjoner som bidro til å lage et brukervennlig program. Eksempelvis valgte gruppa å legge til tooltips, som nevnt tidligere og kan ses i figur 4.1, på knappene som gjerne ikke anses som selvforklarende. Ettersom menyene er det mest omfattende i programmet, vil gjerne de mange valgene brukerne kan ta, virke overveldende. Ulike løsninger for å forklare disse menyene ble diskutert, blant annet å ha en bruksanvisning i hovedmenyen eller flere forklarende vegger. Disse løsningene er derimot tungvinte for brukeren og lite effektiv i motsetning til å ha en liten beskrivende knapp tilgjengelig til alle tider.

5.1.3. Svar fra brukertester

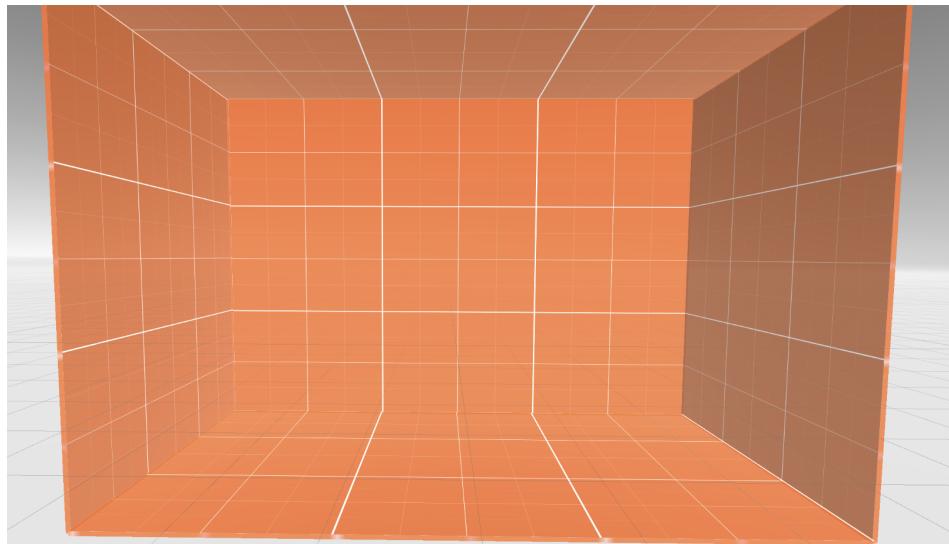
Gruppa fikk flere gode forslag fra brukertester som de valgte å ta videre. Forslagene var hovedsakelig løsninger som gjorde programmet mer brukervennlig.

Eksempelvis hadde gruppa allerede en knapp som gjorde det mulig å forhåndsvise bevegelser til enkeltobjekter. Flere testere etterspurte en knapp som gjorde det mulig å forhåndsvise hele testen, slik at man kunne se hvordan testen så ut og hvordan objektene bevegde på seg i forhold til hverandre, uten å måtte gå ut og starte testen. Denne funksjonen ville vært meget gunstig dersom man for eksempel ønsket at en objekts bevegelse skulle være avhengig av et annet objekt.



Figur 5.9: Knapp for forhåndsvisning av fullstendig test/eksperiment

Noen av testerne nevnte at rommet ikke ga en følelse av dybde slik det var tenkt. Tidligere hadde også oppdragsgiver foreslått å legge til et slags rutenett oppå eksperiment-området. Med tanke på forslaget fra oppdragsgiver også kunne hjelpe med tilbakemeldingene fra testerne, bestemte gruppa seg for å lage et tredimensjonalt rutenett til rommet slik at man både kan plassere objekter mer nøyaktig, men også gi et forsterket dybdesyn.



Figur 5.10: Eksperiment-området med rutenett

5.2. Diskusjon av administrative resultater

5.2.1. Kanban-tavle

Selv om Kanban-tavla hjalp med å organisere arbeidsoppgaver og planlegging, følte ikke gruppa at den hadde en stor effekt. For det første var gruppa ganske liten og det ble lettere å spørre og planlegge verbalt enn å måtte åpne tavla hver gang. I tillegg var det gitt få funksjonelle krav. Dette gjorde at gruppa gjerne kom på en ny funksjon i farta og da måtte de ta seg tid til å legge til funksjonen i tavla før de startet på oppgaven, noe som muligens kunne ha redusert effektiviteten over tid. En fysisk tavle kunne ha vært mer effektiv, da dette hadde gjort det lettere å skrive ned idéer i farta og gruppa hadde sluttet å måtte åpne nettsiden hver gang noen skulle sjekke tavla, samt oversikten over oppgavefordelingen hadde vært synlig til enhver tid.

5.2.2. Timeregnskap

Gruppa ble tildelt en arbeidsstasjon med utstyr som skulle bli brukt under utviklingen. Det var derimot mange tekniske problemer med utstyret ved starten av prosjektet som gjorde at utviklingen måtte bli utsatt flere uker. Dette ga gruppa en sen start på utviklingen og gjorde at gruppa måtte omgjøre timeplanen. Eksempelvis ble ikke programmet klart for brukertesting før det var for sent for å gjøre betydelige endringer basert på svarene gruppa fikk. Dette ga derimot gruppa god tid til å samle inn informasjon om programvaren og systemet som skulle bli brukt. Ikke alle på gruppa hadde erfaring med VR eller utvikling i Unity, og dette ble en ypperlig mulighet for å utforske og se på kurs angående programvaren.

5.2.3. Gruppearbeid

Det var et bra samarbeid i gruppa. Det oppstod ingen alvorlige konflikter underveis i prosjektperioden, og har dermed ikke vært nødvendig å ta i bruk noen av tiltakene beskrevet i arbeidskontrakten. Dersom det var noen uenigheter mellom gruppemedlemmer, har dette blitt løst med små diskusjoner som ble gjennomført problemfritt.

5.3. Diskusjon av metode

5.3.1. Utviklingsprosess

5.3.1.1. Smidig utvikling med Kanban

Gruppa forsøkte å ta i bruk en smidig metodikk i løpet av utviklingsprosessen. Ettersom programmet ikke skal være bundet av strenge krav og ha muligheten til å kunne endre seg etter brukerens tilbakemeldinger til enhver tid, er det viktig å kommunisere kontinuerlig med oppdragsgiver. I løpet av utviklingen ble dette derfor gruppas største utfordring, da oppdragsgiver og brukeren av programmet hadde en meget fullpakket timeplan. Gruppa hadde ikke mulighet til å møte oppdragsgiver ofte nok til at de kunne ha en stor innvirkning på produktet. Videre ble det satt få krav til programmet og siden oppdragsgivere ikke hadde et klart bilde av sluttresultatet, var de fornøyd med det meste gruppa bestemte seg for. Dette medførte at det var gruppa som måtte ta de fleste beslutningene og var påtvunget en redusert kommunikasjonsflyt med brukerne.

5.3.1.2. GitLab

En annen utfordring som reduserte gruppas effektivitet var bruken av GitLab. Underveis i prosessen fant gruppa ut at GitLab var lite kompatibelt med Unity. GitLab ble brukt i utgangspunktet slik at gruppemedlemmene kunne jobbe og legge til funksjoner parallelt med hverandre, men ettersom prosjektet hadde få scener jobbet medlemmene ofte med samme filer. Dette førte til konflikter hver gang noen skulle laste den nyeste versjonen av koden som bare kunne bli løst ved å gå tilbake til den forrige versjonen. I tillegg hadde gruppa kun ett sett med utstyr, slik at kun én person hadde muligheten til å teste koden sin om gangen. Dette påvirket gruppas effektivitet i stor grad, da de nå måtte utvikle en versjon av koden på sin egen PC, for å så bytte til gruppas "hoved-PC" også skrive koden på nytt, men tilpasset til VR, for å så teste og sjekke at det de hadde skrevet faktisk fungerte. Til videre arbeid, burde det ha blitt lagt mer innsats i å lete etter en løsning og kanskje utforsket alternativer som Plastic SCM som er Unity sin egen versjonskontroll.

5.3.2. Visualisering

Python er det vanligste og et sterkt anbefalt programmeringsspråk for datavisualisering og analyse. Dette virket derfor som det mest naturlige valget og gruppa forsøkte å bruke dette programmeringsspråket til å begynne med. I løpet av testen blir all informasjon om hvor en pasient ser lagret som punkter. Videre ble det skrevet en kode som omgjør punktlisten til et varmekart slik at det er mulig å se hvor det var hyppigst øyebruk i løpet av testen. Problemet var at siden programmet er skrevet i Unity og koden er skrevet i Python, måtte brukeren kjøre testen ferdig, for å så åpne Python-skriptet og kjøre denne

koden for å få resultatene. Siden programmet er ment for personer med lite teknologisk kompetanse, prøvde gruppa å automatisere så mye av programmet som mulig.

Det ble først gjort flere forsøk på å kjøre Python-koden fra Unity. Gruppa prøvde først å bruke "Python for Unity", en programvare utviklet av Unity for å kunne kjøre Python-kode i Unity. Det ble oppgitt i dokumentasjonen at Python for Unity krevde Python 2.7.5 eller tidligere og den seneste versjonen av Unity 2019. Gruppa hadde brukt Python 3 og prosjektet brukte en Unity 2021 versjon. Videre ble derfor både Python-koden omgjort til Python 2.7.5 og prosjektet ble konvertert til en eldre Unity versjon. Kjøringen av Python-koden ble etter dette automatisert, slik at dataen ble plottet etter hver test, men Unity krasjet hver gang Python-koden ble kjørt. Grunnen til dette var fordi programgrensesnittet ikke støttet numpy, et bibliotek som var essensielt for plotting av grafer i Python. Gruppa var derfor nødt til å finne nye løsninger for hvordan man kunne visualisere data.

Ettersom Python ikke lenger var en mulighet, ble det undersøkt om C# var et alternativ, siden dette var det programmeringsspråket som allerede var brukt for å skrive resten av koden. Gruppa oppdaget et bibliotek, med navn Oxyplot, som kunne visualisere enkle grafer. I motsetning til Python, gikk installasjonen av dette biblioteket problemfritt, men hadde likevel noen få begrensninger. Selv om Oxyplot fungerer tilstrekkelig for 2D-grafer, har det utfordringer med å representere 3D-data på en nøyaktig måte. Dette førte blant annet til visse begrensninger i henhold til representasjon og visualisering av dybde. Dersom det befant seg flere objekter bak eller foran hverandre i et eksperiment, ville det ikke være noen måter å visuelt skille disse fra hverandre i en todimensjonal graf. Man vil for eksempel i et heatmap ikke vite hvilket objekt det ble sett på, ettersom data fra det ene objektet vil overlappe med det andre objektet, grunnet en manglende dybdeakse. Dersom pasienten ser langs veggene til eksperiment-området, vil disse punktene kun legge seg langs kanten av spredningsplottet og heatmappet, og vil dermed ikke nøyaktig avbilde personens øyebevegelser. Disse problemene med visualisering av dybde er noe alle plottene har grunnet Oxyplot sine mangler på tredimensjonale visualiseringsteknikker.

6. Konklusjon og videre arbeid

6.1. Konklusjon

Oppgavens problemstilling var å utvikle et VR-miljø med øyesporing for eksperimentering, med utgangspunkt i å gi spesialister innen helsesektoren et mer fleksibelt miljø for evaluering av pasienter. Det er mulig å si at gruppa har klart å lykkes med å løse problemstillingen da gruppa har klart å lage et VR-miljø hvor det er mulig å både opprette og gjennomføre eksperimenter, samt hente ut og lagre relevante resultatdata for evaluering av slagpasienter med neglekt. Gruppa har klart å utvikle et VR-miljø med øyesporing som oppdragsgiver har uttrykt stor tilfredshet for med et ønske om å teste programmet på ekte slagpasienter. Ettersom oppgaven fokuserer på å skape miljøet, er ikke miljøets effektivitet av evaluering relevant for gruppa.

6.2. Videre arbeid

Selv om gruppa har klart å innfridd alle kravene som ble stilt av oppdragsgiver og implementert enda flere funksjoner som gjør programmet mer brukervennlig, er det likevel aktuelt å diskutere mulig videre arbeid som produktet kan ha. Ettersom dette kun er en prototype og som tar utgangspunkt i et mulig utfall av hjerneslag og kun en form for neglekt, er det mange ulike veier dette produktet kan videre ta.

6.2.1. Replay-funksjon

En ønskelig fremtidig implementasjon ville vært en form for replay-funksjon. Akkurat nå er det flere former for datavisualisering, men å kunne se selve testen igjen som et videoklipp ville åpnet mulighetene for flere og nye måter å analysere testresultatene på.

6.2.2. 3D visualisering

Grunnet tidspress og manglende dokumentasjon på nåværende bibliotek, er det nå kun mulig å visualisere test-data med 2D plot. På grunn av dette blir ikke all data representert på en nøyaktig måte. Det kan ved videre arbeid være aktuelt å lage et eget system i VR-miljøet som håndterer visualiseringen av data. Dette hadde gjort det lettere og mer praktisk for brukerne av systemet å hente frem og se resultater fra eksperimentene, siden man da ikke ville trengt å åpne separate filer for å se visualiseringen av data. Det kan også gjøre at visualiseringen av data vil være mer nøyaktig i forhold til hvordan det faktisk så ut under eksperimentet, siden det kan visualiseres i det samme miljøet.

6.2.3. Bevegelsesbane

Gruppa har implementert en enkel bevegelsesbane for eksperiment-objektene, hvor objektene beveger i en rett linje fra ett punkt til et annet. I fremtiden kan denne bevegelsesbanen gjøres mer komplisert ved å for eksempel la objektene kunne bevege seg mellom flere punkter. Det kan også implementeres en bevegelsesbane basert på for eksempel bezierkurver, som vil gjøre det mulig å gi objektene krumlinjet bevegelse.

6.2.4. Objekter av forskjellige form

Foreløpig er eksperiment-objektene begrenset til å kun være kuler. Oppdragsgiver har derimot uttrykket et ønske om å kunne ha eksperiment-objekter av flere ulike former og figurer. Ved videre arbeid kan det dermed vurderes å implementere et system som lar brukeren velge mellom flere ulike former og figurer når de skal plassere et objekt i eksperiment-området. Ved å bruke figurer av objekter som en person vanligvis ser i hverdagen sin, kan man muligens få en bedre forståelse av hvordan neglekt kan påvirke en pasients daglige liv og interaksjon med miljøet de befinner seg i.

Samfunnspåvirkning

Gruppas oppgave var å lage et verktøy som skulle være et hjelpemiddel i tillegg til de metodene spesialistene bruker for å evaluere pasienter i dag. Dette verktøyet vil ta i bruk et VR-miljø og det vil da være naturlig å sammenligne økonomiske og miljømessige påvirkninger mellom å ta i bruk VR i motsetning til tradisjonelle evalueringer.

Tradisjonelle tester tar vanligvis i bruk penn og papir. Selv om å printe papir virker mer økonomisk enn å investere i VR-utstyr, er det å kjøpe VR-utstyr kun en engangskostnad. I tillegg er utstyret fleksibelt og kan brukes i flere avdelinger som kan hjelpe med å redusere andre utgifter. Videre er det mange som er bekymret angående hvordan papir påvirker miljøet. Det er derimot flere motstridende kilder om hvorvidt produsering av papir har store utslipper, da mesteparten blir resirkulert. Det er derfor vanskelig å konkludere om tradisjonelle tester eller bruken av VR er det mest bærekraftige for miljøet.

Videre kan det være verdt å nevne helsemessige faktorer med bruken av VR. Å bruke VR kan føre til noe som ofte blir kalt VR-syke. Ettersom VR kan rote med sansesystemet, vil brukeren ofte oppleve symptomer som kvalme, svimmelhet og tap av balanse. Disse symptomene er derimot midlertidige og reversible, men symptomene kan oppstå etter bare noen få minutter av bruk. I tillegg bruker VR-brillene OLED skjermer med blått lys, noe som kan skape søvnproblemer når den brukes om kvelden før man legger seg til å sove. Det er dessuten ikke dokumentert noen langsigte effekter av VR-bruk [29].

Oppgavens problemstilling omhandler hovedsakelig spesialister innenfor det relevante emnet, men det er likevel en tredjepart som vil bli påvirket av oppgavens resultat. Det er derfor viktig å diskutere både spesialistens påvirkning, samt ta hensyn til hvordan pasienten blir påvirket. Det finnes mange utfall man kan oppleve etter hjerneslag og flere utfall kan oppstå samtidig. Noen av de vanligste utfallene er svakhet, klossethet, lammelse på den ene siden og kommunikasjonsproblemer som lese- og regneproblemer [30]. Dagens evalueringer av neglekt tar gjerne ikke hensyn til at pasienten har noen av disse utfallene i tillegg til neglekt, da det krever at pasienten er frisk nok til å kunne skrive og tegne på papir. Disse testene vil derfor ikke kunne differensiere mellom neglekt eller motoriske svekkelse [6]. Dette er noen av problemene dette produktet løser, da det gir spesialistene muligheten til å designe eksperimenter som tilpasser seg utfordringene. I motsetning til tradisjonelle tester som krever manuell respons til visuell sok, vil eksperimentene i VR kun ta i bruk pasientens øyne og visuelle signaler. Dette produktet vil dermed kunne bidra til inkludering og økt sosial bærekraft.

Referanser

- [1] «Hjerneslag og TIA», *NEL - Norsk Elektronisk Legehåndbok*, 23. januar 2023. <https://legehandboka.no/handboken/kliniske-kapitler/hjertekar/tilstander-og-sykdommer/hjerneslag-og-tia/hjerneslag-og-tia/> (åpnet 26. april 2023).
- [2] L. Thomassen, «hjerneslag», *Store medisinske leksikon*. 27. januar 2023. Åpnet: 25. april 2023. [Online]. Tilgjengelig på: <https://sml.snl.no/hjerneslag>
- [3] Helsedirektoratet, «Hva er hjerneslag?», *HelseNorge*, 11. oktober 2016. <https://www.helsenorge.no/sykdom/hjerneslag/hjerneslag-arsaker/> (åpnet 25. april 2023).
- [4] U. Sveen, «neglekt», *Store medisinske leksikon*. 16. oktober 2019. Åpnet: 25. april 2023. [Online]. Tilgjengelig på: <https://sml.snl.no/neglekt>
- [5] «Unilateral Neglect - Physiopedia». https://www.physio-pedia.com/Unilateral_Neglect (åpnet 25. april 2023).
- [6] P. Plummer, M. E. Morris, og J. Dunai, «Assessment of Unilateral Neglect», *Phys. Ther.*, bd. 83, nr. 8, s. 732–740, aug. 2003, doi: 10.1093/ptj/83.8.732.
- [7] Megan, «Left Neglect After Stroke - Definition & Treatment Exercises», *Tactus Therapy*, 7. august 2019. <https://tactustherapy.com/what-is-left-neglect/> (åpnet 19. mai 2023).
- [8] «Figure 1. Apple Cancellation sheet.», *ResearchGate*. https://www.researchgate.net/figure/Apple-Cancellation-sheet_fig1_51129114 (åpnet 19. mai 2023).
- [9] I. Tsirlin, E. Dupierrix, S. Chokron, S. Coquillart, og T. Ohlmann, «Uses of Virtual Reality for Diagnosis, Rehabilitation and Study of Unilateral Spatial Neglect: Review and Analysis», *Cyberpsychol. Behav.*, bd. 12, nr. 2, s. 175–181, apr. 2009, doi: 10.1089/cpb.2008.0208.
- [10] H. Dvergsdal og L. Aabakken, «virtuell virkelighet», *Store norske leksikon*. 5. mai 2023. Åpnet: 10. mai 2023. [Online]. Tilgjengelig på: https://snl.no/virtuell_virkelighet
- [11] J. Bardi, «Virtual Reality Defined & Use Cases», *3D Cloud by Marxent*, 26. mars 2019. <https://www.marxentlabs.com/what-is-virtual-reality/> (åpnet 10. mai 2023).
- [12] «What is eye tracking? | How does eye tracking work - Tobii». <https://www.tobii.com/learn-and-support/get-started/what-is-eye-tracking> (åpnet 10. mai 2023).
- [13] «Virtual reality (VR) | Definition, Development, Technology, Examples, & Facts | Britannica», 9. mai 2023. <https://www.britannica.com/technology/virtual-reality> (åpnet 10. mai 2023).
- [14] «VIVE Pro Eye Features | VIVE Southeast Asia». <https://www.vive.com/sea/product/vive-pro-eye/features/> (åpnet 19. mai 2023).
- [15] A. Holm, «spillmotor», *Store norske leksikon*. 26. januar 2023. Åpnet: 18. mai 2023. [Online]. Tilgjengelig på: <https://snl.no/spillmotor>
- [16] Atlassian, «What is Agile?», *Atlassian*. <https://www.atlassian.com/agile> (åpnet 25. april 2023).

- [17] «Manifestet for smidig programvareutvikling». <https://agilemanifesto.org/iso/no/manifesto.html> (åpnet 25. april 2023).
- [18] Atlassian, «What is version control | Atlassian Git Tutorial», *Atlassian*. <https://www.atlassian.com/git/tutorials/what-is-version-control> (åpnet 25. april 2023).
- [19] «What is GitLab and How to Use It? [2023 Edition] | Simplilearn», *Simplilearn.com*. <https://www.simplilearn.com/tutorials/git-tutorial/what-is-gitlab> (åpnet 19. mai 2023).
- [20] «VIVE Pro Eye Specs & User Guide - Developer Resources». <https://developer.vive.com/resources/hardware-guides/vive-pro-eye-specs-user-guide/> (åpnet 25. april 2023).
- [21] «Unity Game Engine Guide: How to Get Started with the Most Popular Game Engine Out There», *freeCodeCamp.org*, 13. februar 2020. <https://www.freecodecamp.org/news/unity-game-engine-guide-how-to-get-started-with-the-most-popular-game-engine-out-there/> (åpnet 5. mai 2023).
- [22] «OpenVR (Steamworks-dokumentasjon)». <https://partner.steamgames.com/doc/features/steamvr/openvr> (åpnet 25. april 2023).
- [23] thetuvix, «OpenXR - Mixed Reality», 1. februar 2023. <https://learn.microsoft.com/en-us/windows/mixed-reality/develop/native/openxr> (åpnet 25. april 2023).
- [24] «OpenXR - High-performance access to AR and VR —collectively known as XR—platforms and devices», *The Khronos Group*, 6. desember 2016. <https://www.khronos.org/openxr/> (åpnet 25. april 2023).
- [25] «WebAIM: Contrast Checker». <https://webaim.org/resources/contrastchecker/?fcolor=000000&bcolor=2CA0E9> (åpnet 19. mai 2023).
- [26] «AppData – Where to Find the AppData Folder in Windows 10», *freeCodeCamp.org*, 31. juli 2020. <https://www.freecodecamp.org/news/appdata-where-to-find-the-appdata-folder-in-windows-10/> (åpnet 18. mai 2023).
- [27] «JSON». <https://www.json.org/json-en.html> (åpnet 14. mai 2023).
- [28] «Hva er CSV-fil?» <https://www.visma.no/eaccounting/regnskapsordbok/c/csv-fil/> (åpnet 14. mai 2023).
- [29] «What are the risks of virtual reality and augmented reality, and what good practices does ANSES recommend?», *Ansés - Agence nationale de sécurité sanitaire de l'alimentation, de l'environnement et du travail*, 24. juni 2021. <https://www.anses.fr/en/content/what-are-risks-virtual-reality-and-augmented-reality-and-what-good-practices-does-anses> (åpnet 18. mai 2023).
- [30] «Utfall etter hjerneslag», *Norsk forening for slagrammede*. <https://slagrammede.org/utfall/> (åpnet 10. mai 2023).

Vedlegg

Vedlegg 1: Forprosjektplan

Vedlegg 2: Prosjekthåndbok

Vedlegg 3: Visjonsdokument

Vedlegg 4: Kravdokumentasjon

Vedlegg 5: Systemdokumentasjon

Vedlegg 6: Kildekode