Ahmadi, Fereshta

Midthus, Hjelle Steinar

Skorpen, Sindre

Synnes, Brevik Malin

System for utilizing eye tracking in a virtual maritime environment

Bachelor´s thesis in Engineering in Computer Science

Supervisor: Di Wu

May 2023

Abstract

This thesis explores the opportunities of combining eye tracking and virtual reality applications for the Norwegian Coastal Administration (NCA). After creating a virtual reality vessel simulator in collaboration with Morild Interaktiv AS, the NCA wanted to see which opportunities existed for including eye tracking in this simulator.

Preliminary research from an industrial project conducted during the previous semester showed potential of eye tracking for aiding in the productivity of users. A group of people showing interest in this industrial project was assembled to create an implementation of eye tracking in virtual reality as a bachelor's project.

When certain bridge staff are working in a vessel, there are specific areas they are expected to keep an eye on. As an example, some are required to look out of the window as much as possible to not rely too often on instruments.

This is data that can be recorded in virtual reality to provide feedback on the performance of a crew member in a visual way. A virtual reality demo was developed to investigate how this data can be recorded and presented to the user. The demo contains visual models that shows viewing patterns of the user in various ways. Additionally, a website was created to present data as graphs. In this website, the viewing patterns of the user can be compared other users.

The final project shows how eye tracking could be used in a virtual maritime training simulation to provide feedback and comparisons between users. This can be valuable, especially if a large amount of data is collected for extensive comparison.

Sammendrag

Denne avhandlingen undersøker mulighetene til kombinasjonen mellom øyesporing i virtuell virkelighet for Kystverket. Etter å ha utviklet en skipssimulator i samarbeid med Morild Interaktiv AS ønsket Kystverket å se på muligheter for å inkludere øyesporing i denne simulatoren.

Forarbeid har blitt utført gjennom en industrirapport fra et tidligere semester. Denne rapporten viste potensialet til øyesporing som et verktøy som kan øke produktiviteten til en bruker. En rekke personer interessert i industriprosjektet ble samlet for å utvikle en løsning som et bachelorprosjekt.

Når en los er på en båt, skal de helst se mest mulig ut av vinduet for å unngå ulykker og ikke bruke instrumentene for mye. Dermed om disse dataene kan tas opp i skipssimulatoren til Morild Interaktiv kan man bli presentert hvordan oppførselen er over tid på en visuell måte.

Løsningen som ble laget i dette prosjektet ser på hvordan det kan gis objektiv tilbakemelding på brukerens adferd i en virtual reality demo applikasjon. Her kan brukerens synsdata visualiseres gjennom punktskyer og kart som viser hvor brukeren har sett over tid. I tillegg ble en nettside utviklet for å se tilbakemeldingene i grafformat. Her kan de sammenligne sin egen prestasjon mot andres prestasjoner. Dette kan ha stor verdi i treningssimulatorer der det kan leveres raske objektive tilbakemeldinger, og utføre sammenlikninger mellom store mengder brukere.

Preface

About

Thea idea of gathering eye tracking data and visualizing seemed like an interesting project. The group worked in pairs, where they focused on creating a demo where they gathered the data and made a website to show this data. The group worked in the agile method Scrum, where they planned, created the two parts, and ensured effective communication and progress.

The group wishes to thank the following:

Our supervisor, Di Wu. Thanks for letting us have frequent and comprehensive meetings during the drafting of this thesis and being especially available near the deadline.

Our client at the Norwegian Coastal Agency, Odd Sveinung Hareide, who maintained a clear vision of how he wanted the project to progress.

Arne Styve, responsible for the IIR Visualization lab, who provided an additional VR headset, ensuring more efficient development.

OpenBridge team, especially Kjetil Nordby that did a design review for the webpage.

NTNU Ocean Ålesund, for lending us a laptop to speed up development at campus.

Assignment text

The definition of the assignment can be found in appendix A.

Table of contents

Figures

Coding examples

1. Introduction

1.1 Background

The Norwegian Coastal Administration is currently investigating virtual reality maritime training. They are currently utilizing a VR simulator from Morild Interaktiv. One investigation into the state of the art of eye tracking was conducted in an industrial project the fall of 2022 by Steinar Hjelle Midthus and Trine Staverløkk, of which Midthus is participating in this project. This investigation found opportunities of eye tracking usage within VR training simulations.

A bachelor's project was proposed following the industrial project – create an implementation of a feedback system gathering data from eye tracking, either integrating with the VR application Morild Interaktiv already created, or developing an entirely new demo project. The industry project can be found as appendix B.

1.2 Problem

The problem definition of the bachelor thesis is to look at how eye tracking can be used to give objective feedback to a trainee and compare them to an expert's performance.

In the project, the task is to track the eye movement of the captain of the ship. Look at how eye tracking data in a simulation can be used to give objective feedback to the user. Based on the data that is collected, there shall be an analysis carried out that can tell something about possible improvements. It is also a desire that a user should be able to be monitored by a third party who can give feedback during a session. Hence, they can say where they need to improve with the use of their own computer. The wanted solution should gather data about what percentage of the user is looking at different objects and display it on a graphical user interface (GUI). The GUI should also have the added functionality of comparing the trainee's performance against an expert. The data that was gathered should also be stored for further analysis.

1.3 Requirements

The requirements were taken from the assignment text in appendix A and defined throughout the project as wishes from both the supervisor and the project owner.

1.3.1 VR Demo

Using Unity engine

One of the goals of the project was an ability to integrate eye tracking solutions into the simulator software developed by Morild Interaktiv. Their software is based on the Unity engine, which required this project to be developed with Unity as well. During further discussion, integration into Morild Interaktiv's application stopped being a requirement, in favour of creating an open project they could base their solution on.

VR headset compatibility

Another requirement was to develop and ensure compatibility with modern VR headsets, with or without built-in eye tracking. The options here were the HP Reverb G2 Omnicept Edition, the Meta Quest Pro, and the Pico 4 Pro Enterprise. Of the three, HP and Meta's HMDs were available through the university and NCA. The group initially targeted both HMDs, but later moved on to just having the Meta Quest Pro as the target hardware.

Eye tracking requirements

One requirement was gathering data on an object category basis. Objects should be categorized as for example windows or props and stats for these categories should be saved.

It should be possible to see some visualizations of eye tracking data in the demo. This can be of several different types of data or visualizations if it is related to eye tracking.

Demo content

The demo should be able to be "played" from start to finish. Have a situation that can be completed by the user with a clear start and end. This can provide a clear way to compare data.

The users should also be able to do several types of interactions, like picking up objects and interacting with menus in the world. This way making the gathering of data less boring for the end user.

Unity Editor requirements

The logic created in this demo should be easy to set up for developers in a new scene. This is intended to make it easier for developers utilizing our project as a basis for modified scenes or new eye tracking applications entirely.

1.3.2 Backend

The backend should be able to handle persistent data, this data should be accessible via endpoints. The backend should also be able to handle different users and make it possible to compare data from different users.

1.3.3 Frontend

The frontend should be able to display different sessions from the backend and compare these. It should also show visualizations from the gathered data. The website should be fully functional both on mobile and desktop devices. The website should follow an established maritime UX standard, called the Open Bridge standard.

1.4 Limitations

Working with the Norwegian Coastal Administration and Morild Interaktiv came with some limitations. The frontend was requested to comply with the OpenBridge System standard, due to NCA having familiarity with that standard. The VR demo was required to be developed in Unity, so that Morild Interaktiv could utilize our findings more easily in their internal Unity project. The demo was not required to be in a naval setting, but with NCA as the main client, the group decided to set the demo in a ship bridge setting. The demo also needed to run properly on target hardware, that being laptops with dedicated graphics at least as or more powerful than an Nvidia RTX 3060.

1.5 Structure

The structure of the report:

Chapter 1: Introduction

Introduction of the project with an explanation of the background, the problem, requirements, and limitations.

Chapter 2: Theory

Presents the theoretical background of the solution and the development. The foundation of the bachelor thesis.

Chapter 3: Method and Material

Description of the methods utilized and material that is used out of this project.

Chapter 4: Results

Presenting the results of the completed project.

Chapter 5: Discussion

Discussion and reflection of the achieved results, results related to the problem thesis and further development.

Chapter 6: Conclusion

Concludes the bachelor thesis as a thematic and subsequent discussion.

1.6 Acronyms and Glossary

1.6.1 Acronyms

NTNU – Norwegian University of Science and Technology

IIR – Department of ICT and Natural Sciences

IHB – Department of Ocean Operations and Civil Engineering

HMD – Head Mounted Display

VR – Virtual Reality

XR – Extended Reality

CSS – Cascading Style Sheet

HTML – Hypertext Markup Language

OOP – Object Oriented Programming

ORM – Object Relational Mapping

ECS – Entity Component System

SDK – Software Development Kits

GUI – Graphical User Interface

UI – User Interface

UX – User Experience

IDE – Integrated Development Environment

PBR - Physically Based Rendering

REST – Representational State Transfer

HDRI – High Dynamic Range Image

POC – Proof of concept

1.6.2 Glossary

Backend – The server side of the website. Stores and treats data.

Frontend – The visual side of the website. Connects the user to the backend.

VR demo – A demo VR application created to showcase eye-tracking in VR, through the game engine Unity. Feeds data to the backend.

Screen Space – A 2D coordinate system based on the display an application is running on. Often used in game development for GUI. If a third dimension is introduced to the coordinate system, the Z axis represents the inward/outward of the screen.

World Space – A 3D coordinate system used for 3D scenes. Used for scenes in Unity, and especially relevant for GUI in VR.

2. Theory

This chapter will be about the theory of the project. The project management method, some common eye-tracking terms and technology will be stated here.

2.1 Software development

2.1.1 Agile development

Agile development is a project management and software development methodology that emphasizes iteration and continuous delivery of value to a user (Atlassian n.d.). Instead of delivering a complete product all at once, agile teams usually break down their work into smaller, more manageable increments. This approach allows for ongoing evaluation and adjustment of requirements, plans and outcomes. This enables teams to respond more quickly to changes in circumstances, and with less complexity than traditional project management methods.

Sprints

In agile and scrum methodologies, a sprint is a crucial time-bound period in which a scrum team focuses on completing a defined amount of work (Atlassian n.d.). By breaking down larger projects into sprints, teams can ensure steady progress towards the end goals while also maintaining flexibility and adaptivity. Sprint management is essential for successfully delivering high-quality software products in a timely and efficient manner. This will minimize complications and simplify the development process.

Sprint reviews

The sprint reviews assist the developers to gather feedback from the stakeholders (Zoho n.d.). This meeting is held at the end of each sprint. The developers will talk about the process in the project.

User stories

A user story can be described as a non-formal and broad software feature, presented from the viewpoint of an end-user. The primary objective of a user story is to express how a software feature will benefit the user and provide value to them.

Meetings

In Scrum, meetings promote transparency and facilitate regular communications within the team (Radigan n.d.). This will help to plan, discuss, and gather feedback on the work. Sprint planning and daily stand-ups are a part of scrum meetings.

Roles

According to the scrum model, there are three roles: product owner, scrum master, and development team members (West n.d.). The product owner sets clear direction since they understand the customer and business needs. The scrum master is the one holding it all together and should ensure that scrum is being done well. This implies that they aid the product owner in determining the value, enable the delivery of such value by the development team, and help the scrum team to continuously improve themselves in practical terms. The development team members consist of people that do the work, such as engineers, designers etc.

Daily stand-ups

Daily stand-ups, also called daily scrum, are short meetings daily to discuss the progress and identify potential blockers (Radigan n.d.). The purpose is to provide a brief and efficient update to the entire team regarding ongoing activities. Unlike a status meeting, it aims to quickly inform everyone in a light-hearted and enjoyable manner. The atmosphere should be informative and enjoyable. The team should be standing up to keep the meeting short optimally no more than 15 minutes.

Scrum

Scrum is a framework developed to help with complex product development. Scrum is one of the leading frameworks for software developers but can be useful in other contexts (Ramsøy 2022). It is based on empiricism which means that knowledge comes from experience and decisions are based on what is known. Scrums have three fundamental roles. The first one is a product owner, the second is the development team and at last the scrum master.

2.1.2 Cohesion and coupling

High cohesion

High cohesion is the wanted type of cohesion (Josikakar n.d.). Cohesion is how well element in a module work together to their purpose. If they are closely related or if they are loosely related. High cohesion is that they are closely related and only serve one purpose. It makes it easier to make changes to the code since one function has one task. The function is consistently, and the reliability of the system is improved.

Loose coupling

Loose coupling has a lot of advantages. For instance, it improves maintainability and scalability (Josikakar n.d.). It reduces the impact of changes in one module to the other modules. It facilitates adding new modules and removing the existing ones. Loose coupling gives enhanced modularity that allows modules to be developed and tested in isolation. These are the reasons for wanting loose coupling. Coupling refers to the interdependence between the modules.

2.1.3 Functional programming and object-relational programming

JSON Web token

JSON Web Token (JWT) is an authentication method that uses a token to identify the user and their credentials (Okta n.d.). Authentication based on tokens uses the token in each request to the server to authenticate the user.

Object-oriented programming paradigm

The object-oriented programming paradigm is a programming paradigm that has objects as their focus (MDN contributors 2023). These objects are represented as "blueprints" that are called classes that says what fields and methods the object has. The fields represent any data like text or numbers, and methods define the behaviour of the object. When an object is made with its defined values through a constructor that is called an instance. These values are held by the objects fields and belong only to this object.

Functional programming

Functional programming is a paradigm that uses functions as the main method of programming (Hargrave 2023). Each function represents a way of solving a simple problem and is then called from multiple sources to do the same actions multiple places.

Object-Relational mapping

When it comes to the Object-relational mapping (ORM) is a mapping method for databases that combines the advantages of object mapping and relational mapping. (Chehab 2023). The advantages of using an ORM is that the objects themselves and their relations are stored in the database. This way making it more like the code that the data is built on. One major disadvantage is the complexity it introduces the system.

2.1.4 Quality assurance

User testing

User testing is the testing process where you test your product on real users. The users get a task preform in realistic conditions (OmniConvert 2023). The purpose with this testing is to evaluate the usability of the product. For getting results that is relevant the user must not be directed too much, because the product should be intuitive. The user should feel comfortable with using the product even though they are not familiar with it. There are different types of user tests. There is the usability testing, A/B testing, focus group and beta testing. This process will eventually tell the developer if their product is ready to be launched for real users.

Unit tests

Unit tests help the developers to validate that each unit of the software works as intended and meets the needs (pp_pankaj 2023). These tests are automatically run each time the code is executed and will help the developer determine if the issue is made by the developers.

Code review

Code reviews assist the developers to identify bugs, sharing the developer's knowledge, increasing collaboration, increasing their code quality, and helping the developers learn the source code (GitLab n.d.). This is important because a strong code review process will set the foundation for continuous improvement and prevents unstable code from shipping to customers.

2.1.5 Distributed version control

Distributed version control system or DVCS makes it possible for each member of a project to have their own copy of the complete repository (GitLab n.d.). The team members can commit and merge locally. An example of a system that uses DVCS is Git. DVCS help developers that work in team to create a robust workflow. It gives a flexibility with for example working home or offline. When a developer pushes changes to their own copy others can set a code review process to ensure that the quality to the code is decent.

2.1.6 Security

Authentication and authorization

Authentication is the process of checking if the person is who they say they are to get access to methods or data (Okta n.d.). It is important to not give the wrong data to the wrong people. Authorization is the process that checks if you have access to get the requested information.

GDPR

The General Data Protection (GDPR) does explicitly state the specific requirement of user authentication for accessing personal data, it does emphasize the need for appropriate security measures to protect personal data (Intersoft consulting n.d.). Article 32 of the GDPR states that organizations are required to implement appropriate technical and organizational measures to ensure a level of security appropriate to the risks presented by the processing of personal data. This includes measures such as ongoing confidentiality, integration, availability, and robustness of processing systems, and regularly testing and evaluating the effectiveness of security measures.

2.1.6 Design process

Interaction design

Interaction design alias IxD, is an essential part in user experience design (Interaction Design Foundation n.d.). It means the design of interactive product and services where the designer has a high focus on the item in development to include how a user will interact with it. Interaction design is the interaction between a user and the product. IxD works in five dimensions, the fist is words. The second is Visual representation. The last ones are physical objects and space, time, and behaviour.

OpenBridge Design System

The OpenBridge Design System serves as the primary design during the application design process. This system compromises a set of rules and principles that were implemented throughout the development phase.

- **Application:** The OpenBridge standard has a strict main structure with a fixed top bar on top and a set of mandatory and optional functions. It should take up a full-screen space and scale responsively to a range of screen sizes (OpenBridge n.d.).
- **Top bar:** The top bar must be on each page and show where the user is currently on the webpage. This top bar should include a navigation menu, an application label to the left and an account button to the right (OpenBridge n.d.).
- **Cards:** Maritime user interfaces have optional cards embedded in a user interface. These types of cards are reusable and can be shared across many applications. The system includes variations depending on its content. Each card has an optional minimize function that allows users to collapse the card if it is possible (OpenBridge n.d.).

- **Help & Support:** The support page must have a search bar and the buttons must be clear and rich. It should also have a close button so that the user can go back to the last application view (OpenBridge n.d.).

Palette

- **Typography:** The font that should be used is Open Sans. The text styles should be used for normal text content, static text, and components. The instrument text should be used for dynamic data. The instrument text should be in a tabulator format (OpenBridge n.d.).
- **Colours:** The colour should be organized into types that reflect the function each colour has in an application. The colours are predefined palette sets for bright, day, dusk, and night (OpenBridge n.d.).
- **Styles:** will visually format an interactive element. The state of an element will change behaviour once interaction has occurred and the style change (OpenBridge n.d.).

Don Normans principles

Don Norman's 7 principles are also considered while developing the application (educative n.d.). The seven principles are the following:

- **Visibility:** Things must be visible so that users can access them.
- **Feedback:** Give feedback to the user so that the user knows whether their actions were successful or not.
- **Affordance:** The link between how something looks and how they are used.
- **Mapping:** The design must resemble what they are doing.
- **Constraints:** Restrictions for users so that the user does not become overwhelmed.
- **Consistency:** Patterns need to be recognized and learned by users, so using the same patterns will not frustrate the user.

2.2 Design patterns

Design patterns are ways to solve a general problem that comes often to developers when applications are made (Source Making n.d.). These patterns are often so general in nature that the implementation and how it should be done can be decided by the user.

2.2.1 Observer pattern

The observer pattern is a commonly used design pattern that solves the problem of constantly checking when a state changes (Freeman, Robson et al. 2021). This pattern works by having the objects that wants to "listen" to a state subscribe to the object.

When the objects state changes it goes through and alerts the observers about the change. This way reducing the need of constantly checking the state of the object.

2.2.2 Façade pattern

"Facade patterns take a complex subsystem and make it easier to use by implementing facade class that provides one, more reasonable interface." (Freeman, Robson et al. 2021) The pattern uses a higher level of simple interfaces, that then simplify the subsystem. The important thing about the pattern is that it is intent. Façade makes one interface that assembles the subclasses into one. That is the reason that makes it so much easier to use for the user. The user interacts with the interface instead of all the subclasses.

2.2.3 Entity Component System

The ECS pattern is a pattern commonly used in game development (Unity n.d.). "Entities" are object references without data, and "components" are pieces of data that can be connected to entities. "System" is the system that acts upon the components. An example of this could be a system of "time", which iterates on a "move" component to move a "car" entity.

2.2.4 Singleton pattern

The singleton pattern is commonly used pattern when a class only needs one instance of that class to be accessible for the rest of the code (Freeman, Robson et al. 2021). For that class to only make one of itself the code must ensure that the constructor cannot create a new instance and the class itself can control and send the instance to other classes. This is often done through static get method for the class and a private constructor.

2.2.5 Builder pattern

Builder pattern is used for preventing that the steps of creating the object, get mixed. Some steps come before other steps (Refactoring Guru n.d.). Too solve that we encapsulate the way a complex object is constructed, and then allows the object to be constructed in multistep. This pattern is often used for building composite structures (Refactoring Guru n.d.). This design pattern makes it possible to decide what the object will have of possibilities. You have one class that builds the object. Subclasses under is possibilities for building the object and this pattern build it in the right steps.

2.3 Virtual reality

2.3.1 XR

XR stands for extended reality. XR is combination of both VR and AR. Virtual reality and argumented reality (Midthus and Staverløkk 2022). AR is overlays digital content on top of the real world when VR is a virtual environment to create realistic situations. That means that XR is uses both. You can also say that XR uses mixed reality that uses the best of VR and AR for capture the real world (Qualcomm 2022).

2.3.2 3D graphics

Large parts of the 3D graphics are handled in the game engine, but theoretical knowledge of computer graphics is still needed to utilize the game engine effectively.

PBR Materials

Physically based rendering (PBR) materials change the rendering of surfaces to represent different physical properties. This is achieved through various bitmaps applied to the material in steps.

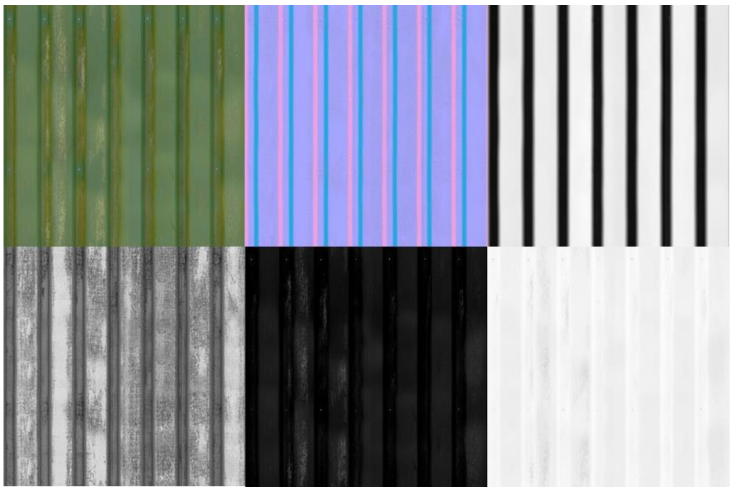


Figure 1 A painted green corrugated metal wall PBR material consists of multiple maps. From top left to bottom right: Diffuse, normal, displacement, ambient occlusion, specular, roughness.

The map that provides the most significant difference is the diffuse map (see figure 1, top left), alternatively referred to as the colour map. This image decides how the surface will be coloured. In some cases, this can be enough to provide the detail a material needs (Marschner and Shirley 2015).

A normal map (figure 1, top middle) changes the direction of lighting on a given point, providing significant details. The direction is decided by the RGB colour in the position, where the RGB values are interpreted as a 3D vector (Marschner and Shirley 2015).

Displacement map (figure 1 top right) changes the height of a given point, based on a black and white value which corresponds to 0 and 1 (Marschner and Shirley 2015).

Ambient occlusion (figure 1 bottom left) is used to determine how hard or soft shadows should be on a given point of a texture (Marschner and Shirley 2015).

Specular (figure 1, bottom middle) determines how much light is reflected off a given point of a surface (Marschner and Shirley 2015).

Roughness (figure 1, bottom right) represents how rough or shiny a material is on a scale from 0 to 1, where 0 is completely shiny, and 1 is a rough surface scattering light. This is interpreted through a black and white bitmap (Marschner and Shirley 2015).

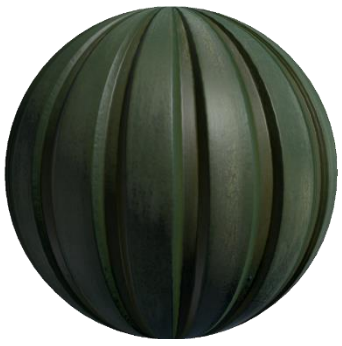


Figure 2 All maps combined and applied to a sphere in the Unity game engine.

HDRI

An HDRI is a panoramic image that provides reflection values and lighting. These can function both as the sky in the scene, and baked reflections.

Figure 3 A panoramic HDRI visualized as a sphere in the Unity game engine.

2.3.3 Game engine

A game engine is a software framework containing various relevant tools for game development. Rendering technology for realistic 3D graphics, scripting language interfaces, input handling, and UI tools are examples of common tools found in game engines.

2.4 Eye tracking in VR

2.4.1 Eye Tracking terms

Fixation

Fixation is the number of times that the trainee looks at a spot or area for a small period (Midthus and Staverløkk 2022). It is often defined as 50-200ms to register as a fixation.

Fixation duration

Fixation duration is the amount of time the fixation is held at a certain point in space (Midthus and Staverløkk 2022).

Average fixation duration

Average fixation duration is found by taking the fixation duration and dividing it by the number of fixations (Midthus and Staverløkk 2022). This way getting the average time spent per fixation.

Area of interest

An area of interest (AOI) is an area in space that is interesting to track get data from in a session (Midthus and Staverløkk 2022). Instead of focusing on each point the user looks at the AOI represents a general area to simplify the visualization of eye tracking data. It has the same properties to track as fixations, fixation duration and average fixation as a normal observation.

Ray casting

In 3D space, ray casting is the process of sending a 3D vector through space form an origin point, and recording what it hits. This can be used for several applications, such as hit registration for pointers, point registration in eye-tracking, path tracing, or UI interactions.

Gaze plot

Gaze plot is a representation of the location, time, and order that a person used to look at an item (Midthus and Staverløkk 2022). If we have a point of interest, the number on that point will show the order of where the person looked. That means that the points will be marked with 1, 2 ,3 for the order. Then we have the size of the point of interest that indicates the time or duration the person used looking at that point. The greater the point is, the more time the person used to look at that point. The gaze plots are sometimes represented in the virtual word by using the points directly on the objects.

Heatmap

Heatmap is a general statistical model. The purpose of this model is to display where the user has used most time (Midthus and Staverløkk 2022). The model uses red and blue colours where the blue is location that is viewed less, and the red are viewed more. The colour gets brighter the more fixations time is used. This indicates that the heatmap is used to see what the user is interested in. Although the heatmap can indicate confusion with the time regions that are viewed for a longer time.

Pupillometry and cognitive load

Pupillometry is defined as the study of the correlation between cognitive load and changes in pupil size (Sirois and Brisson 2014). When the user experiences a higher cognitive load or increased attention the pupil size gets bigger. Pupil size is also a good indicator of fatigue since the size then decreases and the stable behaviour of the pupil gets more unpredictable.

3. Method and Material

This chapter will show the working processes and tools this project.

3.1 Project Management

3.1.1 Team

The team is made up of four students at NTNU Ålesund. All 4 students are in their final year of a bachelor's degree in computer science.

3.1.2 Task Distribution

Steinar Hjelle Midthus was assigned to a team leader role, and divided tasks among other group members. Malin Synnes and Fereshta Ahmadi were responsible for development of the front- and backend for data visualization. Sindre Skorpen and Midthus had a shared responsibility for developing the VR prototype. Midthus handled networking and eye-tracking logic, as well as shader graphics. Skorpen developed VR interactions such as teleporting and picking up items, in addition to scene building and model creation.

3.1.3 Client

The client in this project is the Norwegian Coastal Administration. Odd Sveinung Hareide is the representative of the administration and the project owner.

3.1.4 Advisor

The advisor in this project is Di Wu. Wu is an associate professor and lecturer at NTNU. Her role in this project is to advise the group and be the scrum master. Meaning that she assures that Scrum is being followed within the group.

3.1.5 Meetings with the client

The group had meetings with the client every week. During these meetings, the status of the project was given to the client with a clear and specific agenda. This gave the client opportunities to ask questions and give feedback to the group. The client received demos of how the project worked as well after the meetings were done.

3.1.6 Project structure

The project's structure was clear from the beginning since the project owner defined a good goal for this project. That's when the group decided to separate the project into three parts. A frontend, backend, and simulation demo.

The simulation demo was responsible for gathering the data that was needed from a trainee and sending this to the backend. Since the eye-tracking data gathered could be visualized it would also be possible to look at how the data could be represented in Unity. The project owner also wanted to look at the possibility of having an observer that could tell the trainee what they should look more at in a live simulated setup.

The backend was going to be used to store the data gathered in the simulation demo and handle requests from other applications using the REST principles.

The GUI that would show the data was decided to be on a webpage since it can be developed for both a desktop and mobile. This way it will ensure that the data can be looked at across any media. A webpage would also be possible later to be exported to a mobile app if the right framework was used.

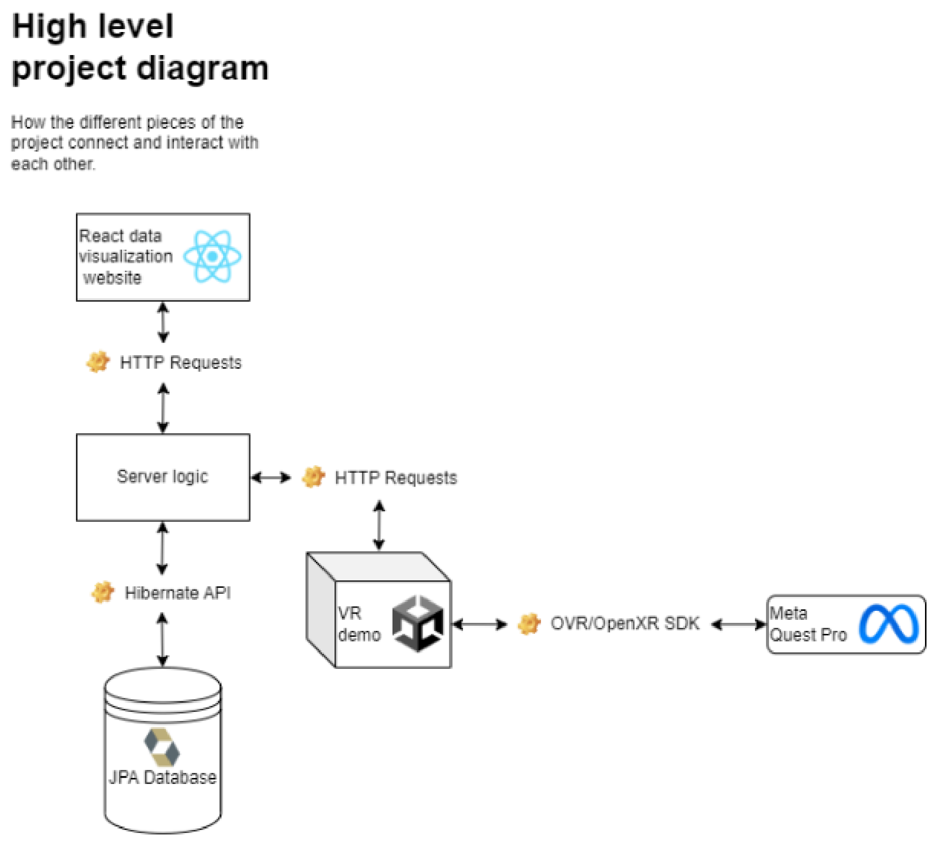

High level project diagram
How the different pieces of the project connect and interact with each other.
React data visualization website
HTTP Requests
Server logic
HTTP Requests
Hibernate API
VR demo
OVR/OpenXR SDK
Meta Quest Pro
JPA Database

Figure 4 High level project diagram

3.1.7 Scrum

Roles and work distribution

The group was divided into two separate pairs. One pair was focused on creating the demo in Unity, and the other one was focused on making visualization. Additionally, the group leader kept an overview of the project progress, and participated in frontend, backend, and demo development.

Sprints

The project was divided into sprints. A sprint helped the group split bigger tasks into smaller sections, so it wasn't that overwhelming to understand. This made it easier to focus on a specific part of a task. Each sprint lasted a week. After each sprint, the group had a sprint review and wrote a retrospective. These were used to summarize what was good in the sprint and what we could have done better. In the beginning, we started a new sprint every Friday, but afterwards, it was every Monday.

User stories

User stories in agile methods were used to describe the requirement specification. In agile methods, the requirements and tests are the same thing. The developers made user stories and acceptance criteria during the development process. This set of different types of criteria needed to be fulfilled when finishing a task.

These types of stories were made in Jira, and tasks were made to accomplish the requirements. The group had an overview of what the main goal was, and how to reach it by taking one step at a time. This ensured that the person taking the task would not be overwhelmed as well.

3.1.8 Communication

Messenger

Most of the informal group communication was conducted through the chat application Messenger. Daily problem discussion and communication regarding planning such as attendance, project, and documentation decisions, as well as task allocation often happened in the Messenger group chat before it was formalized elsewhere.

Discord

Some of the communication happened on Discord as well. Sometimes the sprint reviews were conducted in Discord voice chats. Discord allows users to join voice chat rooms at any time, showing that the user is available. This allows for more spontaneity than for example Teams, where a meeting needs to be set up beforehand. Working remotely, team members could join the Discord voice chat, ensuring a lower threshold for communication.

Microsoft Teams

More formal communication with stakeholders was conducted through Microsoft Teams. One Teams group was made for sharing documents and holding meetings with the stakeholders, and another was made to post progress videos to external parties.

3.1.9 Collaboration tools

During the project, different types of tools were used to collaborate. All documentation, version control, time logs, code, sprints etc. got shared so that the whole group always had access to it.

GitHub

GitHub is the chosen coding platform for version control and collaboration (GitHub Docs n.d.). This platform lets people work on a project together, regardless of where they are. Three different repositories were created. One for the frontend, one for the backend and one for the VR demo. Due to the size of the binary files in the VR demo, a subscription for data packs were used on that repository to ensure sufficient bandwidth. All three repositories were made in one organization belonging to the group developers.

Jira

The chosen software application development tool for tracking issues, managing projects, and automating workflow is Jira. This tracking tool gave the group an overview of each task and the working status (ProductPlan n.d.). The issue board made it clear what each group member was doing and what needed to be done. This made it easier to avoid several group members working on the same task at the same time.

Confluence

Confluence made it easier to create, collaborate and organize all the work documentation in a single place (Atlassian n.d.). Meeting notes, sprint retrospectives, agreements, decision logs, agendas for meetings and more is saved on the Confluence page. An additional advantage of Confluence is its connectivity to Jira. This connectivity allows sprint retrospective notes to be automatically created once a sprint in Jira is ended for example.

Microsoft Teams

Microsoft Teams was also used in forms of communication, collaboration, and file-sharing (Microsoft n.d.). The invitations to the meetings could be sent here. The status meetings with the supervisor and client were often held on Teams. Teams have a chat function where different types of files were shared with the client and supervisor efficiently.

Microsoft Word

Initial drafting of the bachelor’s thesis was written in Microsoft Word due to its tools that can help collaborative writing. Within Word, multiple group members can edit the document at the same time, while always seeing the most updated version. Additionally, tasks can be assigned and shown where something needs to get done very precisely. The group members can add comments to other group members’ work. This will help with the improvement of the project.

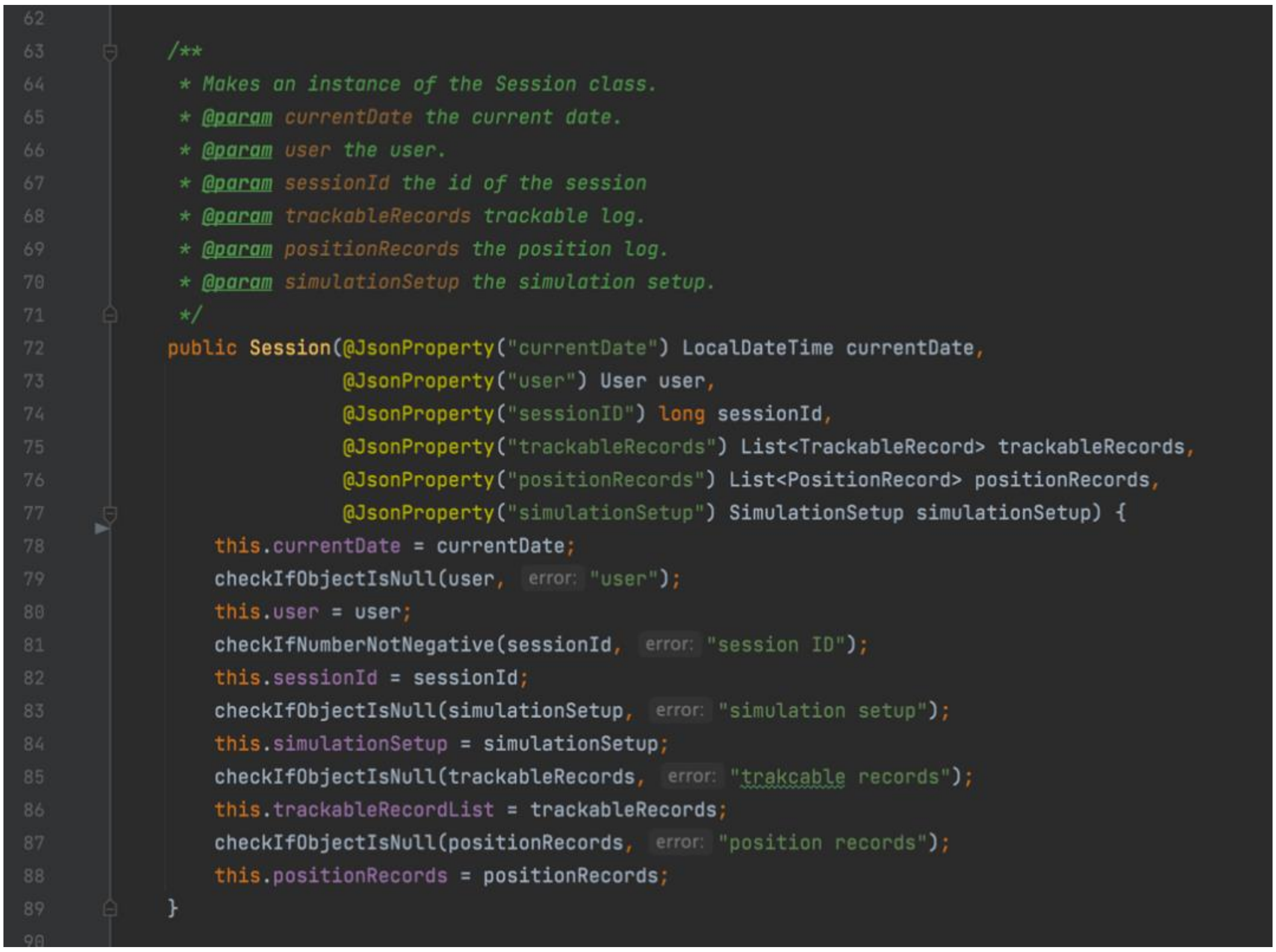
Documentation

Documentation is an important part of the coding process. The group members can easily understand what the others are trying to do with their code, without having to ask. This makes the process more structured. The group have been strict on documentation from the beginning. The reason is that this will help to debug the code if needed later and learn from their own mistakes. It will also help future developers understand the code and make it easier for them to maintain and update it if needed.

Backend documentation

The backend documentation contains extensive documentation following the Java documentation standards. Excluding the endpoints, the documentation is comprehensive enough for the application to be understood without having to read the code itself.

The code in the backend is documented well. Everything is coded so that the other group members easily can understand what the code means without questioning the code. The code is written in Java, with the IDEA tool IntelliJ.



```java
    /**
     * Makes an instance of the Session class.
     * @param currentDate the current date.
     * @param user the user.
     * @param sessionId the id of the session
     * @param trackableRecords trackable log.
     * @param positionRecords the position log.
     * @param simulationSetup the simulation setup.
     */
    public Session(@JsonProperty("currentDate") LocalDateTime currentDate,
                   @JsonProperty("user") User user,
                   @JsonProperty("sessionID") long sessionId,
                   @JsonProperty("trackableRecords") List<TrackableRecord> trackableRecords,
                   @JsonProperty("positionRecords") List<PositionRecord> positionRecords,
                   @JsonProperty("simulationSetup") SimulationSetup simulationSetup) {
        this.currentDate = currentDate;
        checkIfObjectIsNull(user, error: "user");
        this.user = user;
        checkIfNumberNotNegative(sessionId, error: "session ID");
        this.sessionId = sessionId;
        checkIfObjectIsNull(simulationSetup, error: "simulation setup");
        this.simulationSetup = simulationSetup;
        checkIfObjectIsNull(trackableRecords, error: "trakcable records");
        this.trackableRecordList = trackableRecords;
        checkIfObjectIsNull(positionRecords, error: "position records");
        this.positionRecords = positionRecords;
    }
```

Coding example 1 Example of code documentation in the backend (Session.class)

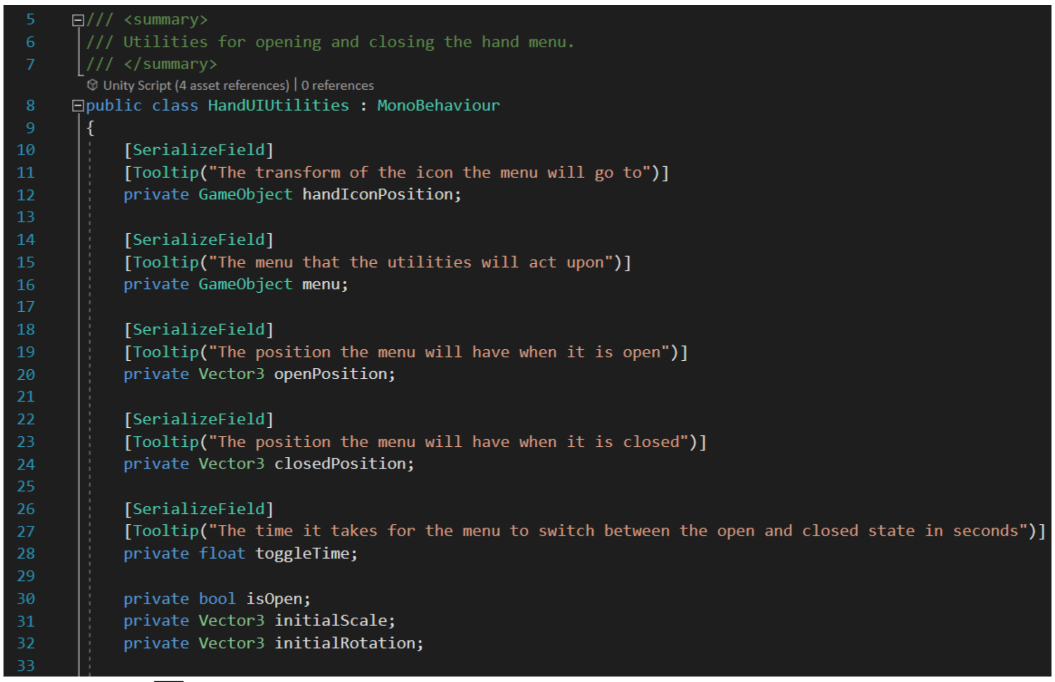
Frontend

The code in the frontend is documented the same way that the backend is documented. This is written in React JS and in the code editor Visual Studio Code. All the methods and classes are documented on the frontend.

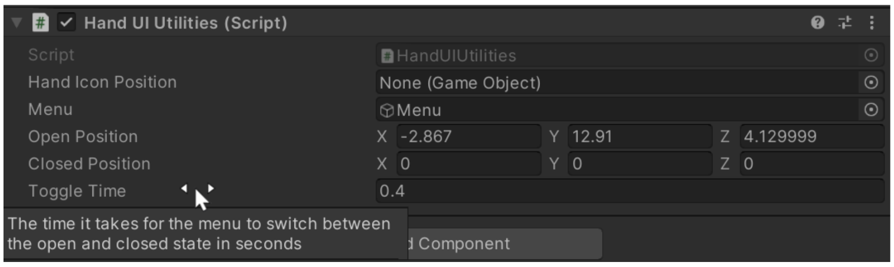
Unity

Several of the code snippets used in Unity are components, which can be accessed by developers in the Unity Editor, possibly without ever seeing the code. Because of this, Unity has various functionality for presenting components in an understandable way for Unity developers.

Components are attached to game objects and accessed via the "inspector". The inspector allows users to edit variables and run certain functions. Public variables are visible in the inspector by default. Private variables addressed with a "SerializeField" header will also be visible, while maintaining their private accessibility. "Tooltip(String)" lets users mouse-over the variable in the inspector in order to see a tooltip. Additional properties include but are not limited to: Headers that segment variables in the inspector, number ranges that create convenient sliders and graphing.



```
/// <summary>
/// Utilities for opening and closing the hand menu.
/// </summary>
Unity Script (4 asset references) | 0 references
public class HandUIUtilities : MonoBehaviour
{
    [SerializeField]
    [Tooltip("The transform of the icon the menu will go to")]
    private GameObject handIconPosition;

    [SerializeField]
    [Tooltip("The menu that the utilities will act upon")]
    private GameObject menu;

    [SerializeField]
    [Tooltip("The position the menu will have when it is open")]
    private Vector3 openPosition;

    [SerializeField]
    [Tooltip("The position the menu will have when it is closed")]
    private Vector3 closedPosition;

    [SerializeField]
    [Tooltip("The time it takes for the menu to switch between the open and closed state in seconds")]
    private float toggleTime;

    private bool isOpen;
    private Vector3 initialScale;
    private Vector3 initialRotation;

```



Hand UI Utilities (Script)

Script	HandUIUtilities
Hand Icon Position	None (Game Object)
Menu	Menu
Open Position	X -2.867 Y 12.91 Z 4.129999
Closed Position	X 0 Y 0 Z 0
Toggle Time	0.4

The time it takes for the menu to switch between the open and closed state in seconds

Coding example 2 Example of how serialized variables are displayed in the Unity inspector. (HandUIUtilities.class)

In the case of coding example 2, the component is used to play an animation where the menu flies and shrinks into the hand of the user. Through serialized fields, the developer utilizing the script can change the timing and positioning of the close/open animation in runtime, which can be helpful during developer testing.

3.2 Hardware and tools

3.2.1 HMDs

Meta Quest Pro

The Meta Quest Pro was used to develop and test the VR demo. This HMD contains in-built eye-tracking hardware, which can be calibrated to fit different users. Due to this headset being new on the market, and having a high cost, no more than one Meta Quest Pro could be issued to the project. It was initially used for general development and testing of the VR demo until an additional HMD was provided. When the group got access to an additional HMD, the Meta Quest Pro was provided to the developer that would work most closely with eye-tracking implementation. (Midthus and Staverløkk 2022)

Meta Quest 2

The Meta Quest 2 was also used to perform testing and development of the VR demo (Meta n.d.). This HMD does not contain any eye-tracking hardware, but in the case of this project, it is functionally identical in all other aspects. This allowed two developers to work separately with the VR demo. This HMD was issued to a separate developer which was assigned to tasks not related to eye-tracking, such as world-space menus, hand interactions, and scene building.

HP Reverb G2 Omnicept Edition

The group was originally issued an HP Reverb G2 Omnicept edition (Midthus and Staverløkk 2022). This Windows Mixed Reality HMD contains built-in eye-tracking, pupillometry and heartrate and can be accessed through the Omnicept or Tobii SDK. This HMD only saw usage early in the project and was later dropped for two reasons. The first reason was licensing problems with Tobii and Omnicept. The second reason was compatibility with the Oculus HMDs. The group had issues with getting the VR demo to run using both Mixed Reality and OVR, so it was decided that development for the HP Reverb G2 Omnicept edition would be dropped in favour of Meta Quest 2 and Pro.

3.2.2 Computers

Two laptop computers were provided for the project. The computer with the lowest specifications, the Alienware computer was set as the target hardware of the demo project. The computer specifications are as follows:

IHB Lenovo Legion computer

Processor: AMD Ryzen 7 5800H-Processor

Graphics card: Nvidia GeForce RTX 3080

RAM: 16GB DDR4 RAM

Storage: 1000GB SSD

Operating System: Windows 11

IIR Alienware computer

Processor: Intel i7-12700H

Graphics card: Nvidia GeForce RTX 3060

RAM: 16GB

Storage: 500GB SSD

Operating System: Windows 10 Education.1 HMDs

3.3 VR development

3.3.1 VR

Since the ship simulator developed by Morild Interaktiv AS is based on Unity, the project also had to be based on the same engine. This way the project would encounter the same issues that Morild would and solve them so that they don’t have to.

Unity

Unity is a game development engine used for video games, digital twins, as well as other real-time 3D applications.

The engine supports a wide variety of platforms, such as consoles like the PlayStation 4, or the most common operating systems, like Windows 10 or Linux, as well as 17 other platforms. This project is primarily built to work with Windows 10 specifically (Khronos n.d.)

Unity also supports several VR devices through the OpenXR standard. Along with Unity’s XR Interactions toolkit, this allows for setting up a basic VR application with head tracking and controller interactions relatively quickly (Unity n.d.)

In scripting and scene composing, Unity uses the entity component system paradigm. Objects in the scene are stored as game objects. The game objects can have multiple C# scripts attached to them as components, creating different behaviour. One common example of a component is the transform component, which specifies the scale, position, and rotation of a given game object.

To make these components cooperate, Unity contains an event and broadcast system. Components can listen to events, so that when an event is invoked, a given method can be called in the component. In the inspector, a developer can add as many function calls of different objects as needed. One example from our project is the TaskListProducer component which listens to Task objects to show them in the GUI. When Task objects are changed, added, or removed, they invoke the UpdateLists() event, which updates the TaskList and TaskListProducer. Broadcasts work similarly but are limited to the gameobject the broadcast is called in, as well as its children. Broadcasts also apply to specified functions in components, while events have a broader usage.

VR headset SDK Licensing

The project from the beginning had two headsets that where capable of doing eye tracking in VR. The Hp Reverb G2 Omnicept edition and the Meta Quest Pro. Both headset's support OpenXR, but eye tracking is not implemented into that library yet.

The Omnicept headset is the first one that was considered since the university had acquired the headset for the previous industry project. But a problem that was encountered was that the library Tobii XR never responded to the request for an educational license in both the industry project and in the beginning of the bachelor thesis. Another problem was encountered with the Omnicept but is also described in the industry project that is an attachment.

Quest pro is then the option that we landed on. Since there is no need for an educational license.

OpenXR

OpenXR is an open standard that aims to solve fragmentation in the XR development space (Khronos n.d.). There is a large number of devices in the XR space which used differing standards. This made development of XR applications for different hardware configurations cumbersome. OpenXR acts as a middleware so that developers can ship their products to more devices while having to do less porting work.

Oculus Integration SDK

Some aspects of the demo utilize the Oculus Integration SDK, such as the hand models and eye tracking input. Ideally, use of this SDK would be limited, so that use of multiple HMDs and controllers could be simplified.

3.3.2 3D Modeling

Blender 3D

Blender is a 3D content creation suite. It is free and open source under the GNU General Public License (Blender Freedom n.d.). In the project, Blender is used to model and texture 3D models for use in the VR demo. These models are exported as.fbx and imported into Unity. Due to differences in material standards, material properties such as metallic can be lost in import/export. Because of this, some inspection and post processing is required on model import to Unity, to ensure the imported model looks as expected.

Unity primitives

Unity contains a suite of shapes, referred to as primitive objects. These primitives can used to construct basic scenes and objects very quickly. This was used to prototype the scene layout, and carriable objects before the scene was modelled with the external tool Blender.

Blender modelling

Blender was used to do the more detailed 3D modelling in the demo. While Unity allows for placement of primitives and basic assigning of materials, Blender allows for more sophisticated editing. Some conveniences include texture coordinates that can be edited to fit objects of strange shapes more easily, objects that can be mirrored and spun to reduce repetition, model smoothing, and assigning materials to any small part of an object. This increased the possibility of detailed models such as keyboards, boats, and a lighthouse in the distance.

External assets

External assets were utilized where time or experience was lacking. One area that used a large amount of external assets was texturing. These were sourced via public domain texture websites such as polyhaven.com and ambientcg.com. Producing PBR materials of the quality found in the public domain would take time and camera hardware which would be a challenge for the group to source. It was important to only retrieve public domain assets, as they can be redistributed in an open repository.

An additional external asset that was implemented into the project was the VR hands. These are taken from the Oculus SDK. These hands were already rigged and animated, which saved time for modelling other assets.

Shaders and visual effects

The shaders were made in Unity using a tool that is called shader graph. This shader graph allows developers and graphical designers to make shaders by connecting nodes instead of writing shaders manually (Unity n.d.). Almost the same tool is also developed for visual effects where the user uses nodes and logic to trigger actions (Unity n.d.).

The reason that we used these tools instead of writing shaders and visual effects with code is that this approach was a more understandable way of making these effects for newcomers.

3.3.3 Software Development Kit (SDK)

An SDK is a collection of tools and programs provided by vendors to make building projects for their target platforms more accessible (Yasar 2022). SDKs can contain documentation, use case example projects, APIs, software libraries, and other data that can be helpful for developing on a given platform.

3.3.4 C#

C# is an object-oriented programming language. It shares close similarities to Java and is the chosen language for scripting in the Unity engine (BillWagner, gewarren et al. 2023). It is also used to build .NET applications and takes extensive use of .NET libraries.

3.4 Backend

When it comes to the backend and different technologies, the group quickly decided to pick a framework and language that the group already have a lot of experience with. The reason being that this project is a POC. So, the projects backend is based on java, spring and hibernate. SonarLint and Checkstyle was used to adhere to coding standards and make our code more readable. The members also had experience with these plugins already. Docker is used for the deployment of both the backend and the frontend. This is because the services then run in their own environments that cannot affect each other.

3.4.1 Java

Java is an object-oriented programming language that was developed in 1991 by James Gosling (IBM n.d.). The most useful usage of this language is the "compile once, run everywhere" feature that it has with its own virtual machine. This combined with its wide usage makes this a good choice for many applications.

3.4.2 Spring boot

Spring Boot was chosen since the group already has experience with this framework. At the same time, it's based on Java and has support for the REST-API (spring by VMware Tanzu n.d.). The framework was also chosen since the NCA asked for endpoints that could be accessed at an early stage so that the data could be processed by other applications or algorithms. And the framework used for the frontend is also based on REST requests to display data.

Representational State Transfer (REST)

Representational state transfer (REST) is an architectural style that waits for a request before it responds to it with the according information (visheshy2ey 2022). The REST webservices uses HTTP requests to send and receive data to an end client no matter the programming language on the client and server side. This way ensuring that data can be accessed if the request is sent with a HTTP request. The system is also easily scalable and maintainable since there is not direct code to interfere in changes made to the API.

3.4.3 Spring security and tokens

Since the data gathered is personal data that must be stored for later analysis the backend needed some security. Therefore, we used Spring Security and implemented JWT tokens. Spring security is a framework that adds authentication, authorization, and protection for spring applications (spring n.d.). This way the user must be authenticated before they can access the stored data.

3.4.5 JPA and hibernate

Java Persistence API or JPA is a specification that defines how data can be persisted in Java applications (baeldung 2022).

Hibernate is an open-source Java framework that implements the JPA specification to store ORM objects (javaTpoint n.d.). It handles the setup and query of information from and to the database. The data itself is stored in a relational database through hibernate.

3.4.6 MySQL Database

MySQL is one of the most popular open-source source databases (Vyas 2023). This is a relational database management system. Some benefits of using MySQL are that it provides good security and dependability, is high efficiency, and has excellent workflow control and scalability on demand.

3.4.7 CheckStyle

CheckStyle is an open-source tool that inspects Java code against a set of rules (Checkstyle 2023). This tool can be integrated into a Java project via Maven and IDE plugins. The sole purpose is to save time by doing this task for humans and enforcing a coding standard.

3.4.8 SonarLint

SonarLint is an IDE extension that fixes coding issues. It is like a spell checker for coding (SonarSource n.d.). This extension highlights the bugs and vulnerabilities. It will also give a guide so that it is possible to fix them before committing code.

3.4.9 DBeaver

DBeaver is a free and open-source universal database tool for developers and database administrators (DBeaver n.d.). DBeaver has a lot of benefits. It is easy to use and implement a database. It is also free, open-source, multiplatform and supports any database if the user has a JDBC driver.

3.4.10 Docker

Docker is an open-source project that allows developers to simply deploy their project on a container. Then they can run on the host operating system Linux. *"The key benefit of docker is that it allows user to package an application with all of their dependencies into a standardized unit for software development* " (Srivastav n.d.). The containers have note high overhead unlike virtual machines. They do make the use of underlying system and resources more effective (Srivastav n.d.).

3.5 Frontend

The frontend was decided to be a webpage since it can support both mobile and desktop views with little altering of the webpage. But the webpage itself had to use the RESTful API that the backend has implemented. Therefore, we chose to react as a framework to host the webpage and expand the groups knowledge. The OpenBridge standard was decided to be used to make prototyping of the webpage easier. It's also a standard that the users already are familiar with at the NCA. The group made a prototype in Figma so that the client could give feedback during the development process. With their integrated comment function, Figma made it easier to get feedback on smaller parts of the project.

3.5.1 HyperText Markup Language (HTML)

HyperText Markup Language (HTML) is the first of the three standard web technologies used in the modern web. HTML is the markup language that enables us to organize and assign significance to the content on our website (OnkarRuikar 2023). With HTML, we can create various elements such as paragraphs, headings, and tables, as well as incorporate multimedia elements like images and videos into our web pages. HTML gives the webpage structure and content which then can be manipulated.

3.5.2 Cascading Style Sheet (CSS)

Cascading Style Sheet (CSS) is the second of the three standard web technologies. CSS is a style sheet language used to apply formatting and visual design to HTML content (OnkarRuikar 2023). It allows developers to set properties such as background colours, font styles, and layout, enabling them to create a visually appealing and organized web page. By using CSS, developers can manipulate the structure and style of the webpage in an effective manner.

3.5.3 JavaScript

JavaScript (JS) is a versatile scripting and programming language that empowers the incorporation of sophisticated functionalities on a web page (OnkarRuikar 2023). Whenever a web page goes beyond displaying simple static information and includes dynamic features such as timely content updates, interactive maps, animated graphics, etc. JavaScript is often the driving force behind changes on a webpage. JavaScript forms the third layer as part of the triad of standard web technologies and makes a webpage more responsive.

Prettier

To make the coding process in React JS more efficient, a coding formatter called Prettier was used. This is an opinionated code formatted with support for JavaScript. This tool removes all the original styling and ensures that all the outputted code conforms to a consistent style (Prettier n.d.). Prettier is useful for the group because there was less time spent on formatting the code when it was nested. The magic key binding is pressed, and the code is formatted like that. It is also easy to adapt to the project. In Visual Studio Code it is an extension that needed to be downloaded.

3.5.4 Node.js

Node.js is an open-source server environment (w3schools n.d.). It uses JavaScript on the server. It stands out in building fast and scalable network applications. This server environment can generate dynamic page content through JavaScript. It can create, open, read, write, delete, and close files on the server. Node.js can collect form data. It can also add, delete, and modify the data in a database. The web applications can be in real-time, with two-way connections. JavaScript was used in combination with Node, to get a working application.

To start the application, a package manager is used. This is a package manager for Node.js packages or modules. A package in Node.js contains all the files needed for a module. A module is a JavaScript library, which can be used in a project.

3.5.5 React JS

React JS is a JavaScript library for building user interfaces (React n.d.). It is declarative, meaning that it is painless to create interactive UIs. The framework will be able to optimize the rendering process by updating only the necessary components whenever your data undergoes changes. React JS is also component-based. This means that it can build encapsulated components that manage their own state, and then compose them to make complex user interfaces. The complex data is seamlessly transmitted throughout the application by writing component logic in JavaScript instead of templates. Another benefit to React is that is no need to rewrite code since there are already a lot of ready-to-use codes out there.

3.5.6 OpenBridge Design System

The OpenBridge Design System is a set of design guidelines. This system is owned by the Oslo School of Architecture and Design who has developed it with support from the OpenBridge Design System consortium (OpenBridge 2020). By using this standard, there was a set of rules that needed to be followed. The use of OpenBridge made it easier to develop since it consisted of a set of guidelines to follow and ready-made tools to use. Their package was ready to be implemented in React JS.

3.5.7 Figma

Figma is a prototype tool that allows interaction between how a client can use the design. Prototypes like this are a good way for previewing interactions, share ideas and discuss ideas on how the final product will look like.

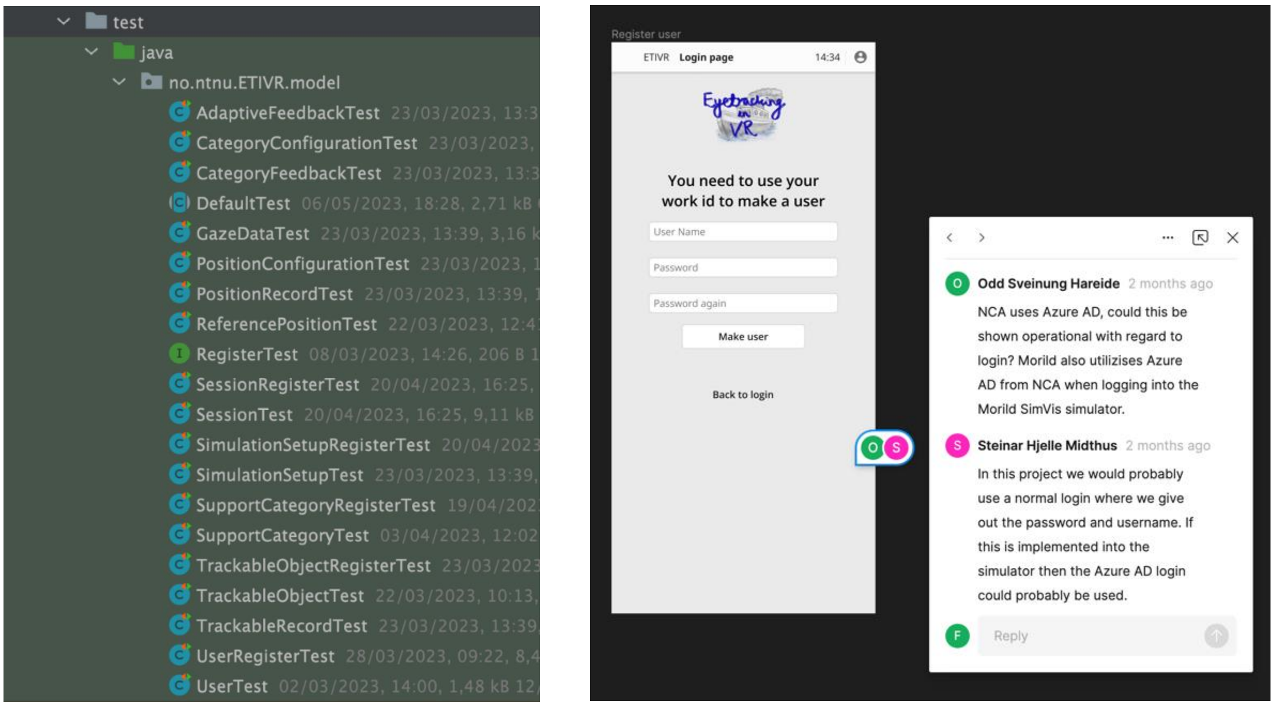
3.5.8 Chart.js

Chart.js is a part of JS Graphics for making graphs chart. Chart.js is a JavaScript library for making HTML-based charts (w3schools n.d.). It is free and one of the easiest visualization libraries for JS. The library contains of bar, pie, donut, bubble, mixed, radar, area, line, and scatter charts.

3.6 Testing

3.6.1 Unit testing

Unit testing consists of testing small parts of a code, often methods. These tests are often automated and are done by a testing framework like JUnit (Java). The tests are made in their own test classes and will be run as a part of the building process. To make the process more effective, a default test class was made and used in each class. This default class had methods that would check for different types of errors and display the errors in tests in a better way. Every object had its own test class. The registers for each object were also tested. These types of tests were a central part of the backend structure. If one of the tests failed, that meant that something was wrong in the code.



```
test
  java
    no.ntnu.ETIVR.model
      AdaptiveFeedbackTest 23/03/2023, 13:3
      CategoryConfigurationTest 23/03/2023,
      CategoryFeedbackTest 23/03/2023, 13:3
      DefaultTest 06/05/2023, 18:28, 2,71 kB
      GazeDataTest 23/03/2023, 13:39, 3,16 k
      PositionConfigurationTest 23/03/2023, 1
      PositionRecordTest 23/03/2023, 13:39,
      ReferencePositionTest 22/03/2023, 12:4
      RegisterTest 08/03/2023, 14:26, 206 B 1
      SessionRegisterTest 20/04/2023, 16:25,
      SessionTest 20/04/2023, 16:25, 9,11 kB
      SimulationSetupRegisterTest 20/04/2023
      SimulationSetupTest 23/03/2023, 13:39,
      SupportCategoryRegisterTest 19/04/202
      SupportCategoryTest 03/04/2023, 12:02
      TrackableObjectRegisterTest 23/03/2023
      TrackableObjectTest 22/03/2023, 10:13,
      TrackableRecordTest 23/03/2023, 13:39
      UserRegisterTest 28/03/2023, 09:22, 8,4
      UserTest 02/03/2023, 14:00, 1,48 kB 12
```

You need to use your work id to make a user

Odd Sveinung Hareide 2 months ago
NCA uses Azure AD, could this be shown operational with regard to login? Morild also utilizises Azure AD from NCA when logging into the Morild SimVis simulator.

Steinar Hjelle Midthus 2 months ago
In this project we would probably use a normal login where we give out the password and username. If this is implemented into the simulator then the Azure AD login could probably be used.

Figure 5 An overview of all the tests in the backend on the left and figma prototype on the right.

3.6.2 Usability testing

Usability testing was used to make sure that the client got what they wanted. Tools like Figma were used to gather feedback on the final product's appearance. The client also got a demo of what the simulation looks like at the end of every week. The reason was that the client could get an overview of how far we have gotten in the project, and how they potentially could implement the project on their own.

The usability tests reveal problems and bugs with the project that the group might have missed. The user tests were conducted with at least one group member acting as the guide, and another taking notes and spectating how the user approached the tasks. With this method, several issues were discovered with the VR demonstration.

The users got the VR headset adjusted to their heads and calibrated for eye tracking. Then they entered the simulation and got tasks that they had to finish. During these tasks, their eye data movements were tracked and gathered. At the end of the user tests, the users got to see their data visualized on the graph on the webpage.

4. Results

When a trainee is put into a simulated environment the performance varies based on their experience. An expert might have different behaviours and habits that they have earned through their experience. Therefore, if their experience and habits can be analysed by using their gaze patterns the performance of a less educated trainee can improve significantly.

4.1 Development process

4.1.1 Eye tracking

Problem

How can the headsets eye tracking be implemented into the VR demo and gather the necessary data for each category of objects.

Process

Even though the eye tracking data could be gathered, the project had to find a way to do it efficiently in Unity, store it and visualize it in the virtual environment. Since the setup with the VR simulator is supposed to be used by one person the system had to be easy to use and self-explanatory.

Also, how the tracking of all the objects in a scene can be done might be difficult if some parts can be environment can be changed. Objects in the virtual world might move and thus change positions from once it was observed if raw eye tracking datapoints was used.

Possibilities

One possibility to gather eye tracking data in Unity is to take the raw position of the observation and store it locally according to the object that it hit as an observation instance.

Another possibility is to define each object of interest as an AOI and track the total amount of fixations and fixation time for each object. This decreases the number of instances that are made per observation since the object is tracking itself.

Solutions

The solution that was chosen to mainly be used is the AOI solution. Where each object that is observed has a component that oversees their own fixations and fixation duration. These objects can then be observed by any headset since the object only had to be alerted when its observed.

4.1.2 Persistence and API

Problem

The data that is gathered in the VR demo needs to be persisted on a server and be easy to access for the users. The backend also needs to have a REST API that can handle requests so that the data can be analysed using different software.

Process

Handling and storing large amounts of data could be a complex task, and in this the risk of changing the code since it’s a proof of concept is high. Therefore, the backend had to be flexible to changes to make the implementation as painless as possible.

The backend also had to support a REST API and handle HTTP requests from users, and at the same time get data from the VR demo and store it. Since the sessions themselves has a user, the data would also need to be connected in some way to indicate what users did what session.

Possibilities

One possibility was to use a monolithic structure for the backend. This way both the webpage and REST API could be hosted by the server and no framework was needed to host the webpage. The VR demo could use the REST API to store and get information from the server, and the webpage would have implemented security measures.

Another possibility is to make a backend that only has a REST API and session-less token authentication. This would also reduce the complexity and dependence on Spring Boot implementation and the website could be hosted on another framework.

Solutions

The possibility that made it easiest to implement was the backend with only a REST API to do operations. This was chosen since the backend already needs to support an API and if both the API and website had to be deployed on the backend both had to use the same logic. That could result in a lacking feature set on the REST API.

4.1.3 Webpage

Problem

As the information is gathered in the VR demo and stored on the backend it also needs to be displayed on a webpage to give the user an objective feedback based on their performance. This webpage should also have the capability to compare against an expert’s performance.

Process

The data that is going to be displayed on the webpage must be visualized to a user in a manner that is understandable and easy way. This is done so that the user does not have to depend on an analyst to understand the data.

The different seats and metrics also had to be taken into consideration since each seat has its own metric per object. The user must be able to select all the seats and change between the different metrics.

Possibilities

One possibility is to show the data in tables that can show the different categories and the metric per position in one place. This table would also be large and maybe a bit confusing, but the user could then change the metric in a dropdown or button bar to see fixations and fixation duration.

The second possibility is to show the data directly in a graph. Each graph would be able to change between the different eye-tracking metrics and positions. This change of metrics would be represented in a dropdown menu or button bar like the positions.

Solutions

The solution chosen was the graph representation of the data. This is because it shows the data in a visual way instead of plotting a lot of data in a rather chaotic table. It also allows for the data to easily be compared against each other since all that is needed is another dataset that can be visualized in the same graph with different colour coding.

4.2 Eye tracking implementation

The eye tracking in Unity is designed to be universal in some matters. Since the headset can be changed out without affecting the logic on a too radical scale when it comes to the main implementation. All the eye-tracking classes and basic logic have their own controller classes that are responsible for reacting to the events inside the simulation.

The controllers in Unity are designed to separate the game interactions and the main logic. These controllers are then components of the game object that defines a certain behaviour for the object to do on the logical classes it holds. Some of these controllers utilize the façade pattern to separate the logic and the reactions from the game object. This is done to ensure loose coupling and high cohesion between the logic classes and controllers in Unity.

The managers in Unity are designed to handle the large interactions between the different objects in the virtual environment. Managers also handle the communication between themselves.

4.2.1 Eyes and raycasting

The Meta Quest Pros movement SDK does not have any API methods to show what the user is looking at by default, but it can take the position and rotation of the eye and show it in Unity world space. To get what the eyes are looking at a gaze vector is calculated. This vector is found by calculating the centre position between the eyes and their combined rotation and direction. Then using raycast or sphere cast to shoot a ray and hit the game objects and their colliders.

After an object is hit by the raycaster controller broadcasts a message on the game object to alert it of its new state. This broadcasting system is included in Unity and calls this method on all components of the object. This way ensuring that an object might have more components reacting to the same method in different ways. One component can track the fixations and fixation duration, and another can spin the object on its axis.

The broadcasted message is also transmitted to the children of this game object so they might react accordingly.

The raycaster object class is an abstract this way the implementations can support multiple eyes or just one. So, if the user only has their right eye the new implementation can only use one point of reference like the single raycaster object. Instead of two points of reference like the Eye caster object. When the raycaster object hits a point in space the "new" position is only changed if the distance to the "old" hit is larger than 0.06 meters. This value can also be adjusted on the raycaster object by using the editor.

The raycaster object also implements the observer pattern so that other classes can subscribe and get watched objects. An example of this can be seen in the figure 6 where the white circle is a component that subscribes to the position of the eyes to visualize the gaze hit point. The red arrow illustrates the point of origin for the raycast and its direction.

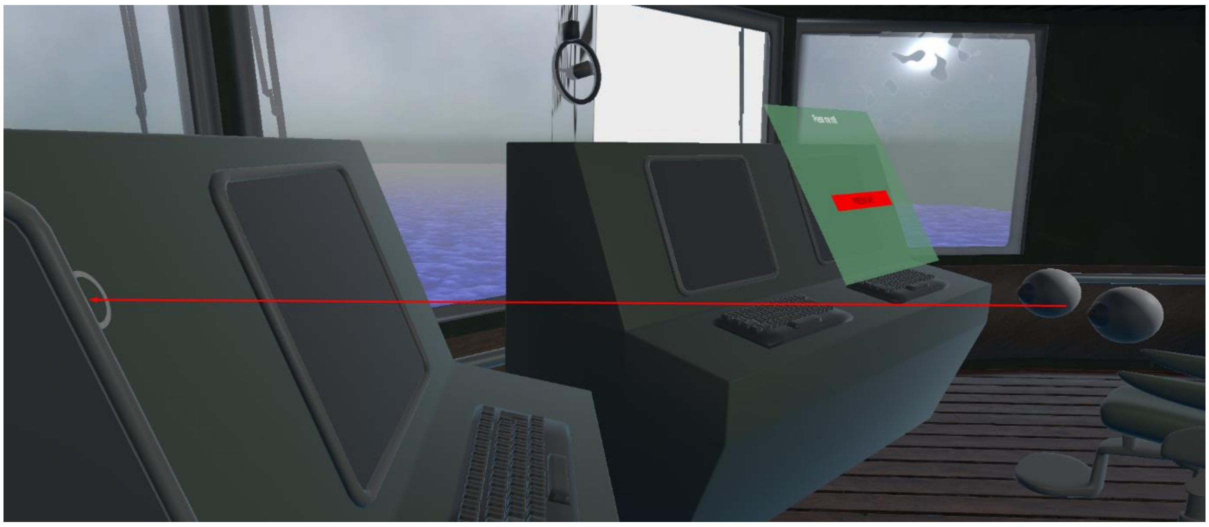


Figure 6 Visualization of the gaze hit and ray casting from the eyes.

4.2.2 Position and trackable object

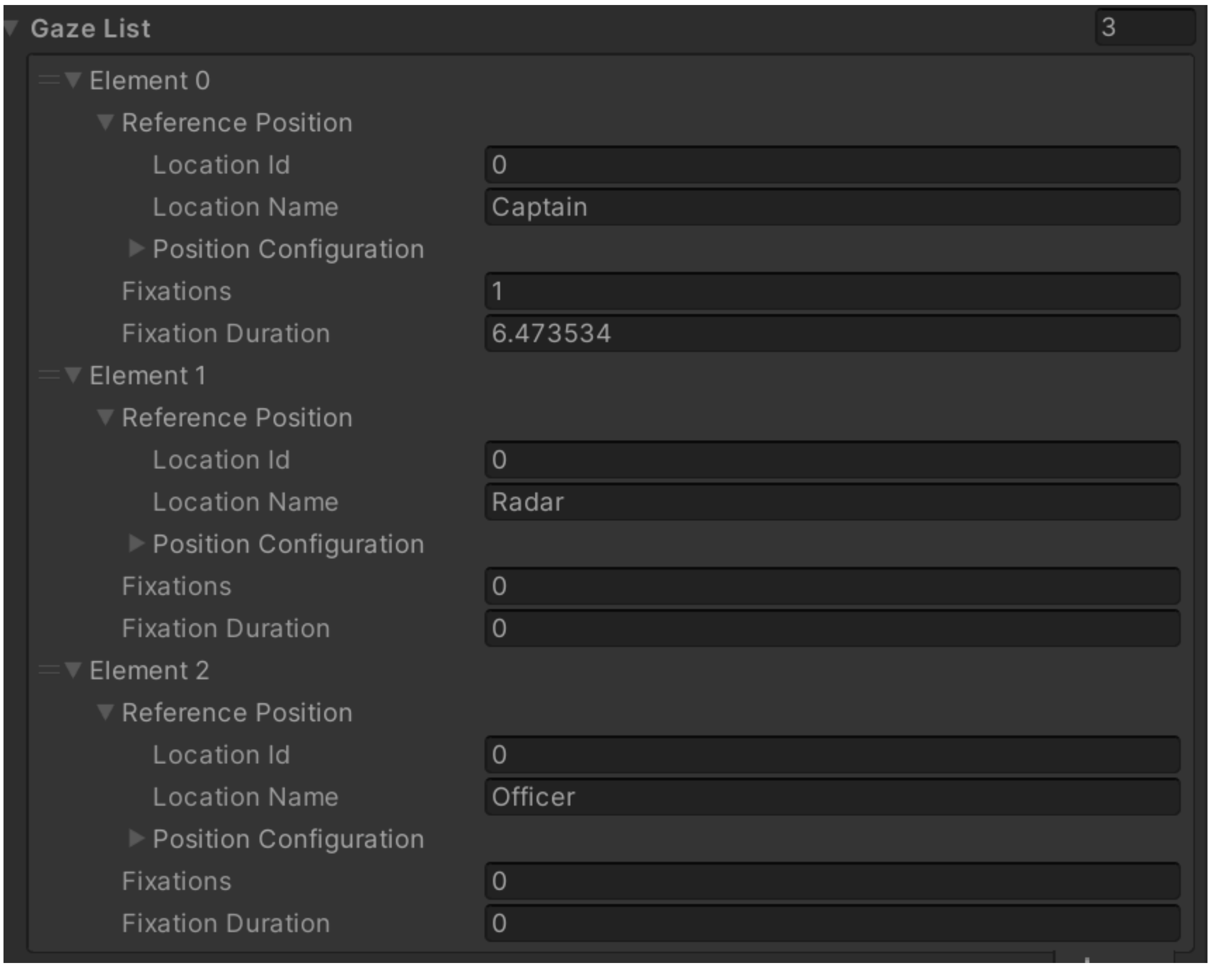
A game object that is defined as an area of interest is called a trackable object. This object only identifies the object by a unique name, unique numeric identifier, and category. The category of the object is used to calculate the total metrics for each category.

The positions that the user can teleport between are static since the simulator that Morild has made also contains static positions. Therefore, they are predefined with a unique name and unique numeric identifier. When an observation is made on a trackable object, the position of the user is also taken into consideration and stored in the gaze data class.

The gaze data class records the number of fixations and fixation duration for a certain position for each trackable object. By having this class, the performance per category of seat 1 can be evaluated based on what the trainee looked at when they were in seat 1. This object is then stored inside a trackable record.

Since the position and trackable objects are logic classes used to identify the different objects the data per session is stored in other objects called records. The record for the trackable object holds the different gaze data instances for each position. Almost the same is done for position where the record holds the time the user has been standing at the position, and the feedback received on the position.

Feedbacks are records for a certain time that are calculated throughout the session. The feedback itself is based on the trackable objects and uses the category to find out the fixation duration for each position. If the total time of all the categories is less than the recorded position time the remaining time is added into the "other" category. The feedback is automatically recorded every 10 seconds and can be adjusted.



```
Gaze List                                   3
  Element 0
    Reference Position
      Location Id               0
      Location Name             Captain
      Position Configuration
    Fixations                   1
    Fixation Duration           6.473534
  Element 1
    Reference Position
      Location Id               0
      Location Name             Radar
      Position Configuration
    Fixations                   0
    Fixation Duration           0
  Element 2
    Reference Position
      Location Id               0
      Location Name             Officer
      Position Configuration
    Fixations                   0
    Fixation Duration           0
```

Figure 7 example of gaze data separation based on the position

4.2.3 Sessions and predefined setups

All the objects that are in the scene are stored as a simulation setup. This class holds the positions and trackable objects to simplify the storage of the sessions. Instead of storing each trackable object once per session, it is stored in a predefined object on the server to reduce the number of objects per session. This also allows for multiple setups that have their own unique defined areas of interest. The session itself is stored inside the session object. The session object holds the simulation setup that the session was done in and the records of all the positions and trackable objects. Thus, only holding the recorded data of that specific session.

Simulation setup controller has the responsibility of gathering the trackable objects and reference positions that are child objects of this controller. Thus, the setup is collected when the VR demo is started.

4.2.4 Point records

The point record is when the point of contact between the cast ray and the object is hit and placed into a class. This class contains the world position, trackable object component if the object hit has one and the position relative to the object the ray hit. Then the record is stored inside a container class that holds the transform of the hit object. This is done to increase the efficiency of deploying the objects instead searching for the objects each time they should be visualized.

The point of interest recording extends this point record class and adds an order identifier so that the points of interest can be displayed with what order the user has been looking at the different points for. It also indicates how many points that have been placed since they can differ in fixation duration for each point.

These basic point records are used to make a point cloud. This cloud is an implementation of a "heatmap" that uses transparent dots of red to show where the user has been looking. The brighter the red colour, the more the point recordings are displayed in that area. These red dots are displayed using visual effects and are relative to each object that was observed.

The points of interest recordings are used to generate a gaze plot based on what the user has been looking at for a period. The top number that can be seen on figure 8 is the order of the points that counts from 0 to n. The number on the bottom is the number of seconds that the user looked at each point.

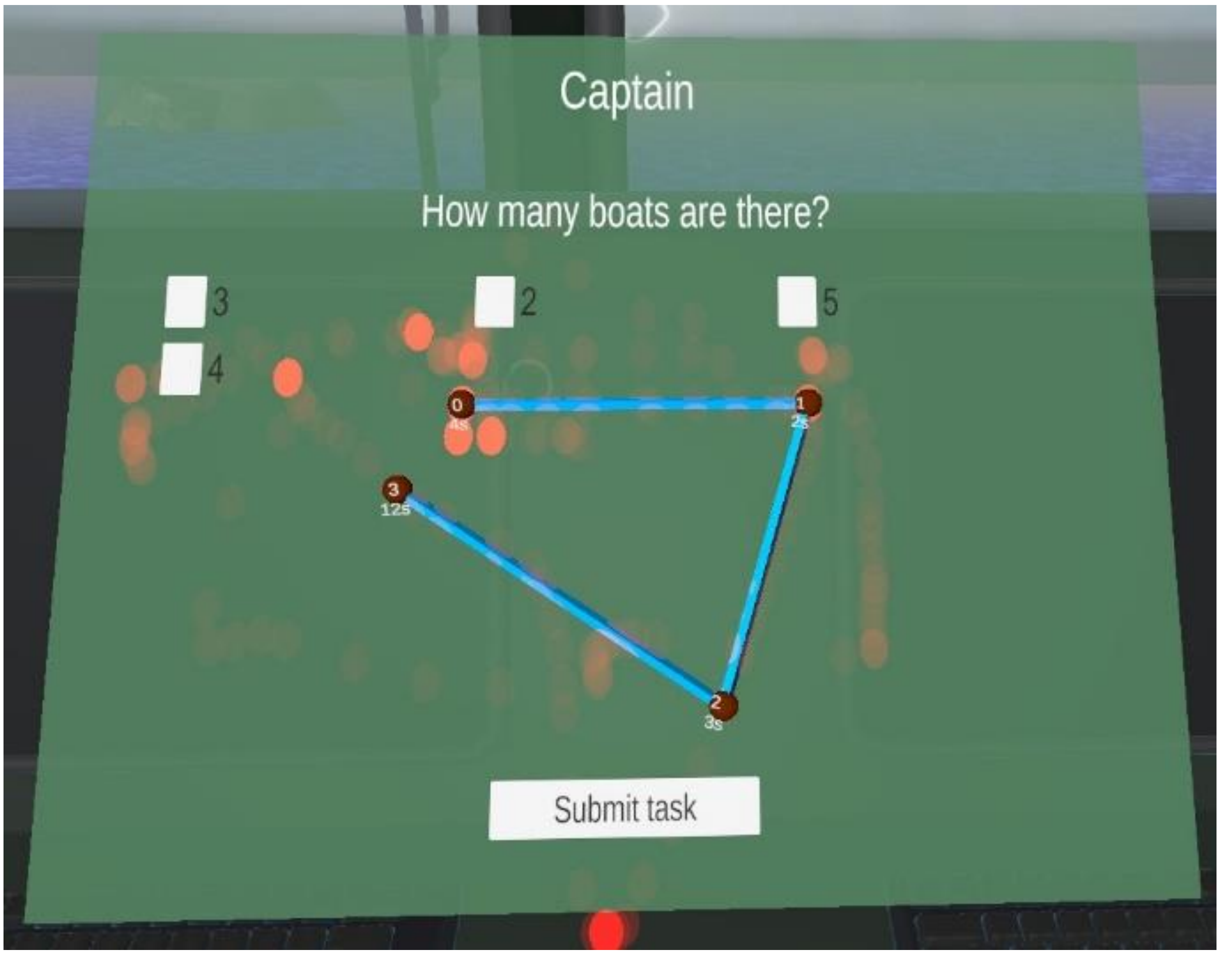


Figure 8 Question panel with both point cloud and points of interest displayed.

4.3 Interactions

4.3.1 Seat teleporter

The seat teleporter is an interactable that the user can teleport to by aiming at them with the hand controllers and pressing the left or right trigger buttons. The teleporter utilizes the component "XRSimpleInteractable" from the XR toolkit for receiving input, as well as keeping track of when a user is aiming at it. When occupying a seat, the player is assigned to the role of that seat, which is reflected in the eye-tracking data. When the user is aiming at a seat teleporter, its collider will gradually become visible to indicate that this is an object the user can interact with.

4.3.2 Grab Interactable

There are several objects that can be picked up and manipulated by the user via the "Grab interactable" component. By aiming at a grabbable object and holding the respective trigger button, the object will teleport into the hand of the user. Custom transforms ensure that the object is grabbed in a position and rotation one would expect the object to be held in, such as from the handle of the coffee cup. The user can rotate and move the grabbable object by using the joystick of the controller that holds the object. This utilizes the "Collider" component to check if rays from the controller hit the grabbable object.

4.3.3 Settings and Tasks menu

A wristwatch is placed on the left hand of the player. On the face of the watch, an icon is placed which indicates a menu can be opened. The player can point at this icon and press the trigger to open the Settings and Tasks menu. The hand icon and menu use several components from the XR interaction toolkit. The XR UI canvas allows UI to be shown to the VR user in world space, "TrackedDeviceGraphicRaycaster" allows the VR controller to use continuous raycasts to interact with UI elements.

This menu contains one tab for each of the menus. The menu can be moved by interacting with the orb underneath the menu (using a grab interactable without physics enabled) and closed either by pressing the "x" icon or pressing the watch icon again.

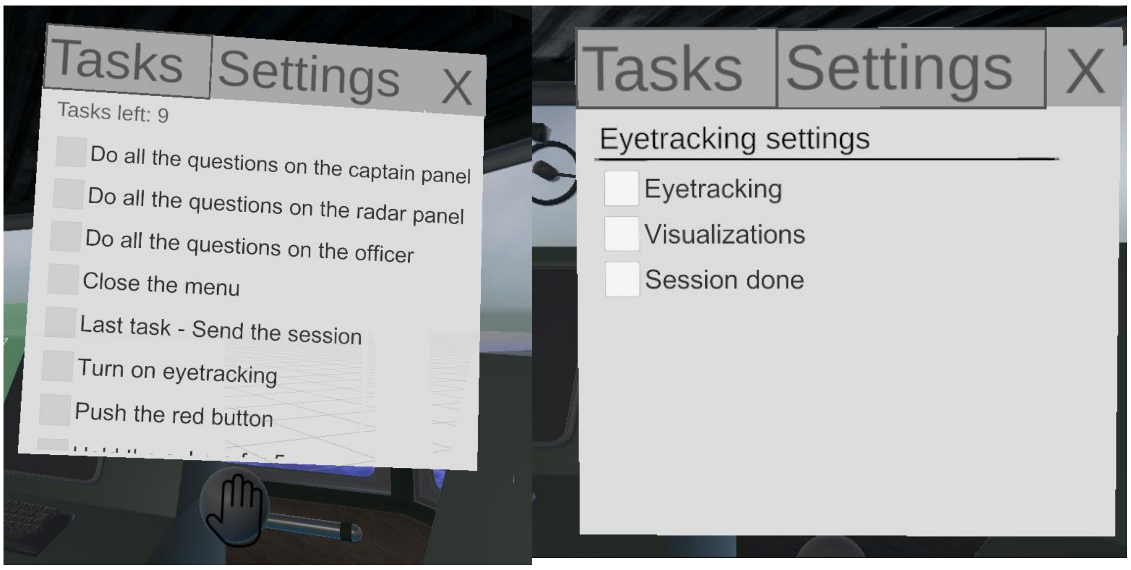


Figure 9 The tasks and settings menu tabs in VR

4.3.4 Tasks

There are different tasks that can be done in the scene to guide the user through and have a scenario that could collect data. The task class is the abstract definition of what a task should be. This is done to have different tasks with different finish conditions for each. An example of this Is the questions class that is done when all the question options are correctly answered according to their correct answer Boolean. If one of them is invalid the whole task is not completed or done. The task order can also be forced if the setting is enabled in the task manager. This way ensures that each task must be completed in order.

Two other task classes are simple tasks and timed tasks. The timed task looks at the time that the user has interacted with the task itself and records it. It can either be in continuous mode or just record time when the task is done. In continuous mode the task needs to be focused on for that specified amount of time or else the task resets the time of the task. The simple task on the other hand is a class that only needs to set the task to be done in its method.

The task controller is also abstract to support multiple different task behaviours. To alert the GUI and different elements that a task has changed their status the controllers has implemented the unity event system. This system also utilizes the unity event system. When a task is completed, the controller alerts the listeners of the state change through the event. An example of the question task controller can be found in figure 8.

Task manger is used to make different default tasks for the user. An example of this is the hold and gaze tasks that could be added. Where each object can be dragged into the configuration field and add different tasks to that object. The manager also utilizes the singleton pattern if the task order is forced. This is done so that the tasks can check if it’s their turn before they are completed.

4.3 3D models

The final scene contains a collection of different 3D models which were utilized to make the scene replicate the bridge environment of a ship at sea.

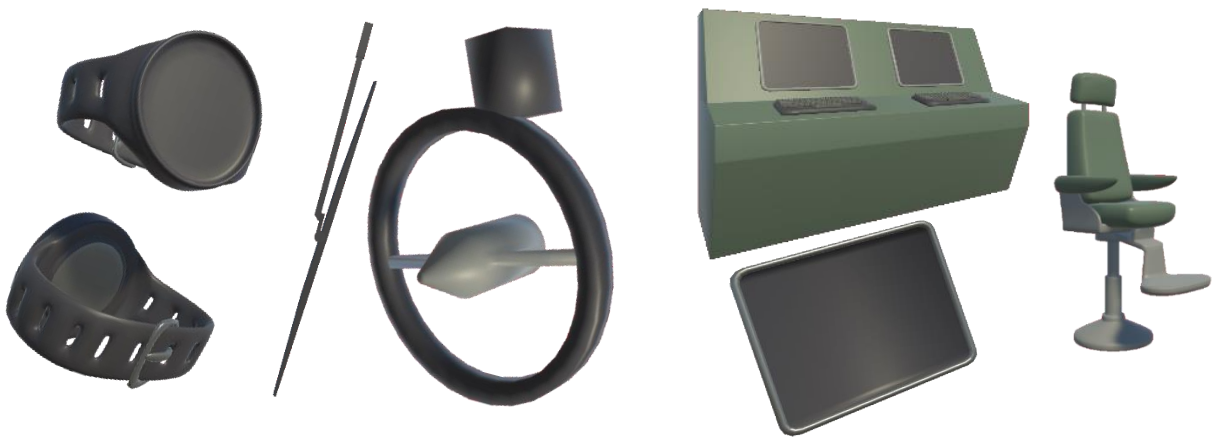


Figure 10 Wristwatch, windshield wipers, terminal and seat teleporter

Wristwatch

Wristwatch model, attached to left left hand of the user to make the "open menu" button look more natural on the user's hand.

Windshield Wipers

Windshield wiper and clear view wiper models. Attached to the windows of the bridge to provide some detail to otherwise empty space.

Computers, seat teleporters

Figure 10 displays a bridge and tablet computers. They resemble the different instrument terminals one could see in a bridge. These are placed in the scene to act as interactable objects. UI screen overlays are placed on top of the black displays.

A chair is also shown in figure 10. These chairs can be teleported to by the user. A lot of real bridge operations happen from a standing perspective, but because of our seated experience with teleportation as movement, we needed a model that could act as a "teleport" waypoint.

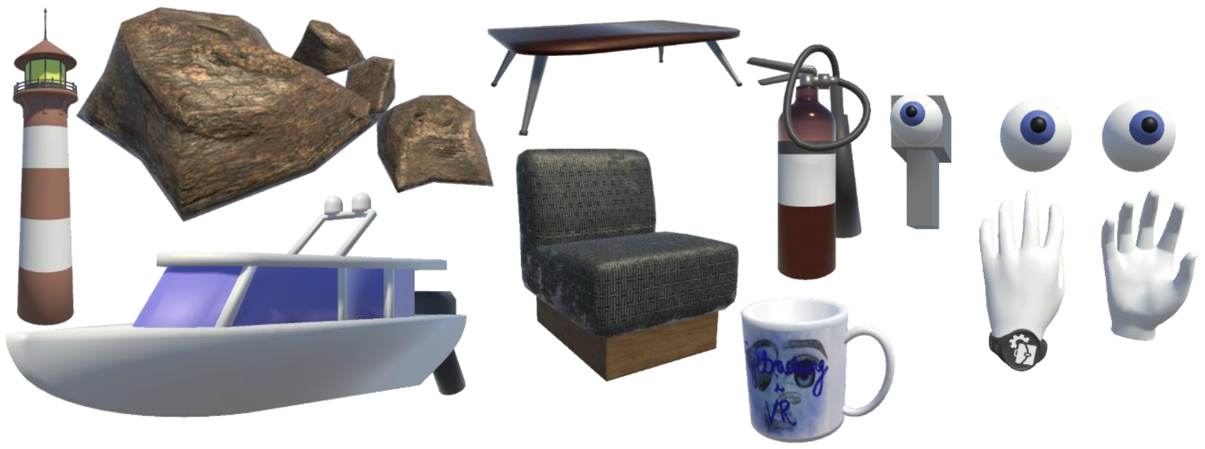


Figure 11 Exterior models, interior props, player models

Exterior environment

Three 3D models were used for the exterior. The lighthouse is placed atop the rocky crags, with boats travelling around on the ocean surface. A shader was attached to the lighthouse to make a rotating spotlight effect. A component was attached to boats to make them move around in the scene.

Interior props

Four 3D models were added to make the back of the bridge more interactive and visually appealing. The wooden table and "sofa" chairs are set up as a casual break area on the bridge.

The fire extinguisher and coffee cup are interactable objects that the user can pick up and carry around.

Player models

The player controller consists of two hands, a set of eyes, and (optionally) an eye "gun". The eyes are not visible to the player perspective and are simply used for debugging in scene view. The eye gun behaves similarly to the eyes and can be used for testing when an eye-tracking HMD is unavailable.

The hand models can open and close based on player input. The watch is placed on the left hand of the player and can be used to open the settings and tasks menu as indicated by the suitcase/cog icon.

Shaders

There was a total of three shaders made for this project that are currently in use. The first one is the light cone effect of the lighthouse. This shader has been made to flicker and represent a light that changes its brightness when it moves towards and away from the user. There is also used a Fresnel effect to simulate the brightness when the lights rotation is closer to the user. An example of what it looks like can be seen in figure 12 with the visual graph included.

The water shader is another shader that is used in the demo. This shader is made to simulate the water in the demo and alters the vertex height of the mesh to simulate moving water. A colour shift is also added to the upper and lower bound of the shader, simulating the real-life colour changes of a water surface.

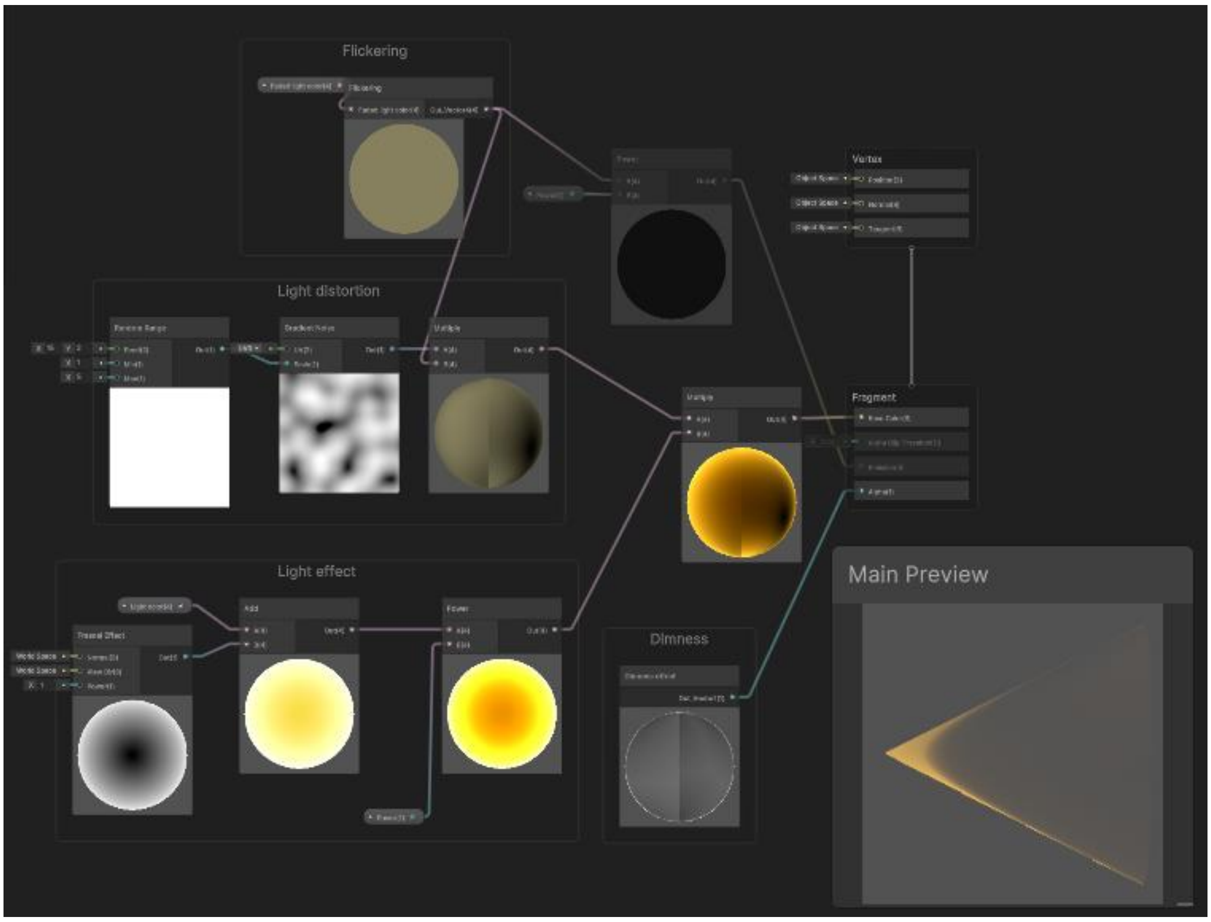


Figure 12 Example of the light cone shader. The cone itself is in the bottom right corner

Visual Effect

The only visual effect that was added is the red dot that appears in the point cloud when it is rendered. This is done to save performance since too many game objects lags the world. This can be seen in 8, where the red dots are the visual effect.

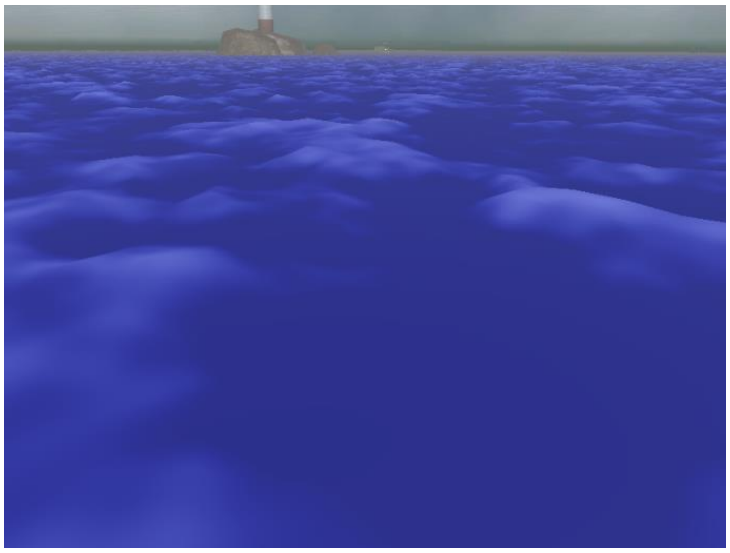


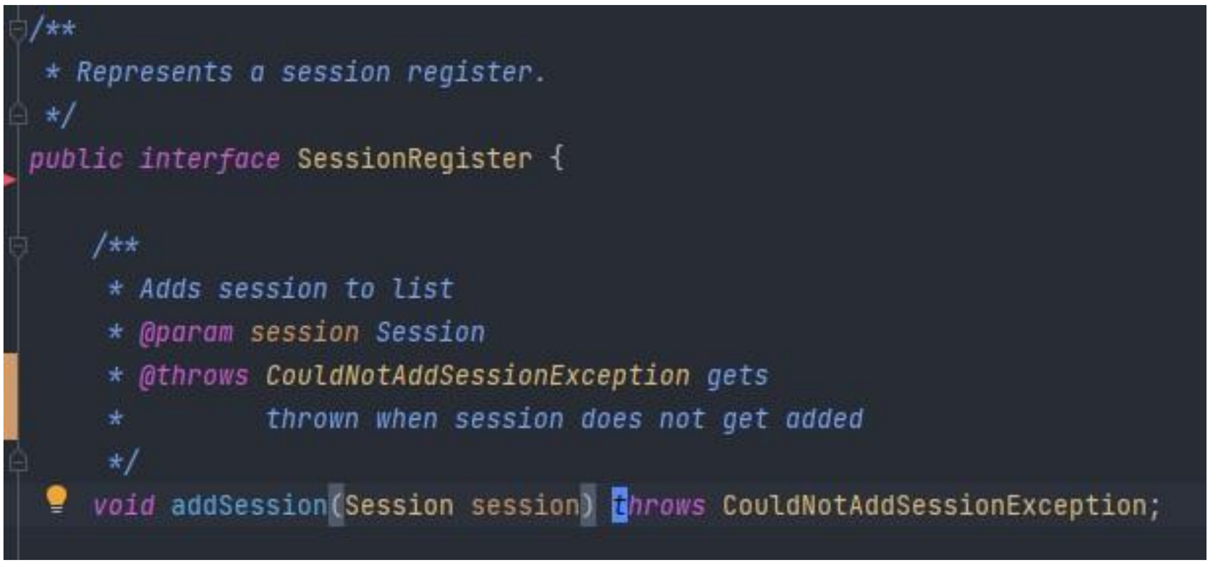
Figure 13 The water shader in the demo

4.4 Persistence and API

The backend is used to store the sessions done in the VR demo for later analysis. The backend itself is a REST API so it supports data fetching from other applications other than the website. The Spring framework is used to persist the data through hibernate, configure authentication, and endpoint management.

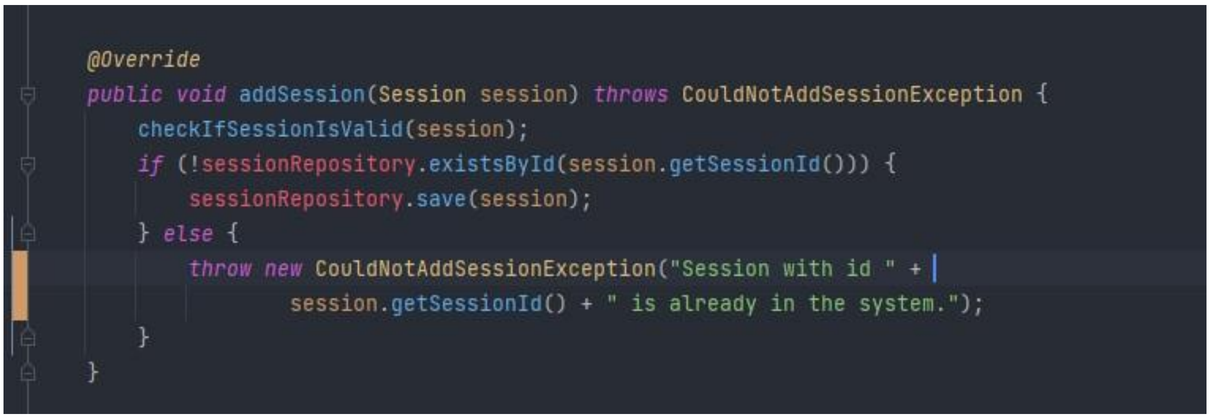
4.4.1 Registers and Services

The register of backend has registers that are abstract definitions of what methods they should have to be implemented. This is done so that the refactoring cost is reduced if the project's storage should be changed in the future. An example of this is the sessions register found in coding example 3 and 4. Where the interface just defines a method on coding example 5 and the SessionService class implements it in coding example 6.



```java
/**
 * Represents a session register.
 */
public interface SessionRegister {

    /**
     * Adds session to list
     * @param session Session
     * @throws CouldNotAddSessionException gets
     *         thrown when session does not get added
     */
    void addSession(Session session) throws CouldNotAddSessionException;
```

Coding example 3 The sessions interface with one of its defined methods. (SessionRegister.class)



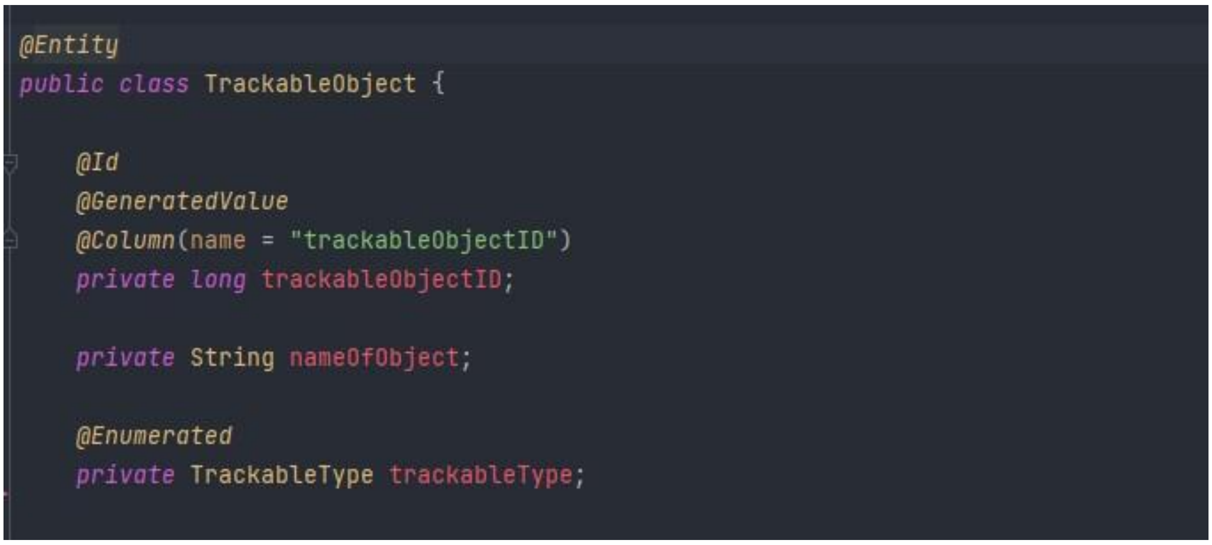
```java
    @Override
    public void addSession(Session session) throws CouldNotAddSessionException {
        checkIfSessionIsValid(session);
        if (!sessionRepository.existsById(session.getSessionId())) {
            sessionRepository.save(session);
        } else {
            throw new CouldNotAddSessionException("Session with id " +
                    session.getSessionId() + " is already in the system.");
        }
    }
```

Coding example 4 Implementation of the sessions register interface. (SessionService.class)

4.4.2 Database and repositories

The service classes use repositories from hibernate to do basic CRUD operations on the database for each class that needs a database connection.

The database itself is automatically generated using annotations on the logic classes in java. An example of this can be seen in coding example 5. To assure that each object gets its unique identifier the GeneratedValue annotation is used.



```
@Entity
public class TrackableObject {

    @Id
    @GeneratedValue
    @Column(name = "trackableObjectID")
    private long trackableObjectID;

    private String nameOfObject;

    @Enumerated
    private TrackableType trackableType;
```

Coding example 5 the trackable objects class and annotations used to store it in the database. (TrackableObject.class)

The number of tables that is generated is 22 where one of them is responsible for autogenerating identifiers also called primary keys. The primary and foreign keys are used to generate a relationship between the different tables. The repositories that are needed for hibernate is also able to have custom methods to retrieve data based on other factors like the username.

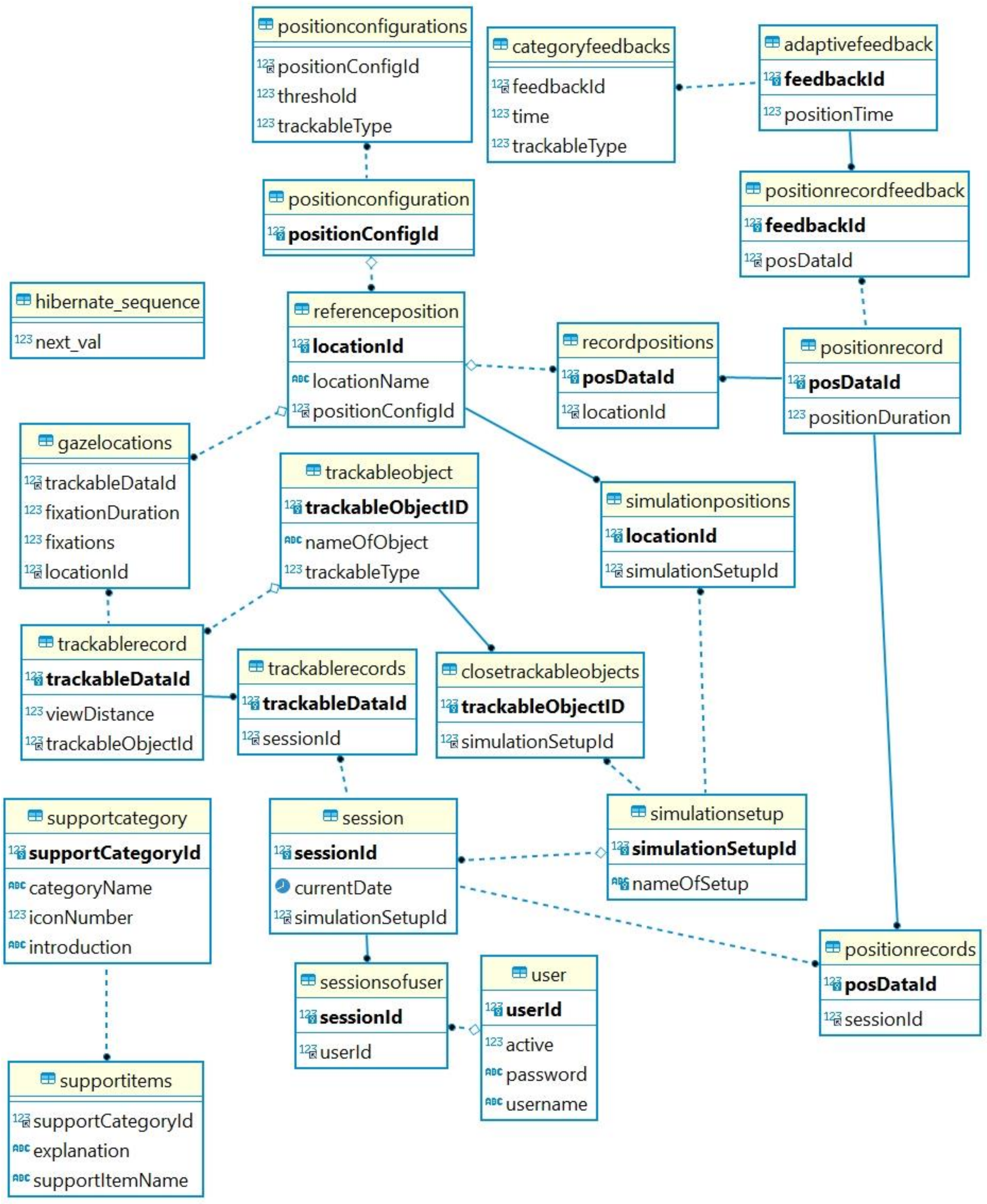
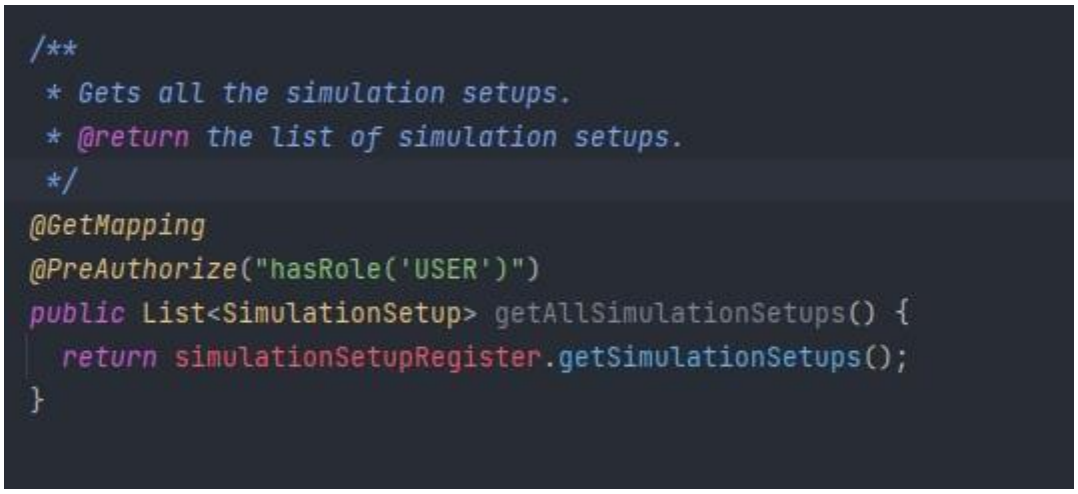


Figure 14 Automatically generated Entity relationship diagram of the database and its tables

4.4.3 Security

The data that is gathered through eye tracking is classified as personal data since its biometric data. That is why the application has implemented security in form of JWT authentication. If a trainee wants to compare their session against another or just simply access their own sessions, they must be logged in. This is done by JWT authentication which is implemented through the spring security package. If the user is signed in, they get a basic role called "User" and then they can access their own sessions, post new sessions, and access other people’s sessions. The preauthorize annotation is used to check if the user Is logged in and has access to a resource before doing any operations.



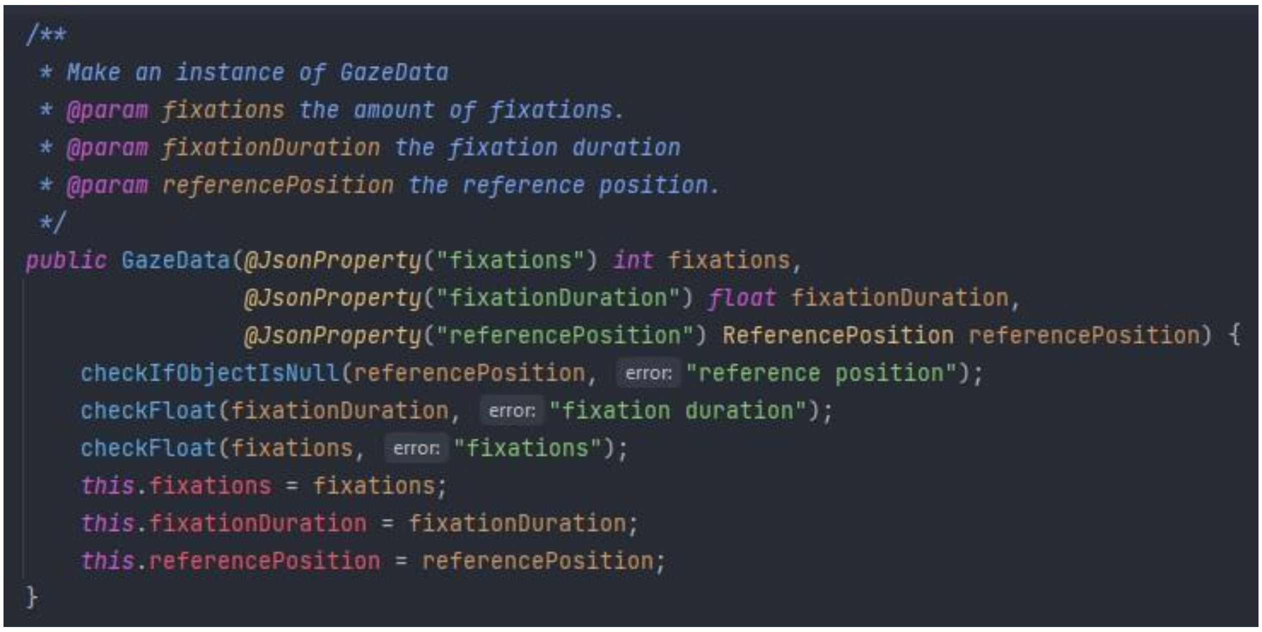
```java
/**
 * Gets all the simulation setups.
 * @return the list of simulation setups.
 */
@GetMapping
@PreAuthorize("hasRole('USER')")
public List<SimulationSetup> getAllSimulationSetups() {
  return simulationSetupRegister.getSimulationSetups();
}
```

Coding example 6 PreAuthorize used to get all the simulation setup. (UserController.class)

4.4.4 REST controllers

The REST controllers on the backend handles the endpoints that are vital to get and store the data from the VR demo. Each endpoint path like user has their own controller to handle requests from that revolve around the users, and each rest controller also has services that match the model class that it is responsible for.

When a post or put request is received, the object is not built in the parameters of spring, even though it is supported. This is done since SonarLint once showed that taking a class as a parameter is prone to malicious attacks. So instead, an object mapper is used from the Jackson library to correctly convert a class from JSON to actual code by using predefined constructors. These constructors are predefined by the group, and all have their input checked so that valid values are set for the instance. The only difference is the JsonProperty annotation that marks each variable and what their JSON name should be. If the number of parameters that the object mapper has access to is more than the assigned to the constructor an exception is thrown. An example of the JsonProperty annotation can be seen in coding example 7.



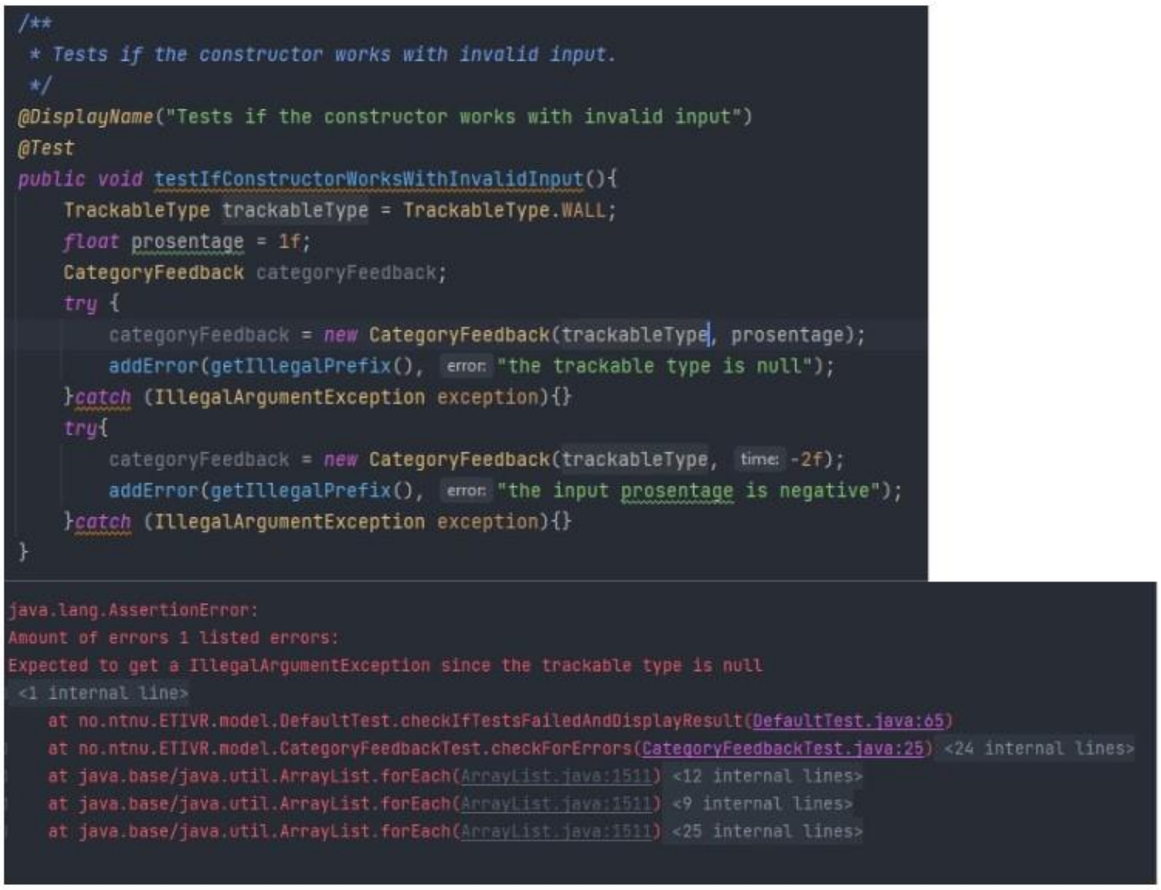
```java
/**
 * Make an instance of GazeData
 * @param fixations the amount of fixations.
 * @param fixationDuration the fixation duration
 * @param referencePosition the reference position.
 */
public GazeData(@JsonProperty("fixations") int fixations,
                @JsonProperty("fixationDuration") float fixationDuration,
                @JsonProperty("referencePosition") ReferencePosition referencePosition) {
    checkIfObjectIsNull(referencePosition, error: "reference position");
    checkFloat(fixationDuration, error: "fixation duration");
    checkFloat(fixations, error: "fixations");
    this.fixations = fixations;
    this.fixationDuration = fixationDuration;
    this.referencePosition = referencePosition;
}
```

Coding example 7 Example of JSON property used in a constructor. (GazeData.class)

4.4.5 Testing

The backends testing is based on Midthus´ previous experiments with tests. Since each test should have its own method, the number of tests needed to test a given method with different parameters can be tiresome. Therefore, this project uses a system that tests either positive or negative outcomes for all the parameters of a method in one test.

In top part of coding example 8 we can see an example of this where both the trackable type and time are tested in the same method but in separate try and catch blocks. If the input value is invalid and there is no exception, the "addError" method will be called. This method uses a string builder to store the error strings until the test is completed. When the test is done the method with the annotation "afterEach" is called and checks if the string builders' length. If the length is larger than zero, then an error has been added to the builder and the number of failed tests is listed as well as their cause. See coding example 8 for a failed test example.



```java
/**
 * Tests if the constructor works with invalid input.
 */
@DisplayName("Tests if the constructor works with invalid input")
@Test
public void testIfConstructorWorksWithInvalidInput(){
    TrackableType trackableType = TrackableType.WALL;
    float prosentage = 1f;
    CategoryFeedback categoryFeedback;
    try {
        categoryFeedback = new CategoryFeedback(trackableType, prosentage);
        addError(getIllegalPrefix(), error: "the trackable type is null");
    }catch (IllegalArgumentException exception){}
    try{
        categoryFeedback = new CategoryFeedback(trackableType, time: -2f);
        addError(getIllegalPrefix(), error: "the input prosentage is negative");
    }catch (IllegalArgumentException exception){}
}
```

```
java.lang.AssertionError:
Amount of errors 1 listed errors:
Expected to get a IllegalArgumentException since the trackable type is null
 <1 internal line>
    at no.ntnu.ETIVR.model.DefaultTest.checkIfTestsFailedAndDisplayResult(DefaultTest.java:65)
    at no.ntnu.ETIVR.model.CategoryFeedbackTest.checkForErrors(CategoryFeedbackTest.java:25) <24 internal lines>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <12 internal lines>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 internal lines>
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <25 internal lines>
```

Coding example 8 Failed test since the input type is not null. Testing method above and error underneath. (CategoryFeedbackTest.class)

4.5 Information and visualization

The product was developed to look at how eye tracking can be used to give objective feedback to a trainee and compare them to an expert´s performance. The main pages that can be found is sessions, session overview, login, register user, profile, and support page. The website, which was made in a combination of React JS and ChartJS It also followed sets of requirements according to the OpenBridge System standard as well.

4.5.1 Design Principles

Most of the design was made according to the OpenBridge system. The client wanted an application that was user-friendly and straightforward to navigate. In addition to these design principles, other design principles were considered as well when designing the application. This is according to Don Norman's design principles. In this section, both principles will be reviewed.

There was a total of two different prototypes since the open bridge standard was followed the first draft of the webpage. When the first prototype changed to the open bridge standard the figma prototype was sent to a member of the Open Bridge project.

They did a design review and checked that the first draft met their expectations. The result was not checked with a design review due to time constraints.

4.5.2 Register and login page

The application is using token-based authentication. The token is stored locally on their devices and when the token is validated, the user gets to log in to the application and use all the different functions available. This token is then stored in a cookie for the session. The token has an expiration time that can be used by the server to check if they still have access before doing any operations. The user can register with a username and password. Then login to the application and get access to all the features.

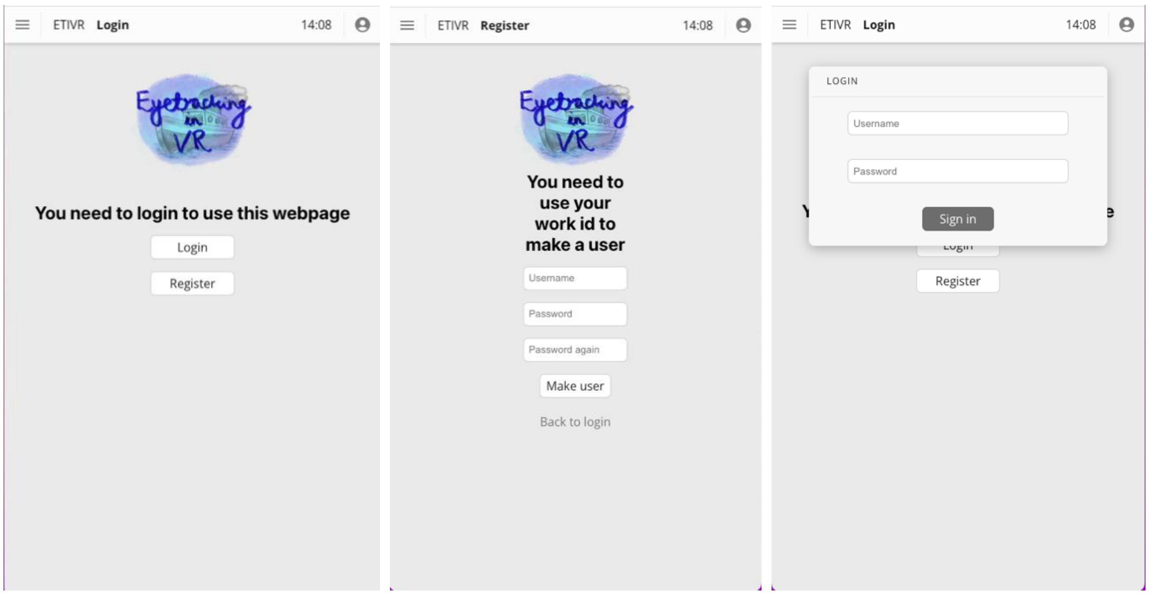


Figure 15 Home page, register page and login page on the webpage

4.4.3 Sessions

The main purpose of the application is to compare eye-tracking data from an unexperienced user against an experienced user. In the application, the users will have a full overview of the sessions for the users and the option to compare them to against each other. The user can filtrate the sessions by date, simulation type and users. The user needs to be logged in to do so. By clicking the "See Session" button, the user navigates to the session overview page.

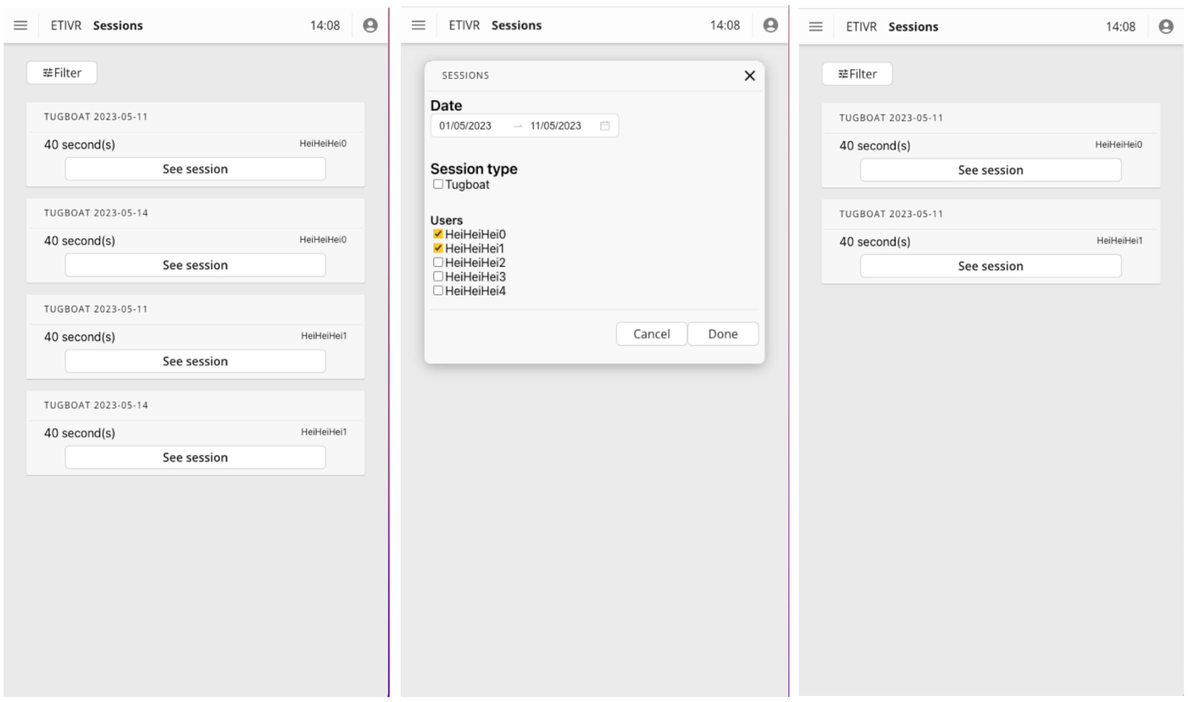


Figure 16 Example of the sessions page with an overview of some sessions on the left, the filter option in the center, and the filtered results on the right side.

4.4.4 Session Overview

This is the page where the user can see the selected session and compare them to each other. The information about each session is stated in the cards above current stats. On the top of the page, there is an option to choose which seat to display the data from. By clicking the "compare" button, the user can choose navigates to the sessions and can choose another session to compare the chosen session against.

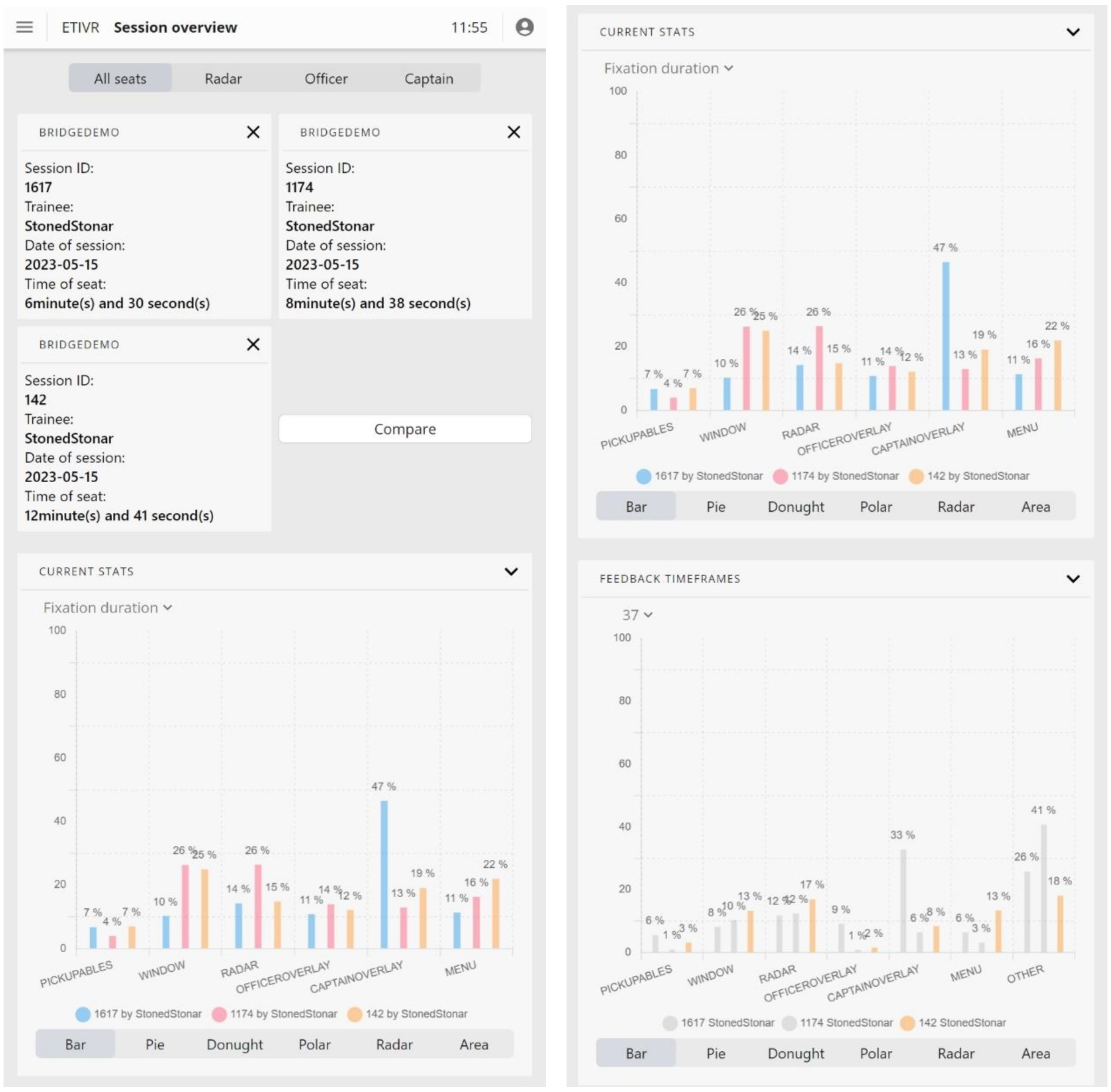
The current stats show the total for all the objects that are observed at the end of the session for that seat. The metrics can also be changed by using the dropdown menu on the current stats graph. The graphs themselves are displayed by ChartJS for react and are not made by us.

Feedback timeframes is the recorded feedback over time that is shown on a graph. There is a dropdown menu where the user can choose the time to compare the two sessions against each other. Each feedback is taken 10 seconds apart, so the math works out to "n * 10 seconds" to find out what time the selected option is for. In figure 16 the selected time is after 6 minutes.

The graphs can also be changed for both cards on the button bar below the graphs.

4.4.5 Profile

The profile page shows the details of the user. The user can see their name, the number of sessions done, the total time used to do these sessions and the different simulation types that has been done. This page serves as a consolidated view to get an overview and of the users' achievements.


ETIVR Session overview 11:55

All seats | Radar | Officer | Captain

BRIDGEDEMO

Session ID:
1617
Trainee:
StonedStonar
Date of session:
2023-05-15
Time of seat:
6minute(s) and 30 second(s)

BRIDGEDEMO

Session ID:
1174
Trainee:
StonedStonar
Date of session:
2023-05-15
Time of seat:
8minute(s) and 38 second(s)

BRIDGEDEMO

Session ID:
142
Trainee:
StonedStonar
Date of session:
2023-05-15
Time of seat:
12minute(s) and 41 second(s)

Compare

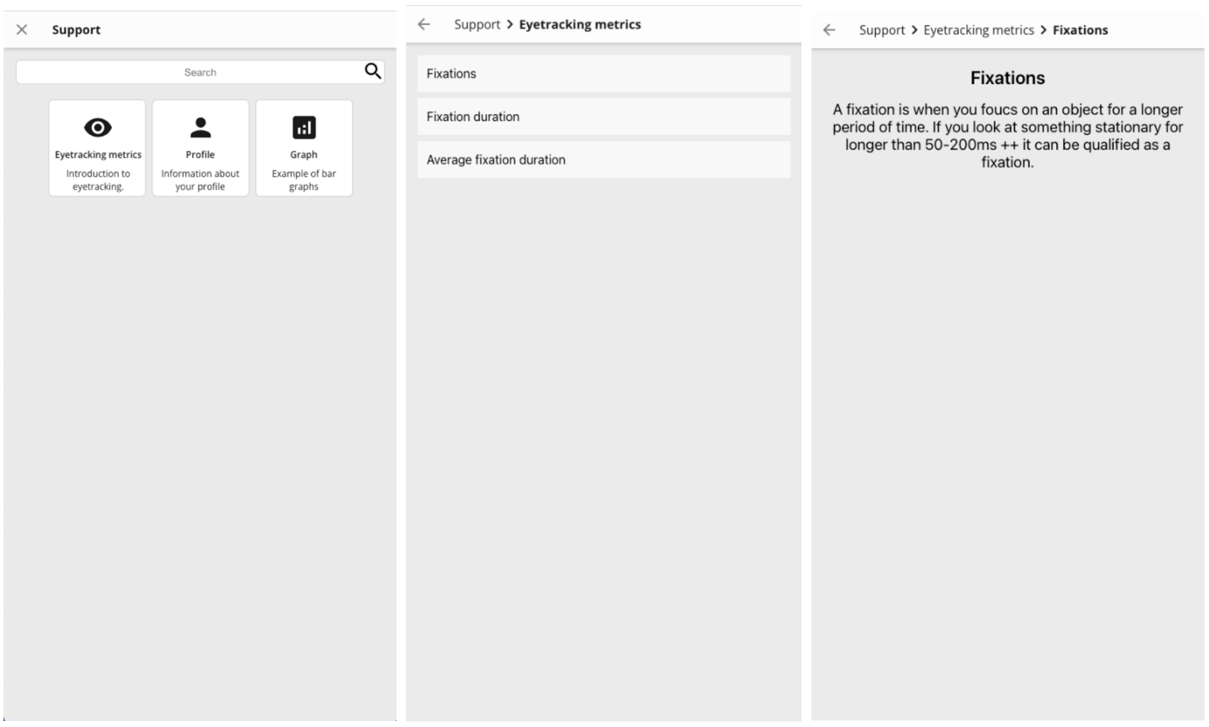
Figure 17 Shows the user tested sessions and their respective graphs.

4.4.6 Help and support

The help and support page are able to help the user understand the different parts of the webpage and how to use the webpage. The search bar allows the user to find exactly what they want. The buttons are divided into bigger categories, such as "Eye tracking metrics" and "Sessions". When clicking the button, the user will get an overview of subcategories belonging to each category. An example of the support pages can be seen in figure 18.

4.4.7 Error handling

The application has a way to alert the user to errors as well. The user gets a notification if they did something wrong or if sessions could not be loaded. This could potentially happen if the backend isn't up and running. Or if the username and password does not match any users in the database.

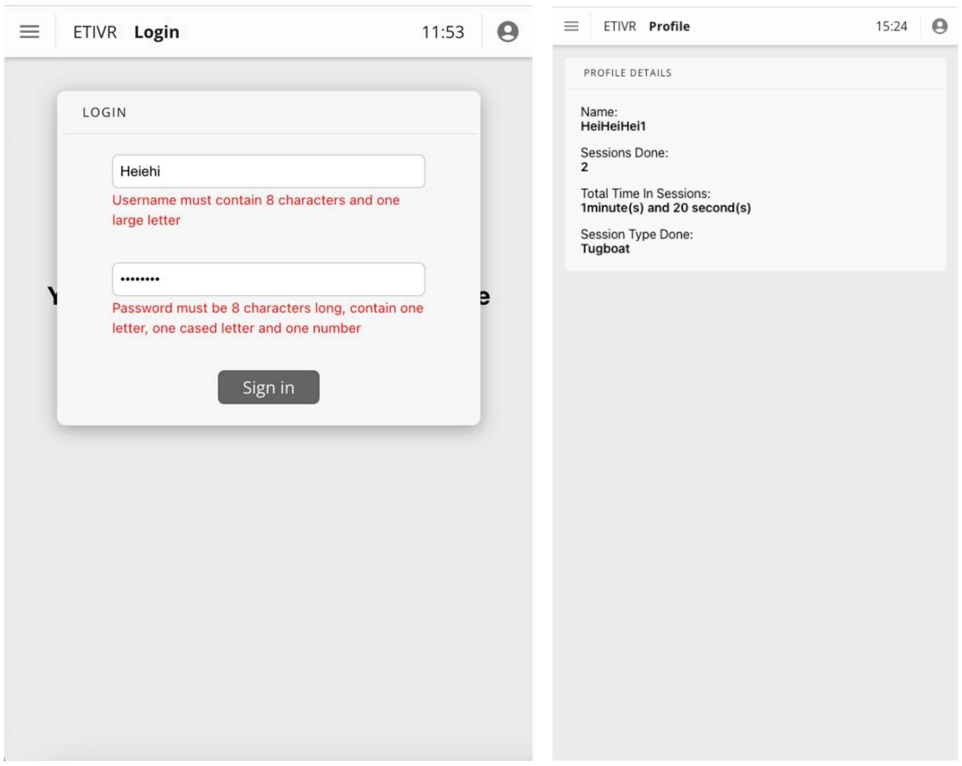


Support
Search
Eyetracking metrics
Introduction to eyetracking.
Profile
Information about your profile
Graph
Example of bar graphs
Support > Eyetracking metrics
Fixations
Fixation duration
Average fixation duration
Support > Eyetracking metrics > Fixations

Fixations

A fixation is when you foucs on an object for a longer period of time. If you look at something stationary for longer than 50-200ms ++ it can be qualified as a fixation.

Figure 18 Help and support page. An overview of definitions related to eye tracking metrics and definition of fixations.



Heiehi

Username must contain 8 characters and one large letter

••••••••

Password must be 8 characters long, contain one letter, one cased letter and one number

Name:
HeiHeiHei1

Sessions Done:
2

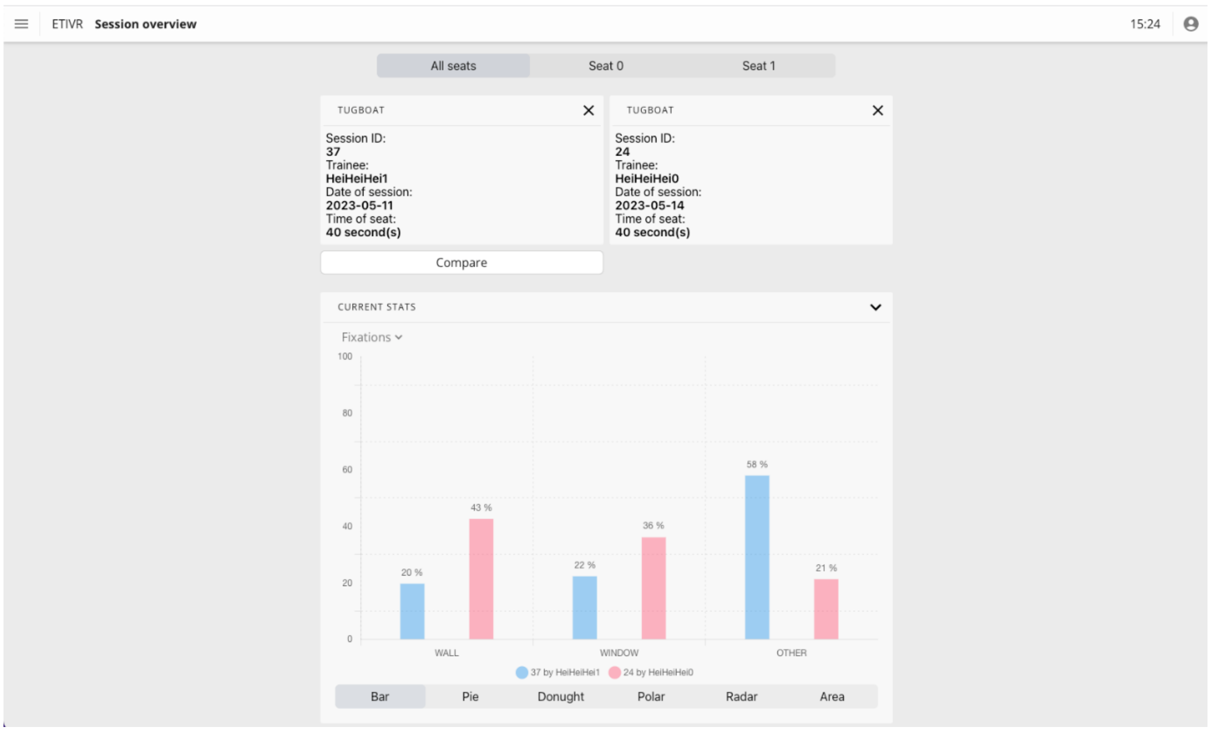
Total Time In Sessions:
1minute(s) and 20 second(s)

Session Type Done:
Tugboat

Figure 19 Error when signing into the account on the left and profile page on the right.

4.4.8 Application responsiveness

The application has the capability to function as both a mobile and desktop website due to its scalability. Consequently, the layout of the application is dynamically adjusted based on the device being utilized. This adaptive behaviour ensures that the user interface and design elements are optimally displayed and organized, irrespective of the device´s screen size or resolution. By automatically adjusting to different devices, the application offers an enhanced user experience and facilitates seamless interaction across different types of platforms.



Session ID:
37
Trainee:
HeiHeiHei1
Date of session:
2023-05-11
Time of seat:
40 second(s)

Session ID:
24
Trainee:
HeiHeiHei0
Date of session:
2023-05-14
Time of seat:
40 second(s)

Figure 20 Example of how the session overview page looks on a desktop.

4.4.9 The top bar

The top bar is always visible in the application. The upper section of the interface includes a navigation bar with a hamburger menu positioned on the left side, offering options to access the sessions, profile and help pages. It also displays the application name and indicates the current page. On the right side, there is a timestamp along with the options to navigate to the profile page and initiate the sign-out process.

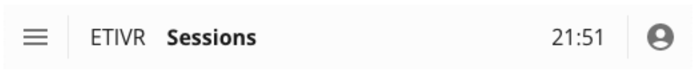


Figure 21 The top menu.

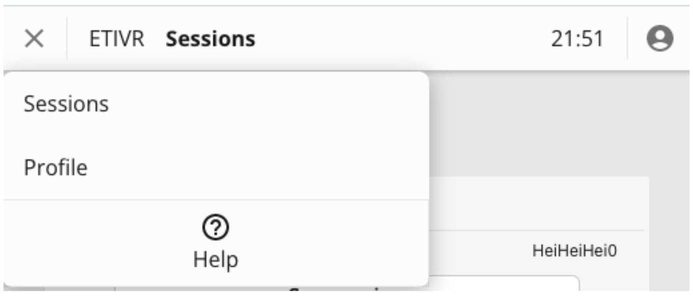


Figure 22 The main menu for the webpage.

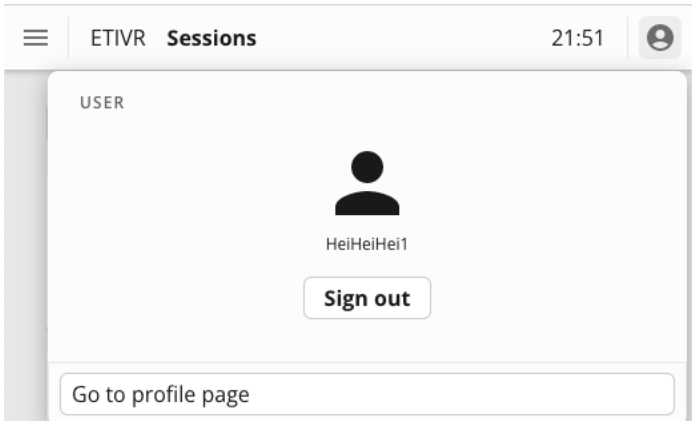


Figure 23 Profile card

4.5 Usability test

4.5.1 Demo adjacent issues

User testing can reveal problems with software which one might not encounter during development testing. User testing was conducted with at least one group member acting as a guide, and one taking notes of the user experience. With this method, several issues were discovered with the VR demo.

Some of these issues were not strictly VR demo related, but problems the testers encountered with the Meta Quest Pro. To get eye-tracking to work accurately, the HMD needs to be recalibrated when a new user tests the demo due to differences such as a user's eye distance. This consists of entering a developer menu in the Quest System Software and performing a calibration task. This task cannot be performed by anyone but the current tester, making it a required step before each demo.

One user experienced blurry visuals, which could have been caused by a lack of explanation of how the HMD needs to be adjusted vertically and horizontally to fit the user's eyes, in addition to an adjustment in the distance between the two lenses.

More issues with the HMD were found by user testing. One of these being the HMD and controllers automatically entering sleep mode. A user experienced problems with this when the controllers entered sleep mode during setup, requiring the user to re-awaken them. If the controllers were to sleep on demo start, the VR hands became an annoyance as they are initially placed at the same position as the user’s head, causing the hands to obscure vision.

4.5.2 Issues with the demo

At one point of testing, the demo was run through the Unity editor. Due to being an environment with variable performance load, the editor hanged and crashed before testing. This could likely be prevented by building a version of the demo that can be run without the editor.

An additional issue noted by testers is a performance drop when point clouds of a significant duration are loaded. There is a visible downturn in framerate when point clouds are displayed, likely caused the large number of particles being spawned.

At one point in the demo, users are asked to navigate a menu and select if boats of certain colours exist in the scene. Several of the testers failed to accurately describe some of the colours, especially on the black boat, which they assumed was some darker shade of grey.

Differing heights of users turned out to be mildly problematic. Shorter users could not see out of the windows of the bridge, but height quickly turned into a non-issue when the user teleported to any seat. Tall users should not be facing any problems.

Users needed a lot of guidance and following up to understand what they needed to do. There were multiple aspects of the demo that were unclear to users and required explanation from the group members. One such issues was opening and navigating the Settings/Tasks menu. One user didn’t notice that the menu could be accessed via the wristwatch on their left hand, or that the tasks menu could scroll to reveal additional tasks. Several of the tasks required additional explanation to how they could be completed.

Several bugs were found by using user testing. One bug is in the rendering of materials. While observing a tester, the material on the chair colliders turned into a bright cyan colour for a short time before turning into the intended transparent green colour. The same user spotted two boats repeatedly colliding with each other and getting stuck.

4.6 Collaboration tools

Jira and Con5fluence were used to ensure that Scrum was being followed. This includes tracking issues, logging time, writing different types of reports, and starting sprints. The name of the project is shortened into ETIVR, which stands for Eye tracking in Virtual Reality.

4.6.1 Jira

Jira is the software used to track issues, log time and start/end sprints. The issues assisted the other team members get an idea of how far the other group members had come and eventually help each other if needed. Jira has an issue board, where all the current issues were listed. This shows what issues had to be done, what issues were under development and what was finished.

Jira is mainly used to make sprints that lasted one week in the end and two weeks at the beginning. All the issues got added here and assigned to the group members when they were prioritized. The issues also had subtasks that had to be done for the issue to be done. Before making issues, the team made different types of user stories that explained what the user should be able to do, from a user's point of view. A description is also added that explains what needs to be done for the task to be marked "done".

The team was responsible to log their time. This helped the team get an overview of what the other team members were doing, and what tasks they used the most time on. All the meetings were also logged. The time gets listed under time sheets, and the whole team have access to it.

4.6.2 Confluence

Confluence was used to write different types of documents throughout the project process. This includes documents like meeting agendas, meeting notes, agreements, and retrospectives. This team workspace assisted the team with keeping track of documents and making documents more efficient.

Meeting agendas were written before every meeting so that the people invited could get an idea of what the meeting was going to be about. These agendas included questions, concerns, and things we had to find out during the meeting.

Meeting notes were written after each meeting. These notes had an overview of where and when the meeting was held, who attended, whether there was any absence, the type of meeting, the purpose of the meeting and what was done during the meeting.

The agreement was written at the beginning of the project. The agreement included the goals, roles and division of tasks, procedures, and interaction. The members of the team had to sign this contract to be a part of the project.

Retrospectives were written after each sprint was done. This made it easier to look back on what was done in the sprint. The team reasoned about what went well and what should have been better during the sprint. If some members had things they could improve, then they would be called out here.

Confluence was also used to make decisions within the group. This function assisted the team with being more democratic and the group members having the ability to vote for what they want.

4.5.3 GitHub

The coding platform, GitHub was used for version control and collaboration. The team made an organization for the project and divided the project into three different repositories. One for the Unity project, one for the backend and one for the frontend. This made it easier for the members to work on the different parts of the project without being dependent on each other. Some parts of the project had a lot of storage as well. Having separate repositories helped avoid unnecessary storage use.

Skorpen and Midthus had the responsibility of creating the Unity project. This project had everything from assets to C# code logic for the Unity model. They worked together during the development process and used the Unity repository. This gave both tracks of who committed and pushed what to the repository.

Brevik and Ahmadi had the responsibility of creating the frontend and backend. They worked on the backend repository first and then began on the frontend repository later. Midthus assisted them throughout the project if it was needed. They worked on different branches and merged their branches into the main branch afterwards.

5. Discussion

This chapter will be discussing the results and reflecting the work that has been done. Any challenges and problems during the development process will be discussed here.

5.1 Implemented functions

The following chapters discuss what requirements we have met according to the requirements of the project owner.

5.1.1 Virtual Reality demo

Gather eye tracking data based on each category of objects.

One of the requirements was to gather eye-tracking data based on each category of objects. This has been accomplished by looking at the different metrics for an object. Each object has a defined category that can be used to sort the performance based on their total metrics for that category instead of only showing it on a per object basis. This is done since the performance per object basis is not interesting and could possibly confuse the user.

Visualizations of different data in Unity

The visualization methods that are implemented is the point cloud and the points of interest. These models show both where the user has been looking, but they do not display the metrics per trackable object. Therefore, in a sense the data has been visualized but only the raw hit positions. The rest of the metrics per object is still not visualized in the VR demo and could be displayed by using the object-based colour coding. The reason that we didn’t implement this was due to the time constraint. Some of the logic was done, but for a colour to change the RGB values had to be altered.

Recording the session was another visualization that could have been implemented and was planned. But the recorder that unity supports can only be used when the game is run in the editor. This limits the usage of it since it cannot be used when the application is compiled and published in a later demo. Therefore, recording was dropped as a possibility.

Have a situation that can be done by a user in the demo.

The demo itself includes a set of tasks that must be done to “finish” their session. These are included so that the group could gather data for the final report. The tasks must either be configured in the task manager or placed directly on the objects themselves.

Interaction with menus and objects

Menu and object interactions were successfully implemented through rays being cast out of the hands of the user. The user can use these rays as pointers to navigate between seats, pick up distant objects, and as a cursor in menus. The hands are also rigged to correspond to trigger input from the controllers, where the hand blends from an open palm to a closed fist or pinch according to how far the triggers are pushed.

Getting the hands from the Oculus SDK to work with this project proved to be a challenge. The controllers of these hands were based on an old input system in Unity, in addition to being focused on Oculus Touch controllers specifically. For this reason, they were incompatible with the new input system in our version, as well as our early goal of supporting multiple controllers and HMDs. The code had to be converted to adhere to the new input system as well as being controller agnostic. The OVR hand rig solution contained input and hand poses for pointing, thumbs up, hand open, hand closed, pinching and closed fists. Due to issues in conversion, only open hand, pinch, and fists ended up being used in the final demo.

The group initially intended to have touch screen interactions in the VR demo. This could be intuitive and convenient for users familiar with touch screen input, instead of using the specific buttons and pointers of the controllers. This could be a good solution in an application where movement is free, but with the fixed teleport locations of the demo, one could quickly find oneself just out of range of an interactable object or menu. An additional problem was that the hands would need a pointer pose, which was lost in the conversion. It would also require more set up, as a component we were already using for hand interactions came with the ray cast interactable included. For these reasons the group decided to go with the ray casted pointers for all interactions in the final delivery.

Easy to setup the logic in a new scene

To simplify the setup of the eye tracking system the trackable objects and referecnce positions are automatically added if they are descendants of the simulation setup controller. This is done by using the broadcasting system and send the simulation setup manager as a parameter where the trackable objects and reference positions uses to add themselves to the manager. Then the simulation setup manager adds these objects to the wanted simulation setup controller. The only problem with this setup is that the manager only supports one simulation setup controller at the time and is an unnecessary step since the simulation controller could do the adding itself. Thus, saving performance and a little code time.

The setup of the menus, tasks and objects that can be picked up are another matter. There is no automatic setup for this since they are using event systems and a lot of logic must be setup to get the systems to work. The menus and pickup able items are to a certain degree automatic but can be hard to debug if something does not work. Colliders can also affect where the "centre" of the object is resulting in weird rotations.

Especially when it comes to the tasks since the system is complex and requires triggers from GUI or other interactions. Some tasks are easily made like the pickup and gaze tasks in the task manager with the configuration of tasks but can also be frustrating for the user if they must hold or look at an object for a continues time.

Adaptive training system implementation

The feedbacks are originally designed to get the different feedback for the current position of the user and display these to the user as an adaptive feedback system. This system would show the different percentages per category to the user live when the feedback is calculated. The feedback was then displayed on a panel that would appear when new feedback was calculated.

This system was not further developed since the percentages that was displayed was distracting the developers during testing. That is why the system was not used during the user tests since the panel itself might distract the user during the data gathering process. The panel would also be counted as its own "category" if the eye tracking was not disabled and effecting the resulting metrics of the session.

A solution to this problem would be to pause the eye tracking when the feedback was displayed to the user, but this would also catch their attention and maybe effect their performance since the world is not also paused. Another solution is to give the feedback through verbal means with a computer-generated voice. This would be less distracting than a physical panel but are harder to implement.

5.1.2 Backend

Persist the data on a database for further use.

The approach chosen is to make an object-relational database on a MySQL server. The model of this data was not specified, which lead to a lot of changes in the database during the project development process.

Tests were implemented to verify the database's validity and to access and persist data. These tests make sure that the data gets modified, added, and removed correctly. All these tests need to pass for the application to run. However, due to autogenerating the tables, the potential for encountering additional issues and bugs might arise. These issues can create problems such as invalid storage of data fields that will result in the field being set to null when pulled out of the database. This is largely avoided with the tests since exceptions are thrown if the data is not persisted correctly.

To avoid problems like this, the use of Cascade type has been necessary in the joining tags. Cascade type enable the systematic handling of the object itself and their associated fields to be inserted into the database. By implementing Cascade type, the system aims to ensure the integrity and consistency of the database by properly managing the relationships between data entities. This approach minimizes the probability of data inconsistencies or misplacements when deleting objects. Thereby enhancing the overall reliability and functionality of the application.

Access the data through endpoints.

The API consists of sending the data like sessions, users, and simulation setup through the designated endpoints. The approach covers the mechanism of accessing data through endpoints by enabling the exchange of information between the VR demo, client, and the server.

User authentication is necessary to ensure data privacy and security, restricting access to authorized users only. It is possible that the requirement for users to be logged in to access their data is influenced by data privacy regulations such as GDPR. Compliance with GDPR may necessitate implementing measures to ensure that only users have access to their personal data.

Even though authentication is not explicitly mentioned, it is commonly understood as an essential security measure to prevent unauthorized access to personal data. By implementing user authentication, the application owners can control and verify the identity of individuals seeking access to personal data, thereby reducing the risk of unauthorized or unlawful processing.

Persist different sessions done by different users and make them comparable.

The sessions that are done by a user is identified by their own user, current date, and session length. This implementation ensures that the user themselves can find their own sessions. It also makes it possible to compare the user's performance against others or their own older performances if the sessions simulation setup is the same.

The reason that the simulation setup needs to match for the sessions to be compared is that the data is recorded on a per position basis. If the simulation setups mismatch the number of positions might differ and thus the performance per seat cannot be calculated and compared. The total metrics could be compared between sessions with different simulation setups but could result into non-productive feedback since the optimal viewing time per position per simulation setup may differ.

5.1.3 Frontend

Compare your session against an expert.

The implementation of sessions allows us to compare any user's performance against any other user's performance if they are the same simulation setup. The compared sessions can also be the same simulation setup but different sessions for the user and be compared against each other.

One problem with the implementation is that there is no way of knowing what experience level the users have. There is no tagging that allows the webpage to identify or sort the sessions based on the users experience level or rank. A solution to this problem is to remember the usernames of the experts and compare against their sessions. Even that is not an optimal solution and there should be a more user-friendly way of achieving this without having to remember multiple usernames. A tag could also be included on the user profile that ranks the user based on predefined experience set by the administrator, and then the sessions can be sorted based on experience level of the attending user. But experience per simulation setup might also wary so the ranking system should include that into its calculations.

See a visualization of the data in some form.

The data of a session is visualized in two different graphs. The current stats graph and the feedback timeframes. The difference is that the current stats have more options to look at more metrics, but the feedback timeframes only show the fixation duration per category.

If the current stats show the feedback duration both the last result of the feedback timeframe and current stats graphs should be identical. But unfortunately, they are not. The reason for that they are different is that the remaining time of the current stats are not added to the other category like the feedback class. This means that the total amount of fixation duration does not match the time for that seat, resulting in a graph that is not the same. A simple fix to this problem is to take the remaining time the when the session is finished and add it to the other category. But due to time limitations this is not done.

Even though the user can change between different metrics on the current stats graph, it might also be confusing. Since the fixation duration is the main metric that the project owner was interested in. But it included so that the number of fixations could later be used to analyze if the user was confused. The feedback could also be altered later to show a time from 30 to 40 seconds since the first feedback can be subtracted the last feedback to get that period. And this would show easily how the user was confused in that timespan with the different metrics.

Optimized for both desktop and mobile

It is optimized for desktop and mobile to a certain degree. Some of the layout is not user-friendly for a mobile user. The buttons are too small for the text, the graph data labels can be pushed too close to each other when comparing a lot of different sessions. And the percentages of the graphs overlap when there are many sessions that is compared. All of these decreases the mobile user experience and needs to be fixed in the future. The reason that this was not done is the time constraint of the project, since the bugs was found when the report was being written.

Follow the OpenBridge system.

The website follows most of the guidelines that the OpenBridge design system has set. But there are a few things that are not properly fixed.

An example of this is when the webpage changes to dark mode. All the components that OpenBridge has made changes their theme, but the additional components that we had to make does not. Therefore, there are some optimalisations that are missing for the webpage when it comes to dark mode. This could easily be fixed if the group had more time.

Other factors like the design are still quite close to the first draft of the webpage. But some changes were made for the user to be able to change graphs. These changes and the final implementation have not received the same design review as the figma prototype because of time constraint. But probably should have, if the group had more time the webpage would also get a design review.

Live sessions on the webpage

This was not strictly a requirement, but the project owner wanted it as a "last function". Because of the time limit we could not fulfill their wish, but most of the logic is there to make this happen.

5.2 Limitations

The Meta Quest pro itself was chosen for this project because of its more open approach to licensing and ease of use. The Omnicept on the other hand was difficult to get the license for both its native SDK and the Tobii XR library. But even though the Omnicept was not used it has some features that the Quest pro lacks.

One of these features is pupillometry. This allows the pupil size of the user to be recorded when they are looking at different objects. This feature would be especially useful to add to the adaptive training system, since it's a good indication of stress and fatigue. The adaptive training system could then be used to adjust the environment so that the user does not experience visual or mental fatigue. A report could then be generated every minute to log and check how the cognitive load on the user is being affected. Combine that with the heartrate feature of the Omnicept, and it is even easier to log and check when the user is stressed or overloaded by the simulation.

Even though these features are impressive for their use they could also have their downsides. Since with the gaze pattern, pupillometry and heartrate could say a lot about someone's health or conditions that are supposed to be private. So, the data itself had to be stored in a secure manner to not leak this particularly sensitive information.

5.3 Integration

5.1.1 VR

Since the project is made in unity the integration of this project must be done there. The eye tracking logic that has been made during the project could be easily extended by another headset in some cases. The main logic with the objects themselves tracking their time is easily extended, since only a Boolean must be changed, and a method needs to be called on the object.

The recorded points and points of interest are harder to be implemented by other headsets since it relies on raw gaze position and hit position. Other headsets might call methods on the objects themselves and not revealing their hit position. Thus, the integration of this mechanic to other headsets might be difficult.

The logic itself is easily extended since all the code is documented and different design patterns are used to ensure easier extendibility.

5.2.2 Backend

Using the backend and integrating other systems is also quite developer friendly. This is due to the implementation of the register classes where the final user easily can change to another database implementation. The user is also a general class that can be changed easily to adapt better to other login systems. There is no interface for the users though, so it needs some work.

Endpoints themselves are also accessible and documented well. This makes it easier to get the data form the server to other applications. The only problem with the endpoints is that there is no service like Swagger to show the documentation on a webpage. The user needs to get access to the source code or the Javadoc to know what endpoints are accessible with what permissions.

5.4 Deployment

The backend, database and frontend are hosted on the servers provided by the institute. This way the user data that was collected during the testing can be accessed afterwards and displayed in this report.

Backend

To deploy the backend, one needs to either have an editor or docker installed. Docker is used to run both the backend itself and the database in separate containers and simplifies the setup of the system. This secures the database against a fatal crash on the backend, and maintenance can be done easier since only the part that needs maintenance is taken down.

If the docker container is deployed the user only needs to run the prompts that are included inside the readme file for the project. The only thing that needs to be done before deployment of the backend is to fill out the password and username of the database in the docker compose file.

Frontend

The frontend is also deployed by using docker but needs to be deployed separate from the backend and database. The instructions to deploy the webpage is included in the readme file of the GitHub repository. The distribution service like Nginx needs to be routed or decided by the user themselves.

VR demo

To run and test the VR demo a Unity editor is needed. Since the demo itself is not ready to be compiled and ran by any users. This is done since some login information and settings in different controllers also needs to be set to the correct IP address and port to store data on the backend.

5.5 User Testing

Conducting tests on new users provided valuable insights into improvements that could be made in the project that the developers might miss. Not having seen the demo before, the testers allow developers to understand how a new user experiences the application. By conducting user tests, the group spotted several specific instances where improvements could be made.

One improvement could be in the initial testing start up process. Users reacted to the amount of setup and guidance needed before getting to the demo. The amount of guidance could be reduced by building the demo to work with the Meta Quest Pro hardware itself. This would remove the need to establish a link between the computer and the Quest, which would reduce the initial tasks to eye calibration, and starting the demo application. This could have other problems, however. The demo is intended to be set up similarly to the system the stakeholders are currently using, which is a laptop and Quest Pro set up with Oculus Link. Building exclusively to the Quest would differ from this set up. It could also cause performance issues, as the Quest Pro is significantly weaker than the current target laptop hardware.

Another improvement could be with the visuals of the boats outside of the bridge. Users had difficulties with guessing the correct colours on the different boats. Several factors contribute to this issue, such as fog, simple material properties, and low model polycount. Steps to mitigate this colour issue could be to increase the polycount of the boat models to increase reflection accuracy, increase contrast between colours to make colour choice more obvious, and remove misleading colour choices in the multiple-choice menu. But at the same time the tests must contain some harder tasks so it’s not easy for the users. Since in a normal boat environment it might be challenging to identify the boats and their details.

Boats were also difficult to keep track of when they were moving around. Making the boats more unique from each other could help as they are currently sharing the same model with different materials. Their colliders could also increase so that they are less likely to crash into each other.

The point cloud setting creates 50 spheres per second where the user has been looking. The significant performance hit when point clouds are enabled could be reduced by decimating the triangle count of the sphere being used. The default sphere in Unity has a triangle count of 768 triangles. This sphere could be replaced by a simpler sphere with 130 triangles. Visual difference would be negligible, because of the small size of the individual points of the cloud. Another solution is to display less number of points at the same time and have a timeframe that the point cloud is displayed for.

Differing user height was one issue, but it was only a problem before the user teleported somewhere and the height got normalized. A fix for the initial height could be to teleport the user somewhere in the start of the demo. This is because the headset starts at a certain position that is not the height of the user.

The settings or tasks in-world menu had several user experience issues that could be improved. Users didn’t understand intuitively that both the tasks and settings lists could be scrolled through by pressing the trigger and dragging with the pointer on the background. A potential improvement for this could be to add a scroll bar, allowing the user one more way to scroll, in addition to conveying that these areas are in fact scrollable.

Opening and closing this menu was not intuitive to users. When the menu is closed, the users do not see that the menu “travels” into the left-hand wristwatch. A clearer way to convey this could be to have a trail show up during the travel time. This would show a path to the hand which could cause the user to investigate further. A task could also be assigned to open the menu from the left hand; thus the user instantly knows where the menu is.

The hit registration of in-world UI elements proved to be inconsistent. One tester reported problems with their hit registering on a button they tried to press for instance. This problem is likely due to careless use of borders in the Unity UI system. The borders define what counts as a "hit" in the UI. Hit registration could be improved if the UI was revisited and each element received a more accurate border and larger physical size.

When users are almost finished with their assigned tasks, they get tasked with sending their session to the server. This is done in several steps. Accessing the settings menu and unchecking the eye tracking setting reveals a button marked "send session". Pressing this button sends the session and completes the task. These steps could be more intuitive if the "send session" button was always visible in the menu but greyed out until eye tracking has been stopped.

When users managed to send their data, they were quick to send it several times on accident. They were uncertain if the data was sent or not. To reduce redundant data and server strain, a dialog asking the user if they are certain they want to re-send data could help. There should also be some sort of confirmation indicating if the server has received the data or not. A dialog should also appear when the user has sent their session so that they know the interaction has happened.

5.6 Development process

Using Jira as an issue & project software

The group initially did not use Jira and Confluence for documenting their development process. However, upon receiving recommendation from one of the lecturers, the group started using Jira to track issues, assign tasks to members, and manage sprints. The group began by creating user stories and corresponding issues, which were then assigned to individual members. Confluence was used for retrospectives and meeting notes.

Sprints

The sprints lasted two weeks in the beginning, since the group had another subject in the beginning of the semester. After that subject was done in March, the group began having sprints that lasted one week. Even though the sprints had a defined start and end they were sometimes over time. This was due to most of the members being busy at the retrospective day, and sometimes it was forgotten. To combat this the group should have been stricter with following Scrum and not adjusted it accordingly to the group members when collisions happened. This could have been handled better by having the meetings, even though it only fit for some members.

Many sprints ended with a status update to the client and supervisor either with a meeting and/or with video. During these updates, the group received feedback from the project owner and supervisor on what problems or tools we could use. The project owner then often gave feedback on what to focus on when the group gave options for future focus fields.

Sprint reviews

The meetings with supervisor and project owner helped to keep track of the project and its progress. Meeting notes were written during these meetings and demo videos of how far the project had progressed was done, but like sprint reviews. The videos were made when the project had made visible progress and was done not every week but many of them. Each agenda had status reports about the project, so it looks like sprint reviews.

Videos was done because of the visual part of the project. The project had a lot of things that was hard to describe without having a video showing it. Eye tracking is not that easy to show without a demo. The quality of the videos could have been better, but the field of view differs from the headset to the unity editor. So, the poor field of view in the unity editor limits what the recording can see.

Logging time

The group faced challenges initially as they were inexperienced with the Jira software and its features. This resulted in incomplete time logging during the early stages of the project. Also, a sprint of the project was done before the group got access to Jira. This means that the total time of all the group members should have 10-20 hours of work per person.

Initially, the use of Jira and individual responsibility for managing issues and logging time was effective. However, as the project progressed, some group members tended to forget to log their time, leading to inefficacies and incomplete log of the time. Furthermore, the development process faced delays in the beginning due to another subject that required more time than anticipated. As a result, the group had only two days a week to work on their bachelor project.

Some of the group members also had more subjects, which lead to using less time for the bachelor project. A part of the deal was that these subjects would not affect the bachelor project a too much. During the project, the group had events they had to attend to with their families. This made it even more challenging to gather the group members and do the work needed.

Using Confluence as a team workspace

Confluence was used in the beginning after a lecturer recommended us to use it. Later in the projects process teams was used more when the report grew, but retrospectives and meeting notes was still stored on Confluence. This made it more structured and efficient when writing the report documents. The supervisor also had access to Confluence.

Issue scope and epics

Some of the issues should have a better-defined scope. The reason being, that some of them had a bigger scope, making it harder to do. The same can be said about epics that should have been broken down into smaller parts, making it easier to sort the issues and make new issues. There were six major epics, but if these were divided into smaller fragments, it would be easier to make tasks and user stories for them. Since they are not covering such large topics.

6. Conclusion

6.1 Conclusion

6.1.1 Teams conclusion

The complete solution of the bachelor thesis is solved using unity, spring and react. The main requirement from the client was solved by making us able to compare two sessions against each other. However, there is no easy way of knowing who’s an expert without remembering their username. Regardless, the goal was achieved in some matter. Some of the requirements was also defined during the project, thus the scope grew big and could have been smaller to reduce the number of unmet requirements.

In the webpage the performance from one user can be compared against another with graphs and charts. The webpage additionally offers the user the ability to create a user, login, change graphs and see their sessions. The users themselves are not integrated into any predefined system. Therefore, this project is easy to setup and try out.

However, there are more testing that should have been done. That is because the group should confirm that it is beneficial and informative for the potential users. The graphs were shown after a test, except there were no real user tests on the webpage. Furthermore, the tests that was done with the VR demo should have been done with more than three users.

The team is quite satisfied with the execution of the scrum. Despite the sprint reviews that was not done properly throughout the project, and some sprints exceeding their time. The usage of Jira and confluence as tasks tracker and documentation manager was also a good experience that showed how real projects are run. Except from the learning curve that was needed to learn these tools during the project's execution. Also, the logging of time on tasks for the group could have been handled better.

6.1.2 Clients conclusion

It seems like the client was happy with our exploration of eye tracking in VR. As stated below:

The project has been informative for the Norwegian Costal Administration. It has shown the possibilities and limitations of eye tracking in VR, at the same time it has also shown us how fast the technological development is in this domain. It also shows the possibilities within combining different technologies.

6.2 Further work

6.2.1 VR Demo

Given more time to work on the project, there are several improvements to be made. Regarding the VR demo, one of these improvements could be better interactivity. Currently, the interactable tasks are primitive in comparison to actual VR training scenarios. The user should have interactable levers and buttons that can control the ship the user is in. This could provide more valuable data than simpler tasks.

An additional improvement could be in visualization within the demo. A shader showing a heatmap of where the user has seen would be an intuitive way for others to understand the user's gaze pattern without looking at graphs. Other models like perception map and object-based-colour coding could also be added.

6.2.2 Backend

The backend could also get improvements. One improvement that could be made is converting the database from JPA to a relational database. This could provide faster storage speeds and allow for the storage of more raw data. More raw data storage provides more freedom to the users, allowing them to perform their own detailed analysis of an eye-tracking session. Each endpoint could have been documented with Swagger as well. The backend could have stored a point record. More visualization methods in VR could have been implemented.

6.2.3 Frontend

The frontend could use some improvements. If the group had more time, dark mode implementation could have been done. The ability to choose between night, dusk, day, and so on could also make the functionality of the website more personalized for the users. Another implementation could have been alerts on the navigation bar when a new session is uploaded. This would give the user more feedback. The website could also be less clunky since it can be perceived as kind of messy. The token could have been automatically refreshed, to have a proper user experience. Another function that could be implemented is the option to filter based on if the user is an expert or not. This would give the user an example of how good the results can be.

Societal Impact

Ethical concerns

Investigating and developing virtual reality eye tracking applications can raise concerns regarding the ethics of using this collected data. This is particularly relevant in this projects case, using a VR headset created by Meta, who also owns Facebook. Facebook keeps extensive track of user information, creating profiles and grouping persons by different marketability standards (Dewey 2016). From this, one could extrapolate and raise concerns regarding the data retrieved by the Meta Quest Pro, which can track eye movements and recognize room arrangements.

In our project, the tracked information is stored privately in a database requiring special access and is not used for marketing or sold in any way. There is still reason for concern however, as our development in these avenues can raise interest in potential breaches of privacy from other parties.

Sustainable goals

Development of this project can aid in the following UN sustainable development goals:

Goal 8: Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all

Through active development of maritime VR training, these types of training simulators can become more accessible. This can provide interested parties without access to real-life training a way to practice, get certifications, and start working at actual vessels (United Nations n.d.).

Goal 14: Conserve and sustainably use the oceans, seas, and marine resources for sustainable development.

Due to potential improvements in maritime training from emerging eye tracking technology, more training could be performed in simulators instead of at-sea. Reducing the amount of training needed with actual ships has potential of decreasing total emissions from seafaring vessels. Reduced CO_2 emissions reduces the acidification rate of the oceans, which can aid in the survival of sea-life (United Nations n.d.).

References

Atlassian (n.d.). "Confluence basics." Retrieved 01.05.23, 2023, from https://www.atlassian.com/software/confluence/resources/guides/get-started/overview#about-confluence.

Atlassian (n.d.). "What is the Agile methodology?". Retrieved 18.05.23, 2023, from https://www.atlassian.com/agile.

baeldung (2022). "Learn JPA & Hibernate." Retrieved 01.05.23, 2023, from https://www.baeldung.com/learn-jpa-hibernate.

BillWagner, et al. (2023). "A tour of the C# language." Retrieved 20.05, 2023, from https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/.

Blender Freedom (n.d.). "The Freedom to Create." Retrieved 28.04.23, 2023, from https://www.blender.org/about/.

Checkstyle (2023). "CheckStyle." Retrieved 18.05.23, 2023, from https://checkstyle.sourceforge.io/.

Chehab, G. (2023). "What is an ORM? How Does It Work? How Should We Use One?". Retrieved 18.05.23, 2023, from https://www.baeldung.com/cs/object-relational-mapping.

DBeaver (n.d.). "About." Retrieved 28.04.23, 2023, from https://dbeaver.io/about/.

Dewey, C. (2016). "98 personal data points that Facebook uses to target ads to you." Retrieved 21.05, 2023, from https://www.washingtonpost.com/news/the-intersect/wp/2016/08/19/98-personal-data-points-that-facebook-uses-to-target-ads-to-you/.

educative (n.d.). "What are Norman´s design principles?". Retrieved 09.05.23, 2023, from https://www.educative.io/answers/what-are-normans-design-principles.

Freeman, E., et al. (2021). Head First Design Patterns, 2nd Edition. United States, O´Reilly Media,.

GitHub Docs (n.d.). "Hello World." Retrieved 01.05.23, 2023, from https://docs.github.com/en/get-started/quickstart/hello-world.

GitLab (n.d.). "What is a code review?". Retrieved 18.05.23, 2023, from https://about.gitlab.com/topics/version-control/what-is-code-review/.

GitLab (n.d.). "What is distributed version control system?". Retrieved 20.05, 2023, from https://about.gitlab.com/topics/version-control/benefits-distributed-version-control-system/.

Hargrave, B. (2023). "Object- and function-oriented programming concepts and principles." Retrieved 20.05, 2023, from https://developer.ibm.com/tutorials/oo-v-functional-programming/.

IBM (n.d.). "What is Java?". Retrieved 01.05.23, 2023, from https://www.ibm.com/topics/java.

Interaction Design Foundation (n.d.). "Interaction Design." Retrieved 20.05, 2023, from https://www.interaction-design.org/literature/topics/interaction-design.

Intersoft consulting (n.d.). "Security of processing." Retrieved 20.05, 2023, from https://gdpr-info.eu/art-32-gdpr/.

javaTpoint (n.d.). "Hibernate Tutorial." Retrieved 20.05, 2023, from https://www.javatpoint.com/hibernate-tutorial.

Josikakar (n.d.). "Software Engineering | Coupling and Cohesion." Retrieved 20.05, 2023, from https://www.geeksforgeeks.org/software-engineering-coupling-and-cohesion/.

Khronos (n.d.). "Unifying Reality." Retrieved 18.04.23, 2023, from https://www.khronos.org/api/index_2017/openxr.

Marschner, S. and P. Shirley (2015). Fundamentals of Computer Graphics 4th Edition, A K Peters/ CRC Press.

MDN contributors, I. (2023). "Object-oriented programming." Retrieved 01.05.23, 2023, from https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/Object-oriented_programming.

Meta (n.d.). "Buy Meta Quest 2. Get two hit games.". Retrieved 20.05, 2023, from https://www.meta.com/no/en/quest/products/quest-2/tech-specs/#tech-specs.

Microsoft (n.d.). "What is Microsoft Teams?". Retrieved 01.05.23, 2023, from https://support.microsoft.com/en-us/topic/what-is-microsoft-teams-3de4d369-0167-8def-b93b-0eb5286d7a29.

Midthus, S. H. and T. Staverløkk (2022). State-of-the-art report on eye-tracking in Virtual reality. Ålesund, NTNU (Norwegian University of Science and Technology).

Okta (n.d.). "Authentication vs. Authorization." Retrieved 18.05.23, 2023, from https://auth0.com/docs/get-started/identity-fundamentals/authentication-and-authorization.

Okta (n.d.). "Token Based Authentication." Retrieved 01.05.23, 2023, from https://auth0.com/learn/token-based-authentication-made-easy.

OmniConvert (2023). "Why is user testing important?". Retrieved 20.05, 2023, from https://www.omniconvert.com/what-is/user-testing/.

OnkarRuikar (2023). "What is JavaScript?". Retrieved 18.05.23, 2023, from https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript.

OpenBridge (2020). "Terms of use." Retrieved 08.05.23, 2023, from https://www.openbridge.no/home/terms-of-use.

OpenBridge (n.d.). "Application." Retrieved 20.05, 2023, from https://www.openbridge.no/pattern/application.

OpenBridge (n.d.). "Cards." Retrieved 20.05, 2023, from https://www.openbridge.no/pattern/cards.

OpenBridge (n.d.). "Colors." Retrieved 20.05, 2023, from https://www.openbridge.no/guidelines/palette/colors.

OpenBridge (n.d.). "Help & Support." Retrieved 20.05, 2023, from https://www.openbridge.no/pattern/support.

OpenBridge (n.d.). "Styles." Retrieved 20.05, 2023, from https://www.openbridge.no/guidelines/palette/styles-and-states.

OpenBridge (n.d.). "Typography." Retrieved 20.05, 2023, from https://www.openbridge.no/guidelines/palette/typography.

pp_pankaj (2023). "Unit Testing | Software Testing." Retrieved 18.05.23, 2023, from https://www.geeksforgeeks.org/unit-testing-software-testing/.

Prettier (n.d.). "Why Prettier?". Retrieved 08.05.23, 2023, from https://prettier.io/docs/en/why-prettier.html.

ProductPlan (n.d.). "Jira." Retrieved 01.05.23, 2023, from https://www.productplan.com/glossary/jira/.

Qualcomm (2022). "AR, VR, MR, and XR - what they mean and how they´ll transform lives." Retrieved 20.05, 2023, from https://www.qualcomm.com/news/onq/2022/09/ar--vr--mr--and-xr---what-they-mean-and-how-they-ll-transform-li.

Radigan, D. (n.d.). "An agile guide to scrum meetings." Retrieved 18.05.23, 2023, from https://www.atlassian.com/agile/scrum/ceremonies.

Ramsøy, C. (2022). "En kort introduksjon til Scrum." Retrieved 20.05, 2023, from https://www.visma.no/blogg/en-kort-introduksjon-til-scrum/.

React (n.d.). "React." Retrieved 20.05, 2023, from https://legacy.reactjs.org/.

Refactoring Guru (n.d.). "Builder." Retrieved 20.05, 2023, from https://refactoring.guru/design-patterns/builder.

Sirois, S. and J. Brisson (2014). "Pupillometry." Retrieved 20.05, 2023, from https://wires.onlinelibrary.wiley.com/doi/full/10.1002/wcs.1323.

SonarSource (n.d.). "SonarLint." Retrieved 18.05.23, 2023, from https://marketplace.visualstudio.com/items?itemName=SonarSource.sonarlint-vscode.

Source Making (n.d.). "Design Patterns." Retrieved 20.05, 2023, from https://sourcemaking.com/design_patterns.

spring (n.d.). "Spring Security." Retrieved 01.05.23, 2023, from https://docs.spring.io/spring-security/reference/index.html.

spring by VMware Tanzu (n.d.). "Building REST services with Spring." Retrieved 20.05, 2023, from https://spring.io/guides/tutorials/rest/.

Srivastav, P. (n.d.). "Docker for beginners." Retrieved 2023, 20.05, from https://docker-curriculum.com/.

United Nations (n.d.). "Conserve and sustainably use the oceans, seas and marine resources for sustainable development." Retrieved 21.05, 2023, from https://sdgs.un.org/goals/goal14.

United Nations (n.d.). "Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all." Retrieved 21.05, 2023, from https://sdgs.un.org/goals/goal8.

Unity (n.d.). "About Shader Graph." Retrieved 20.05, 2023, from https://docs.unity3d.com/Packages/com.unity.shadergraph@16.0/manual/index.html.

Unity (n.d.). "ECS concepts." Retrieved 18.05.23, 2023, from https://docs.unity3d.com/Packages/com.unity.entities@0.1/manual/ecs_core.html.

Unity (n.d.). "Visual Effect Graph." Retrieved 20.05, 2023, from https://docs.unity3d.com/Packages/com.unity.visualeffectgraph@12.0/manual/index.html.

Unity (n.d.). "Welcome to Unity." Retrieved 28.04.23, 2023, from https://unity.com/our-company.

visheshy2ey (2022). "RESTful Web Services." Retrieved 18.05.23, 2023, from https://www.geeksforgeeks.org/restful-web-services/.

Vyas, K. (2023). "8 Major Advantages of Using MySQL." Retrieved 20.05, 2023, from https://www.datamation.com/storage/8-major-advantages-of-using-mysql/.

w3schools (n.d.). "Chart.js." Retrieved 2023, 20.05, from https://www.w3schools.com/ai/ai_chartjs.asp.

w3schools (n.d.). "Node.js Introduction." Retrieved 28.05.23, 2023, from https://www.w3schools.com/nodejs/nodejs_intro.asp.

West, D. (n.d.). "Agile scrum roles and responsibilities." Retrieved 18.05.23, 2023, from https://www.atlassian.com/agile/scrum/roles.

Yasar, K. (2022). "software development kit (SDK)." Retrieved 20.05, 2023, from https://www.techtarget.com/whatis/definition/software-developers-kit-SDK.

Zoho (n.d.). "Sprint review vs. sprint retrospective." Retrieved 20.05.23, 2023, from https://www.zoho.com/sprints/sprint-reviews.html.

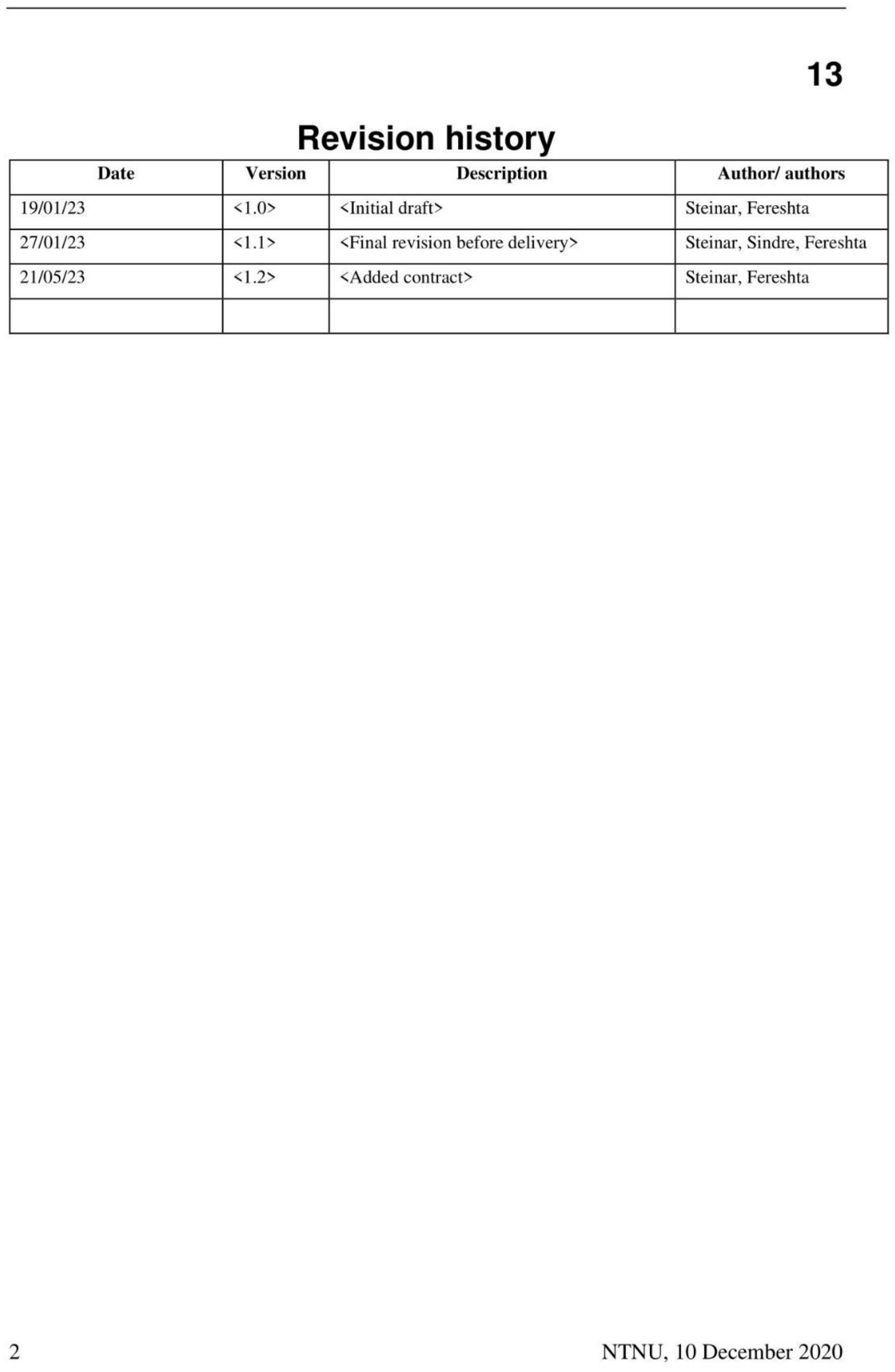
Appendices

A Preliminary project plan

13

Eye-tracking in VR
Preliminary project plan

Version 1.2



13

Revision history

Date	Version	Description	Author/ authors
19/01/23	<1.0>	<Initial draft>	Steinar, Fereshta
27/01/23	<1.1>	<Final revision before delivery>	Steinar, Sindre, Fereshta
21/05/23	<1.2>	<Added contract>	Steinar, Fereshta

Table of contents

1. Measures and frameworks

1.1 Briefing

Why this task. How did you get hold of it.

This project was a part of an Industry project that was done by Steinar Hjelle Midthus and Trine Merete Staverløkk, in fall 2022. The industry project gave an overview over what eye tracking in VR is, what research has been done on it, newer headsets and their capabilities. This sparked an interest, which lead to the industry project being taken as a bachelor thesis.

1.2 Problem statement / project description and performance goals

Task description, first draft of the problem and result targets

The owner of the project wants us to see if there is any promise in implementing eye tracking in an VR simulator, and what kind of data we can gather. Since the simulator that they want to implement this into is portable and aimed to be easy to use the same must be said with the eye tracking data and its visualisations. Also, they want to have an "expert" example to compare the rest of the trainees against.

In the first version we would like to have a basic backend, frontend and a basic simulation that sends data to the backend and lists it on the frontend. The second stage in development we would like to have a more advanced front-end webpage that shows graphs of the collected data. The backend will now have more permanent storage and the unity demo will be more finished with tasks that you can do, and that we can analyse. The third stage will have a complete webpage with authentication and the backend will be complete. Also, a wish from the project owner is to add a functionality to allow live feedback to the user or an adaptive training system. Both are possible, but the adaptive training system will be easier to implement in the first place since it can be done with code.

Problem you will explore in the project. This is a first draft that can be changed if conditions change along the way.

We are going to explore how and if eye tracking data can help with objective feedback to a user in a simulated environment. How can this data in its most basic capacity be used to give feedback to the user, and how the data should be stored, presented, and processed to give good feedback.

The result objectives tell what must be achieved when the project is finished.

The main objective is to see if we can make a sustainable system to analyse eye tracking data from VR and how this can be generalised, so it can be used with multiple headsets. The data must be visualized in an understandable way that the trainees can use to improve their performance.

1.3 Performance measures

What is the goal for you / the group and what long-term effects or gains does the business seek to achieve by making use of the results from the project.

Improved objective evaluations of their trainees so that they can improve their workflow and viewing patterns. Since the instruments on a boat can be more distracting if they are relied on too much and not checked against reality. By using eye-tracking data this can be reflected in the data that we collected. They also want to look at how trainees and experts viewing behaviour is to improve onto it.

1.4 Frames

Need for money, equipment, and time. Special needs materials and rooms.

It would be nice to have access to two headsets and thus far we have acquired a Hp Reverb G2 Omnicept edition from NTNU and an Oculus Quest Pro from Norwegian Coastal Administration. Both headsets have integrated eye tracking. Other things like two laptops are also needed to get the headsets up and running at campus since the group members does not have VR capable laptops.

When working with VR headsets we will utilize the L167 room at NTNU Ålesund that is reserved for people writing their bachelor's degree. If any other room is needed for more space the group members will book one. When the meetings are physical a room should be booked at campus to host the meeting there.

Projects developed with 3D graphics, like we are making in Unity, contains large changes and files, making them ill-fitted for traditional repository solutions such as GitHub and GitLab. GitHub provides a solution for this by using GIT Large File Storage, and GitHub data bandwidth packs. Using these packs cost 50kr/month, and one pack provides 50GB of storage. We will likely need one or two of these packs.

2. Organization

Which actors are involved in the project.

Group members:

- Steinar Hjelle Midthus
- Sindre Skorpen
- Malin Brevik Synnes
- Fereshta Ahmadi

Project owner:

- Norwegian Coastal Agency
    - Odd Sveinung Hareide

Supervisor at NTNU:

- Di Wu

3. Implementation

3.1. Main activities

Listing of main activities.

What is done, who does it, why is it done, how is it done. When is it done, necessary prerequisites before it can be done, documentation / results of what was done.

The list of main activities is supposed to be on Jira. This includes what needs to be done, which member does it, and how it is done. The date of when it is done, the necessary prerequisites before it can be done and the documentation/results of what was done are also on Jira. All user stories are then separated into issues in Jira for the developers to pick from.

We should come together as a team, after each Jira sprint and summarize what we did in a Sprint review. Each sprint should cover one week at the time. Everyone in the group has access to the Jira board, so that they can see what task the other members are doing. This makes it easier to have an informal meeting at the end of a sprint, where each group member shows what they accomplished, while the stakeholders provide feedback. Every other week we will have a meeting with the project owner if they are available and show the progress of the project.

3.2. Milestones

Listing of critical dates

What to do?	Critical dates
Preliminary project plan	27.01.2023
Feedback on preliminary project plan, approval	03.02.2023
Oral presentation in English	21.04.2023
Submit report to supervisor	05.05.2023
Preparation of records. Deliver to Anders Sætersmoen for printing 2 days before presentation.	18.05.2023
Report submission and attachments on Inspera.	22.05.2023
Presentation of assignment in plenary session (students, employees, project owner, invited guests)	19.05.2023
Delivery of the report	20.05.2023

4. Follow-up and quality assurance

4.1 Quality assurance

To ensure the code quality we follow the norms that are set by each language/framework. Also, we supplement by using addons like Sonar Lint and Check style to ensure good code quality. In addition to this, documentation is very important and group members are obliged to write documentation on methods and classes so that no misunderstandings occur during development. All code that is testable should also be tested to ensure that the methods are behaving as we planned.

Code review will also be utilized by request from the group members to ensure good quality. It's not mandatory but will be used by those who wants to improve their knowledge or ask for second opinions.

4.2 Reporting

If there are any social problems can be reported and discussed with any group members, but the group members can also contact the group leader if they are not comfortable discussing it with many people.

After each sprint the group will write a sprint review to assess how good or bad the previous sprint was and what tasks needs more time. Then this review will be stored and put into the final report of the project.

In addition to that, a stand-up will happen every day. This means that all the group members should take 10 minutes before the lunch break and tell each other if they have problems. Maybe the other group members have the solution or suggestions on how to solve them.

5. Risk assessment

Risk analysis that assesses vulnerabilities in the project (event, probability, consequence and measures).

There are many things that can affect this project since there are so many moving parts. The biggest risks are bad processes, too large scope, unforeseen issues with VR and sick members.

An unforeseen issue with VR that could occur is that we don't get an academic license for the Tobii XR library for the Omnicept headset. In the fall of 2022, we also applied for an academic license but got no response after they reached out. The project is now secured using the Meta Quest Pro and its movement SDK instead, but the Omnicept edition headset would also be useful.

Too large scope can also occur since the project is so open to interpretation. There are so many things that can be done in VR like visualize the data in the virtual environment. So that the members need to take one task at a time and ensure that it's doable to a certain degree to reduce the risk. One issue that is not critical should not be in focus for too long to ensure the project's success.

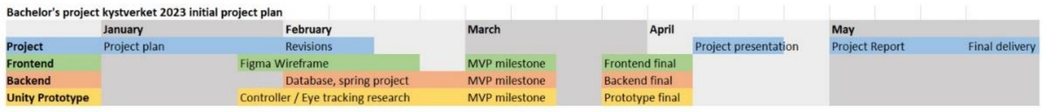
6. Attachments

The following documents are delivered as separate files when submitted to Blackboard in January (mandatory work requirement), but not in the final delivery of main report on 20 May!

6.1 Schedule

Describe the plan for phases in the project. It can be, e.g. Gantt chart or export from master plan in Confluence/Wiki Phases and/or main activities must be put in a time perspective – what happens in which weeks?

13



Bachelor's project kystverket 2023 initial project plan

	January		February		March		April		May	
Project	Project plan		Revisions				Project presentation		Project Report	Final delivery
Frontend		Figma Wireframe			MVP milestone	Frontend final				
Backend			Database, spring project		MVP milestone	Backend final				
Unity Prototype		Controller / Eye tracking research			MVP milestone	Prototype final				

6.2 Agreement documents

6.2.1 Employment contract for the bachelor group

Work contract for bachelor's thesis in collaboration with the Norwegian Costal Administration

Members: Fereshta Ahmadi, Malin Brevik Synnes, Steinar Hjelle Midthus og Sindre Skorpen

Introduction

This work contract is based on a set of typical measurements, task delegation, procedures, and guidelines for interactions for student workers. The work contract is filled with own interpretations of what is meant by these and how to achieve these. Det legges til eller fjernes punkter etter egen vurdering for tilpassing til oppgaven.

Goals (what to accomplish?)

1. *Performance measure*
   Get to know each other, build trust in each other, and increase motivation for the course
   Support each other during the process and do not overlook each other.
2. *Work effectively*
   Report illness or absence. If you do not finish or struggle with a given task, let the group know so we can fix it
3. *Act flexible and solution-oriented behaviour*
   All the team meetings should be available for all members of the group, even digital. The majority decide, but no one shall be ignored.

Performance goals

1. Deliver every task to the right time
   Be done with task at the right time. See the points above.
2. Achieve a certain grade
   Aim at the best possible grade, and work towards it. Then we will get a good grade.
3. Complete the study
   No one has plans of dropping out of the study program.

Roles and division of tasks (How to organize the work?)

What type of roles do we need – which structure should we have?

A. *Team management*
   Team management is made by all the members and the tasks should be set up in a structured way.

B. *Meeting organization (summons, preparation and management)*
   Group management takes the responsibility, but the members should be available too. Members should meet to the scheduled time, otherwise let the others know that the member will be late.

C. *Archives/document manager*
   Every group member has a shared responsibility of handling documents. Documents regarding the project are saved either in a Teams-group or in the project confluence page. Document progress will be tracked in Jira.

D. *Referent*
   The meeting report is written by a member pointed out under the group meeting. This is not a "must". A template will be used.

E. *Delivery manager, quality assurance of what needs to be submitted*
   The group member that takes the responsibility for the delivery, should submit. All the members have responsibility to remember submission deadlines.

Procedures (how to do things?)

A. Group meetings
   Meetings are held with physical attendance if possible. Meetings will be conducted remotely if any members cannot attend physically. Remote meetings are usually held in a Discord server.

B. Notification in case of absence or other events
   Notify early, so that the other group members do not have to wait. If the member is sick, then it is their responsibility to tell the others as soon as it is feasible.

C. Document management
   All of the documentation parts should be on teams. Subsequent in the study, a repository will be made. This repository will contain all the code.

D. *Submission of group work*
   All the group members can see the submission, so that we are on the same page and can point out any possible errors.

Interaction (how to perform together?)

A. *Attendance and preparation*

Look at the videos and try to understand the stuff before we have meetings. Simultaneously meet precisely. Expecting that all the group members is in measure with the substance.

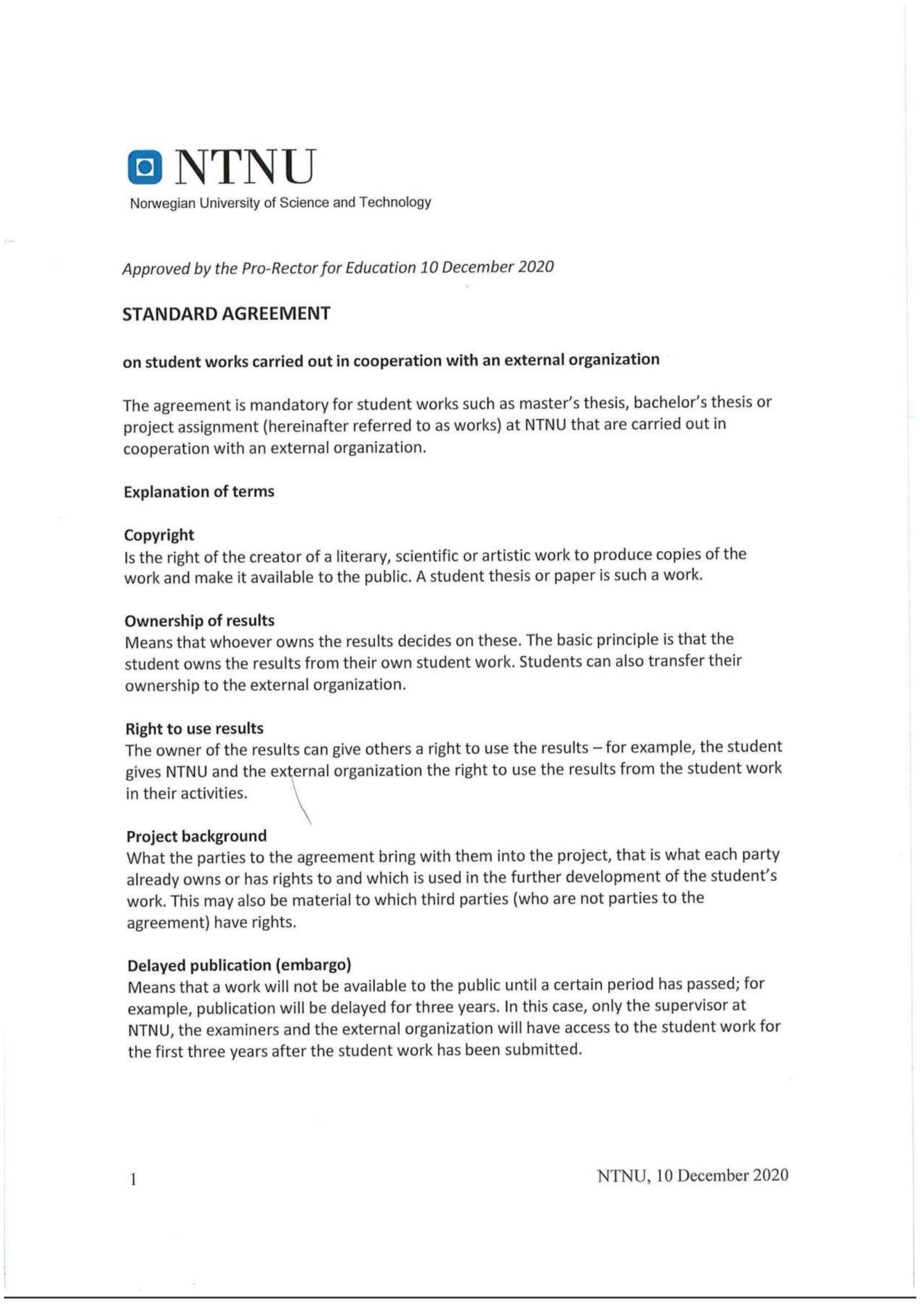
B. *Presence and engagement*
Group members should be present in person or remotely when the bachelor course is active in the course plan. Exceptions can be given if the group member provides notice before they are expected to be present.

C. *How to support each other*
We work effective together and finish the task. If one group member is not working, because of different reasons, the others should try to fill in where needed.

D. *Disagreements, breach of contract*
Discuss with the rest of the team involved, so that the team can agree on a suitable course of action.

6.2.2 3-party agreement

Three-party agreements are drawn up between NTNU, the client, and students. Template available at the learning platform and inside. (Standard agreement)



NTNU
Norwegian University of Science and Technology

Approved by the Pro-Rector for Education 10 December 2020

STANDARD AGREEMENT

on student works carried out in cooperation with an external organization

The agreement is mandatory for student works such as master's thesis, bachelor's thesis or project assignment (hereinafter referred to as works) at NTNU that are carried out in cooperation with an external organization.

Explanation of terms

Copyright
Is the right of the creator of a literary, scientific or artistic work to produce copies of the work and make it available to the public. A student thesis or paper is such a work.

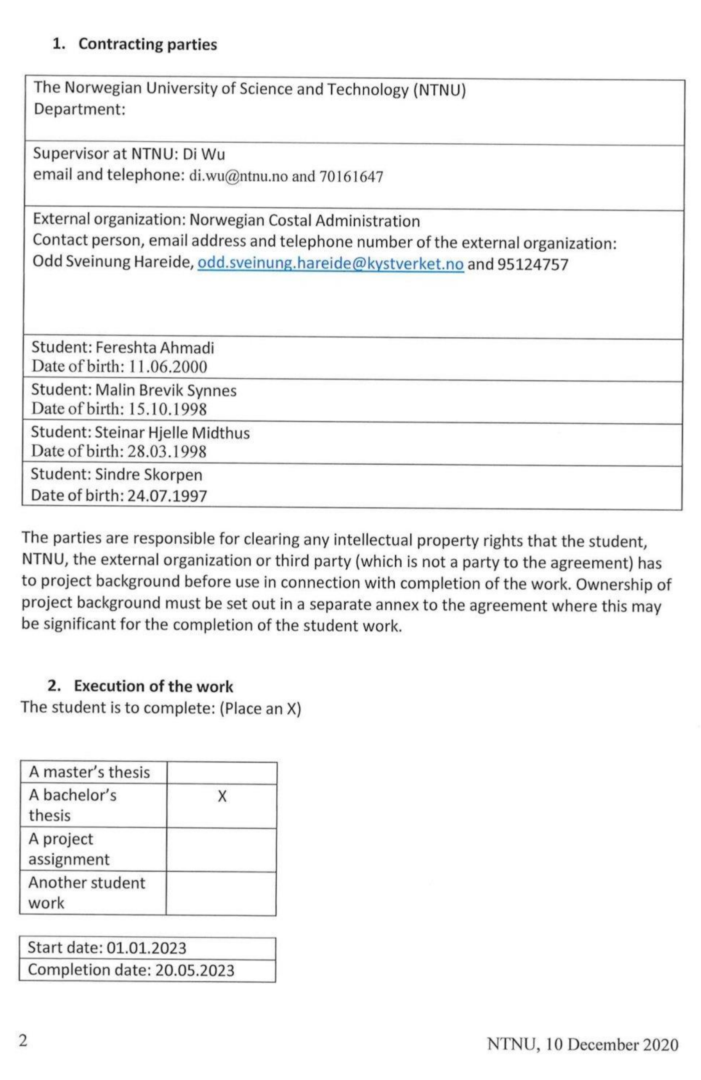
Ownership of results
Means that whoever owns the results decides on these. The basic principle is that the student owns the results from their own student work. Students can also transfer their ownership to the external organization.

Right to use results
The owner of the results can give others a right to use the results – for example, the student gives NTNU and the external organization the right to use the results from the student work in their activities.

Project background
What the parties to the agreement bring with them into the project, that is what each party already owns or has rights to and which is used in the further development of the student's work. This may also be material to which third parties (who are not parties to the agreement) have rights.

Delayed publication (embargo)
Means that a work will not be available to the public until a certain period has passed; for example, publication will be delayed for three years. In this case, only the supervisor at NTNU, the examiners and the external organization will have access to the student work for the first three years after the student work has been submitted.

1 NTNU, 10 December 2020



1. **Contracting parties**

The Norwegian University of Science and Technology (NTNU) Department:
Supervisor at NTNU: Di Wu email and telephone: di.wu@ntnu.no and 70161647
External organization: Norwegian Costal Administration Contact person, email address and telephone number of the external organization: Odd Sveinung Hareide, odd.sveinung.hareide@kystverket.no and 95124757
Student: Fereshta Ahmadi Date of birth: 11.06.2000
Student: Malin Brevik Synnes Date of birth: 15.10.1998
Student: Steinar Hjelle Midthus Date of birth: 28.03.1998
Student: Sindre Skorpen Date of birth: 24.07.1997

The parties are responsible for clearing any intellectual property rights that the student, NTNU, the external organization or third party (which is not a party to the agreement) has to project background before use in connection with completion of the work. Ownership of project background must be set out in a separate annex to the agreement where this may be significant for the completion of the student work.

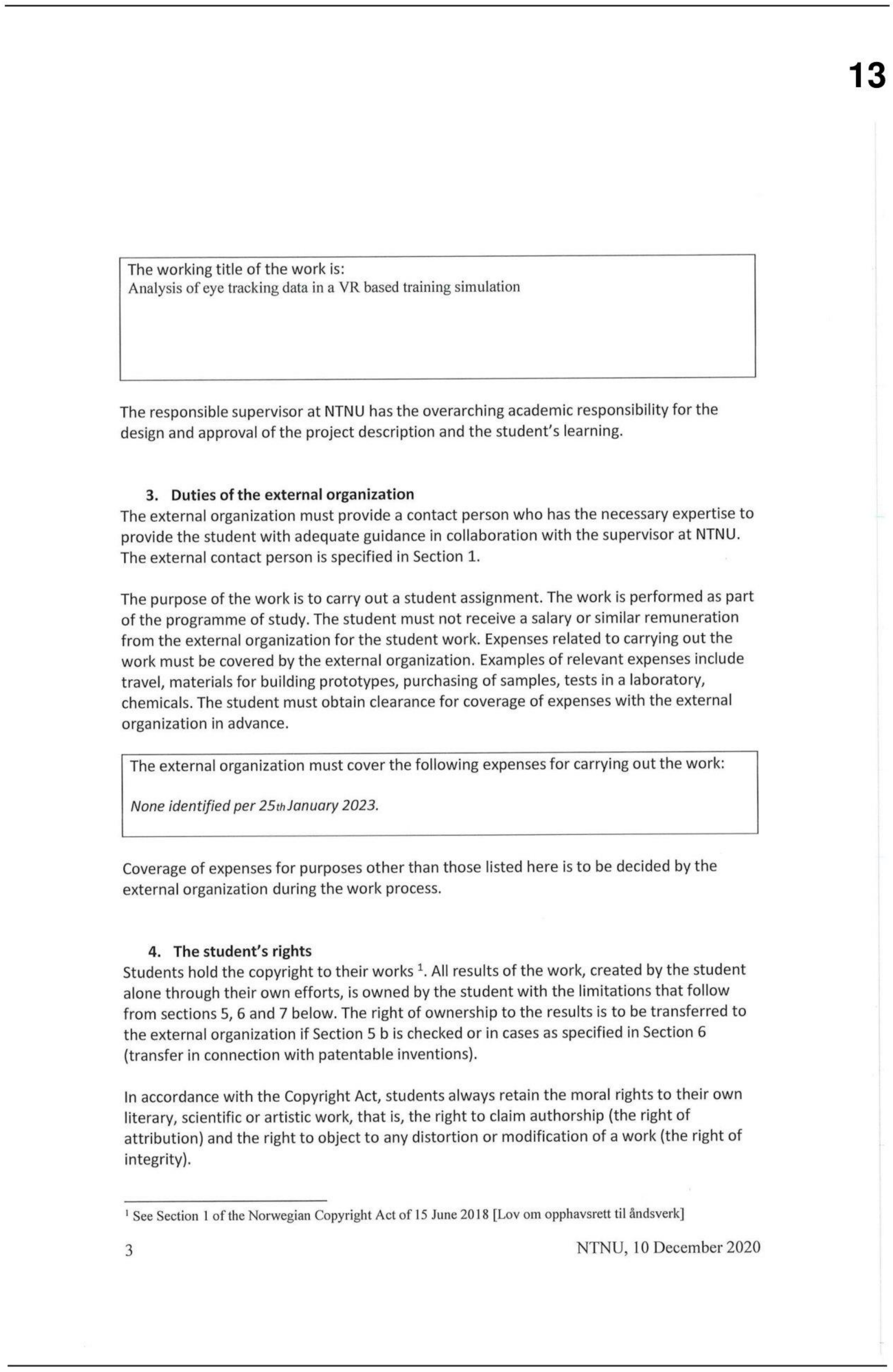
2. **Execution of the work**

The student is to complete: (Place an X)

A master's thesis	
A bachelor's thesis	X
A project assignment	
Another student work	

Start date: 01.01.2023
Completion date: 20.05.2023

2 NTNU, 10 December 2020



The working title of the work is:
Analysis of eye tracking data in a VR based training simulation

The responsible supervisor at NTNU has the overarching academic responsibility for the design and approval of the project description and the student's learning.

3. Duties of the external organization

The external organization must provide a contact person who has the necessary expertise to provide the student with adequate guidance in collaboration with the supervisor at NTNU. The external contact person is specified in Section 1.

The purpose of the work is to carry out a student assignment. The work is performed as part of the programme of study. The student must not receive a salary or similar remuneration from the external organization for the student work. Expenses related to carrying out the work must be covered by the external organization. Examples of relevant expenses include travel, materials for building prototypes, purchasing of samples, tests in a laboratory, chemicals. The student must obtain clearance for coverage of expenses with the external organization in advance.

The external organization must cover the following expenses for carrying out the work:

None identified per 25th January 2023.

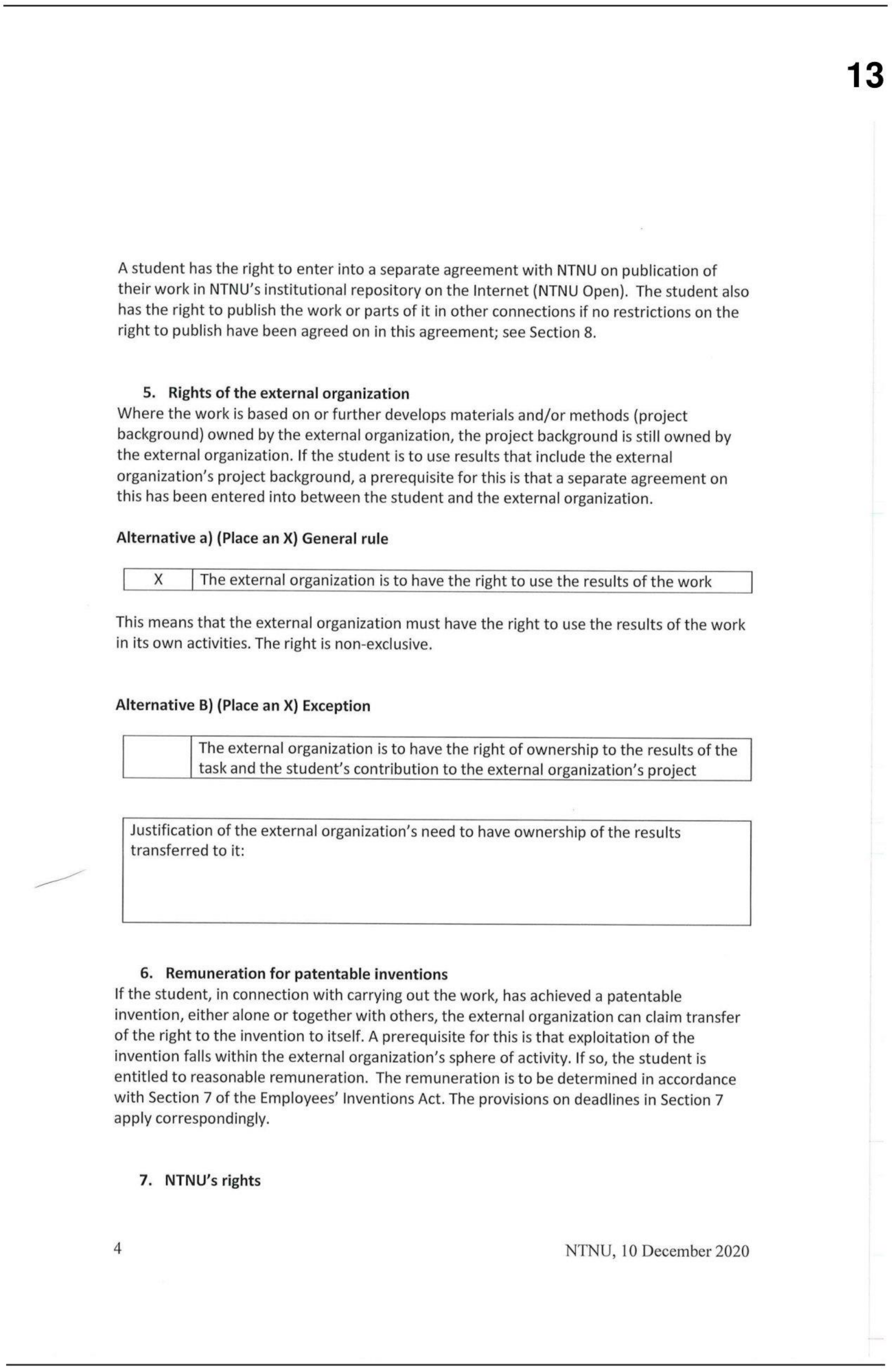
Coverage of expenses for purposes other than those listed here is to be decided by the external organization during the work process.

4. The student's rights

Students hold the copyright to their works [1]. All results of the work, created by the student alone through their own efforts, is owned by the student with the limitations that follow from sections 5, 6 and 7 below. The right of ownership to the results is to be transferred to the external organization if Section 5 b is checked or in cases as specified in Section 6 (transfer in connection with patentable inventions).

In accordance with the Copyright Act, students always retain the moral rights to their own literary, scientific or artistic work, that is, the right to claim authorship (the right of attribution) and the right to object to any distortion or modification of a work (the right of integrity).

[1] See Section 1 of the Norwegian Copyright Act of 15 June 2018 [Lov om opphavsrett til åndsverk]



A student has the right to enter into a separate agreement with NTNU on publication of their work in NTNU's institutional repository on the Internet (NTNU Open). The student also has the right to publish the work or parts of it in other connections if no restrictions on the right to publish have been agreed on in this agreement; see Section 8.

5. **Rights of the external organization**

Where the work is based on or further develops materials and/or methods (project background) owned by the external organization, the project background is still owned by the external organization. If the student is to use results that include the external organization's project background, a prerequisite for this is that a separate agreement on this has been entered into between the student and the external organization.

Alternative a) (Place an X) General rule

X	The external organization is to have the right to use the results of the work

This means that the external organization must have the right to use the results of the work in its own activities. The right is non-exclusive.

Alternative B) (Place an X) Exception

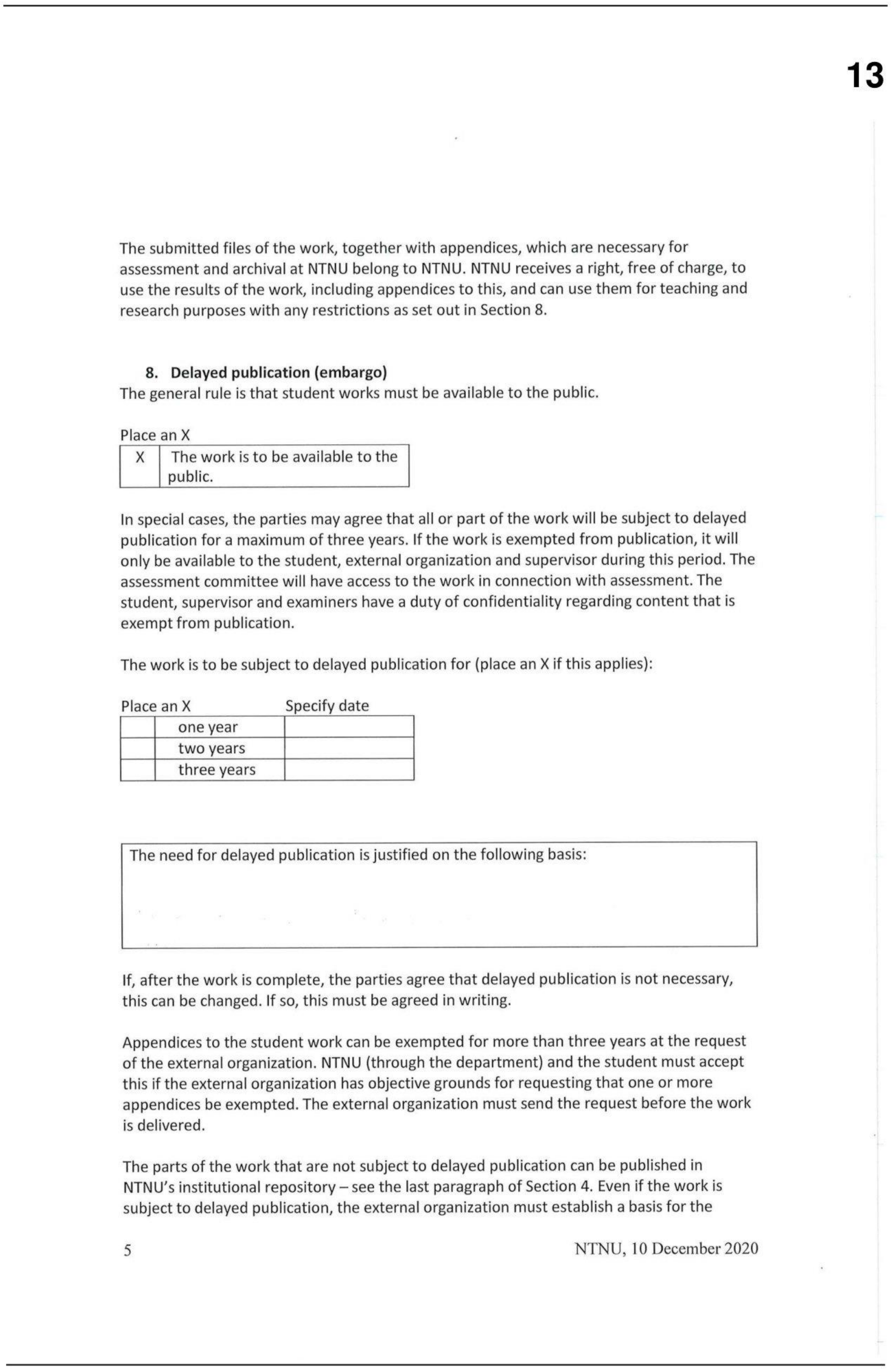
	The external organization is to have the right of ownership to the results of the task and the student's contribution to the external organization's project

Justification of the external organization's need to have ownership of the results transferred to it:

6. **Remuneration for patentable inventions**

If the student, in connection with carrying out the work, has achieved a patentable invention, either alone or together with others, the external organization can claim transfer of the right to the invention to itself. A prerequisite for this is that exploitation of the invention falls within the external organization's sphere of activity. If so, the student is entitled to reasonable remuneration. The remuneration is to be determined in accordance with Section 7 of the Employees' Inventions Act. The provisions on deadlines in Section 7 apply correspondingly.

7. **NTNU's rights**



The submitted files of the work, together with appendices, which are necessary for assessment and archival at NTNU belong to NTNU. NTNU receives a right, free of charge, to use the results of the work, including appendices to this, and can use them for teaching and research purposes with any restrictions as set out in Section 8.

8. Delayed publication (embargo)

The general rule is that student works must be available to the public.

Place an X

X	The work is to be available to the public.

In special cases, the parties may agree that all or part of the work will be subject to delayed publication for a maximum of three years. If the work is exempted from publication, it will only be available to the student, external organization and supervisor during this period. The assessment committee will have access to the work in connection with assessment. The student, supervisor and examiners have a duty of confidentiality regarding content that is exempt from publication.

The work is to be subject to delayed publication for (place an X if this applies):

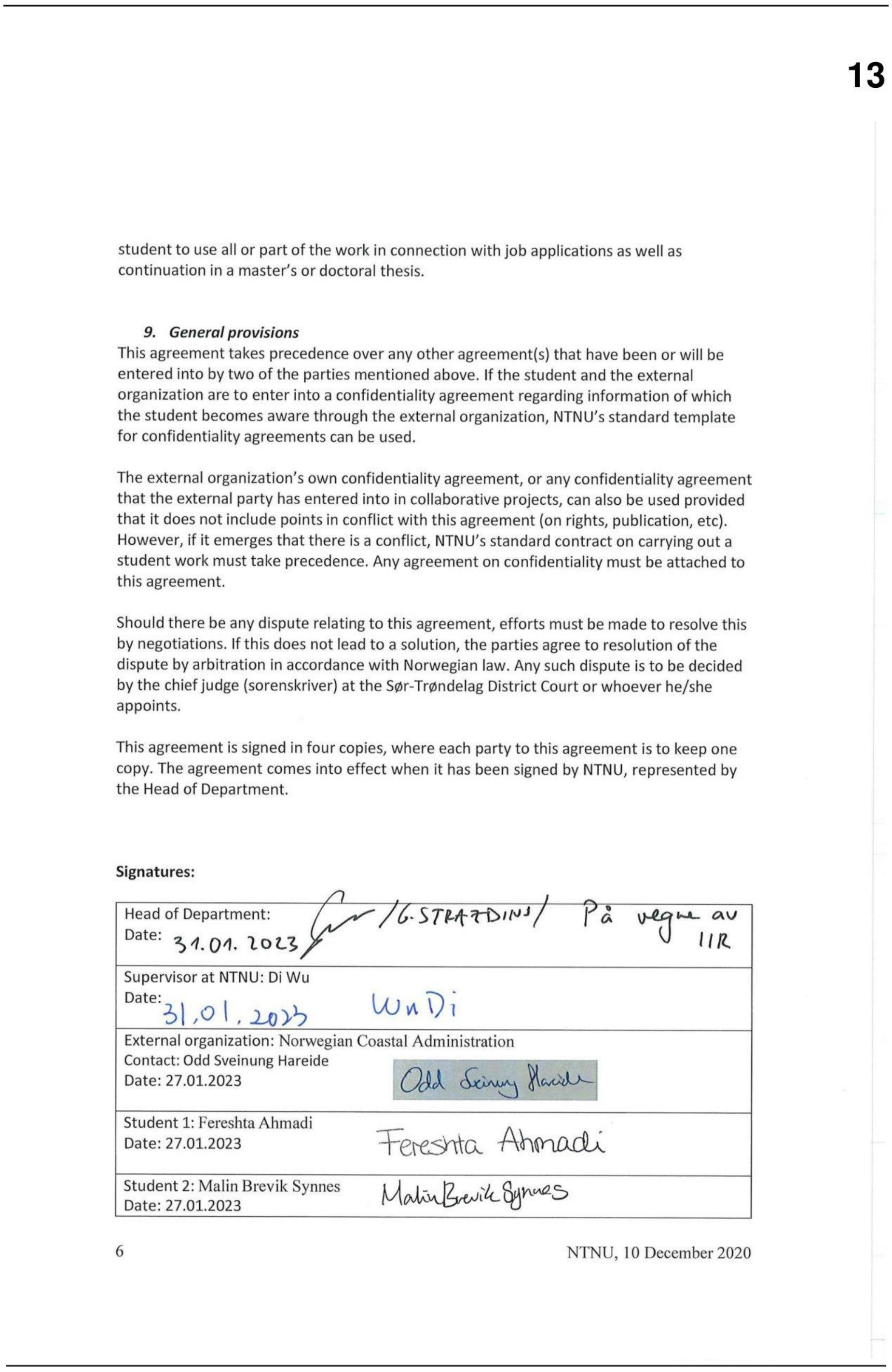
Place an X		Specify date
	one year	
	two years	
	three years	

The need for delayed publication is justified on the following basis:

If, after the work is complete, the parties agree that delayed publication is not necessary, this can be changed. If so, this must be agreed in writing.

Appendices to the student work can be exempted for more than three years at the request of the external organization. NTNU (through the department) and the student must accept this if the external organization has objective grounds for requesting that one or more appendices be exempted. The external organization must send the request before the work is delivered.

The parts of the work that are not subject to delayed publication can be published in NTNU's institutional repository – see the last paragraph of Section 4. Even if the work is subject to delayed publication, the external organization must establish a basis for the



student to use all or part of the work in connection with job applications as well as continuation in a master's or doctoral thesis.

9. General provisions

This agreement takes precedence over any other agreement(s) that have been or will be entered into by two of the parties mentioned above. If the student and the external organization are to enter into a confidentiality agreement regarding information of which the student becomes aware through the external organization, NTNU's standard template for confidentiality agreements can be used.

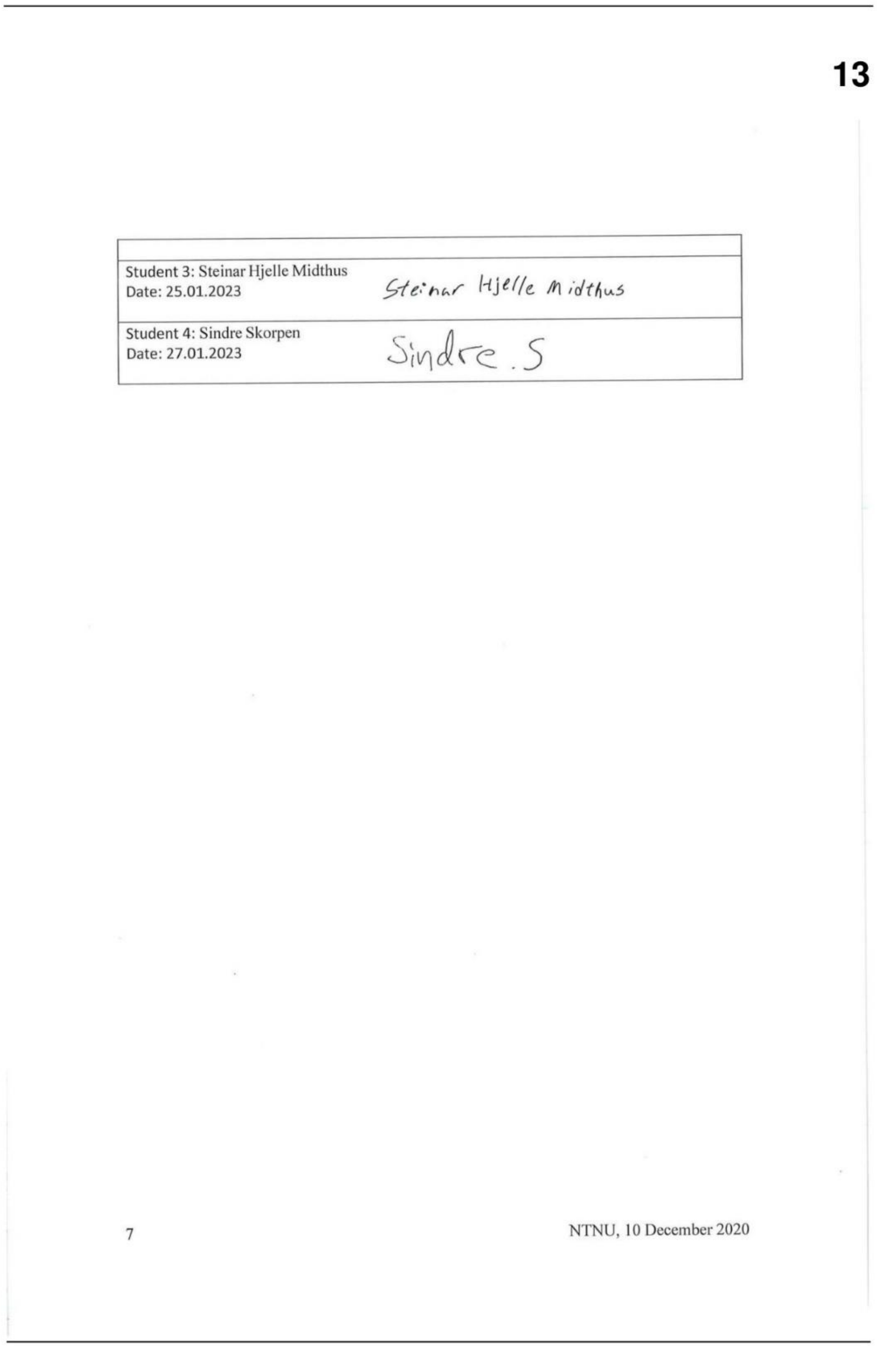
The external organization's own confidentiality agreement, or any confidentiality agreement that the external party has entered into in collaborative projects, can also be used provided that it does not include points in conflict with this agreement (on rights, publication, etc). However, if it emerges that there is a conflict, NTNU's standard contract on carrying out a student work must take precedence. Any agreement on confidentiality must be attached to this agreement.

Should there be any dispute relating to this agreement, efforts must be made to resolve this by negotiations. If this does not lead to a solution, the parties agree to resolution of the dispute by arbitration in accordance with Norwegian law. Any such dispute is to be decided by the chief judge (sorenskriver) at the Sør-Trøndelag District Court or whoever he/she appoints.

This agreement is signed in four copies, where each party to this agreement is to keep one copy. The agreement comes into effect when it has been signed by NTNU, represented by the Head of Department.

Signatures:

Head of Department: Date: 31.01.2023	[illegible] /G.STRAZDINS/ På vegne av IIR
Supervisor at NTNU: Di Wu Date: 31,01,2023	Wu Di
External organization: Norwegian Coastal Administration Contact: Odd Sveinung Hareide Date: 27.01.2023	Odd Sveinung Hareide
Student 1: Fereshta Ahmadi Date: 27.01.2023	Fereshta Ahmadi
Student 2: Malin Brevik Synnes Date: 27.01.2023	Malin Brevik Synnes



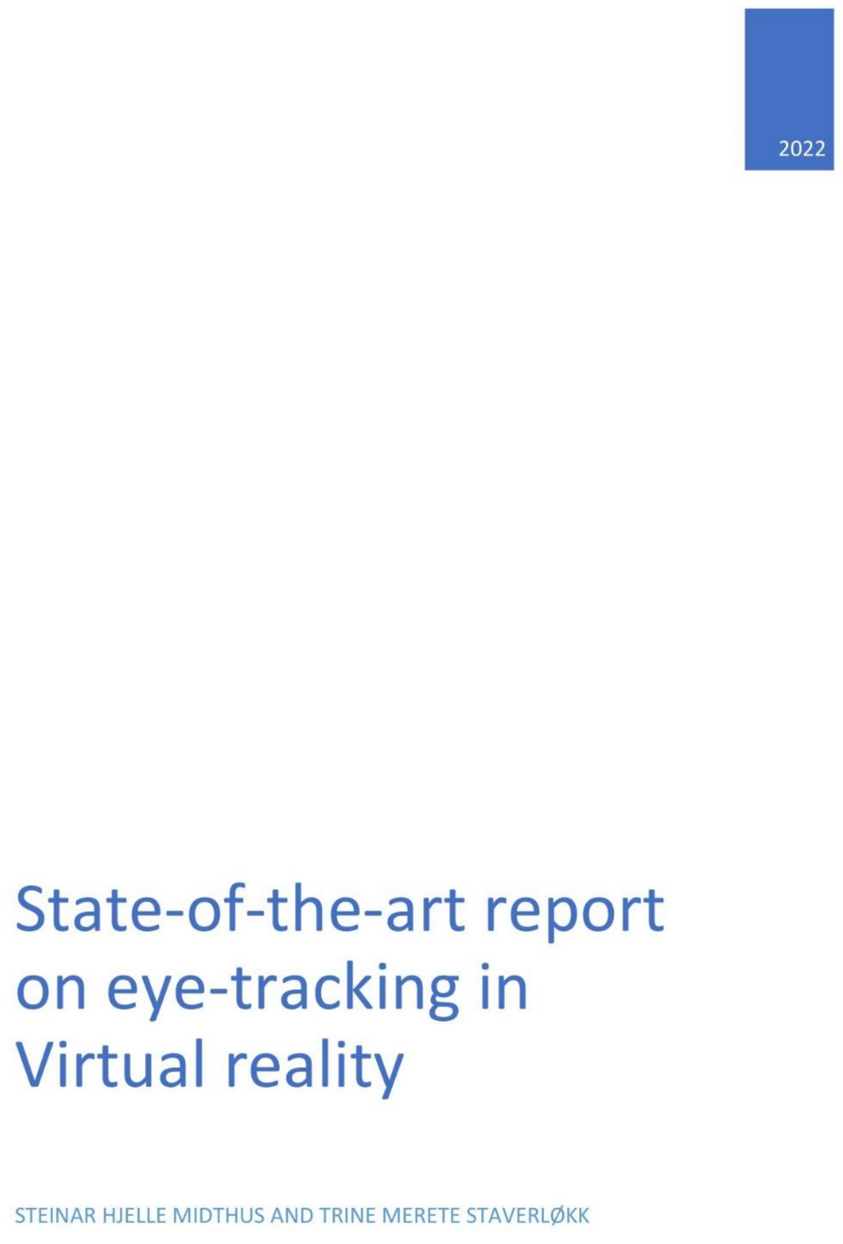
13

Student 3: Steinar Hjelle Midthus Date: 25.01.2023	Steinar Hjelle Midthus
Student 4: Sindre Skorpen Date: 27.01.2023	Sindre.S

7 NTNU, 10 December 2020

1 NTNU, 10 December 2020

B Industry report



2022

State-of-the-art report on eye-tracking in Virtual reality

STEINAR HJELLE MIDTHUS AND TRINE MERETE STAVERLØKK

Innholdsfortegnelse

Abstract

Eye-tracking enables us to observe the behavior of the subjects that are using it by looking at their gaze behavior. It can show what the users find interesting and find patterns that can help us improve their performance. In this report, we explore what eye-tracking in general is and how it works. Then we go through some of the concepts that are needed to understand how eye-tracking works and how it can be implemented with VR. After that, we explain how the making of this report has been in general detail and show some demos that we tried out to get a better understanding of the different concepts. The results show what kind of headsets exist today and the different general specifications they have. We also show what studies we have read to get information about both eye-tracking itself and eye-tracking in VR. This way leading to a discussion about what should be thought of when choosing a headset and what to keep in mind when showing the subjects or end users the results from the eye-tracking to not overwhelm them.

1.0 Introduction

The purpose of this industry project is to make a state-of-the-art report that sums up what research has been done within the space of VR and eye-tracking analysis. We also look at existing and future headsets and what their capabilities are. Some headsets might offer better portability but might not support facial and heart rate tracking. Other headsets might have all the pros except battery life, it all depends on the usage and what you want from the headset.

2.0 Theory

2.1 VR, AR, and XR

Virtual reality (VR) is a term that is used for devices that are used to stimulate certain senses and extend them beyond our reality. (Lawlor, 2016) This can be done by having a head-mounted display (HMD) that has a screen and some speakers. This way alerting what the user's visual and hearing sense is by rendering a new virtual environment (VE) that blocks out the current reality. This virtual environment can then be created to simulate a training situation or something else like a game. Augmented reality (AR) is not the same as VR since it does not block out the real world from its view. (Lawlor, 2016) Instead, AR adds overlays and other information to the current world. Extended reality (XR) is a term that contains both VR and AR. (Leland, 2017) This way binds the two terminologies with similar goals into one term.

An HMD that has all the components needed to run, power, and store the wanted application is called a standalone HMD. Since the headset has all, it needs to simulate the wanted content. Another type of HMD is the weird version that needs an external computer to render the virtual world for it.

2.2 The eye

When light enters the eye through the pupil it hits a layer that is called the retina located in the back of the eye. (Tobii, 2022a) The retina contains both cones and rods that are responsible for perceiving what we are looking at. The spot that is in the center is called the fovea and perceives what the eye is directly looking at. The fovea has the greatest number of cones and rods and thus the best visual quality. The objects that are not currently in focus are in the peripheral vision and have lower visual quality the further from the fovea they are.

According to Tobii XR Devzone (Tobii, 2022b), the movement of the eye can be divided into five different categories. Fixation is the first one where an object is in focus for a certain period and the eye is still on that object. Saccades are the second one that is the rapid eye movement that is done when the eye changes the fixation target. During rapid movement, the quality of the vision is reduced. The third one is smooth pursuit which is when the eye focuses on an object in motion, and the visual quality does not suffer like the saccades. Vergence is the fourth movement that is done when an object changes distance to the eye. The eyes converge when it moves inwards towards the nose and diverges when they move outwards towards the ears. This way the eye can maintain focus even though the distance to the object changes. The last movement is called vestibular-ocular reflex (VOR) and is done when an object is in focus and the head rotates. The eyes keep the focus on the object by rotating after the target, this way visual clarity is preserved.

2.3 Eye tracking and metrics

Eye tracking is a technology that combines sensors, machine learning, and algorithms to determine what the user looks at and how the eye reacted to the stimuli. (Tobii, 2022c) The information received from such a headset can be split into two parts. What the user looks at is called the gaze vector and other information like pupil size.

The metrics that are most interesting for eye-tracking data analysis in VR according to Tobii (Tobii, 2022j) are fixation duration, fixation count, average fixation duration, time to the first fixation, and first fixation duration. Fixation duration is the amount of time a fixation is held onto an object. The fixation count is the number of fixations that the object has had. Average fixation is the fixation duration divided by the average fixation count. Therefore, the average look time per fixation. Time to the first fixation is a metric that looks at how long it takes for a thing to catch the user's attention from the start of the situation. And lastly, the first fixation duration is the time the subject spends looking at the object for the first time.

The different areas in the captured visual field can also be separated into areas of interest (AOI). The time spent inside this area is known as dwell time. (Carter and Luke, 2020)

Calibration is also very important in eye tracking to ensure accurate and precise data throughout the session. (Carter and Luke, 2020) Multiple calibrations through a longer session are a good idea if the accuracy and precision of the data are important. Since under normal wear the eye tracking gear can move around.

2.3 Lens

Since a VR headset is so close to the user, the image of the flat screen must be curved and changed somehow to increase the field of view (FOV). This is done with lenses that alter the behavior of the emitted light from the screen to hit the center of the pupil and make it seem like it is further away. (Wu et al., 2021)

A Fresnel lens works by having small segments of a lens that is used to create the same effect as a normal lens but in less space and weight. (Wu et al., 2021) These segments are arranged so that they reflect light in a similar way to a normal lens and direct the light towards the pupil with an angle difference for the segment itself. Since the Fresnel lens consists of segments that have grooves between them the image quality can be reduced. The grooves also cause some color distortion around objects.

The pancake lens is a polarized lens that uses reflection between a circular polarizer, a reflective polarizer by the eye, and a half mirror in the center to shorten the distance needed to scale up the flat screen to the center of the pupil. (Wu et al., 2021) This way shortening the space needed for the lens considerably. Since the lens does not have any segments like the Fresnel lens it does not suffer from the same degree of color distortion, but the lens suffers from low light efficiency instead.

Since people have different distances between their eyes a VR headset must support different interpupillary distances (IPV). (CAO, 2022) This distance is measured by finding the distance between the pupils of each eye. When done correctly the image perceived by the user will be clearer and not distorted.

2.4 Game engine

A game engine is a framework that contains libraries that makes it possible to make games and simulations. (Arm, 2022) They often offer a 2D or 3D graphics rendering engine that supports different formats for graphics. They also often offer basic artificial intelligence (AI) and often offer physics, sound, and animation engines to make the interaction between the user and the program even more believable.

Some game engines like Unity also offer services like an assets store that contains materials, 3D/2D assets, audio, and much more. These assets are made by other unity users that register themselves as publishers in the store. (Unity, 2022)

2.5 General Hardware and foveated rendering

Chipsets are one of the main components on the motherboard that allows for the hardware to talk efficiently to each other. (Kernel, 2021) Instead of having all the different components have their om chips and standards one controller or chipset is made that handles all the communication.

Foveated rendering is a technique that takes advantage of the visual zones. (Tobii, 2022c) The benefits of foveated rendering can be to save power, speed up rendering, and better visual quality. This is done by rendering objects that are in the center of the user's vision at a higher quality than the objects that are in the peripheral vision. (Tobii, 2022e)

There are two main foveated rendering techniques, static and dynamic. (Tobii, 2022f) Static-foveated rendering is done by assuming the user's head position is always at the center of the screen. This way the user must physically move their head to see other things in more detail. Dynamic foveated rendering is using eye tracking to determine where the user is looking in real-time and thus reporting it to the graphics processing unit (GPU). And therefore, reducing the head movement needed to see objects in detail. Foveated rendering can also reduce the degree of cybersickness since the performance is better in the application. Factors like flickering and latency have also been known to induce cybersickness. (Chang et al., 2020)

2.6 Mental conditions

Cognitive load is the number of resources that the memory of a person must use to learn and process the information taken in. (Sweller, 1988) Visual fatigue is a term used to describe symptoms that is related to watching an object for too long. (Specsavers, 2022) Headaches, sore eyes, and blurry vision are some of the symptoms.

Motion sickness is a phenomenon that happens when the stimulation of the balance organ gets too big and makes the subject feel nauseous or unwell. (NHI, 2022) This can result in various symptoms. When this happens in the virtual world the visual stimulus is different from the balance organs stimuli. Therefore, resulting in similar symptoms as motion sickness. This is called cybersickness. Cybersickness can also come from higher-than-normal latency to the headset displaying the user at another location or rotation speed than in real life, thus causing motion sickness. (Chang et al., 2020) This phenomenon is also known to be related to VOR eye movement. Cybersickness has also been shown to impact cognitive load negatively. (Leroy et al. 2022)

3.0 Method and materials

3.1 Process

3.1.1 Before the goal of the project was defined

The project started after a quick meeting with the project contact from Costal Agency, in this meeting we got a pc, a Hp reverb G2 VR headset, and Tobii eye tracker glasses.
In this project, we tried to work with an agile style. So, each week was a "sprint" of a kind where each week was focused on one or two things. An example of this is week one which was reserved for testing both the ship simulator and the eye-tracking glasses that we got from the first meeting. This was done to get a basic understanding of what we were working with and the problems that could occur. Week two was spent researching what headsets were out there and what the task itself should be defined as. After week two a resolution was found for the goal of the project. The goal of the project is to make a "state-of-the-art" report where we look at what headsets are out in the market today and future headsets that are rumored to be released. At the same time read research on eye-tracking with a focus on the analysis of the data.

3.1.2 Planning

After the project's goals were defined the sprints went into automatic mode. Where it was based on what equipment like a VR headset, we had that week and what we felt like doing. So instead of dedicating ourselves to a weekly plan like a Gantt diagram, we had a basic plan to follow. Each week after that was based on finding articles and reading them, testing the VR headset, testing the examples in the Tobii XR library, writing the report, and looking at the existing headsets and their specifications and capabilities. But only one or two of these tasks was done at the time that week.

3.1.3 Last work phase

The last few weeks were mainly used to go over the research that we have found and make connections between them. At the same time filter out what we wanted to bring into the report and what research was not related to the project. The last two weeks were also used to write directly onto the report and make a cohesive text out of it.

3.1.4 Meetings and updates for industry contact

In the industry project, the meetings between the Norwegian Coastal Agency (NCA) and the group were arranged over mail. When a meeting was planned a meeting notice was sent to all the participants.

The first meeting was an introduction to what the NCA does with its simulator and how to use the equipment that was given. At the same time, we got a pair of eye-tracking Tobii Pro glasses that we could experiment with. The second meeting was arranged so that the supervisor of the project and the industry contacts could have a meeting and so that we could agree on a format for the report and the general goal of the project.

3.2 Game engines

The Hp reverb G2 omnicept supports two different game engines through the Tobii XR library. These are Unreal engine and Unity engine, and the main differences are that Unreal is based on C++ and Unity is based on C#. Since the ship simulator that NCA has developed through Morild is based on Unity we had to make use of the Unity engine as the main test engine if any examples were supposed to be tested or developed. The examples provided in the Tobii XR library are also made with Unity.

3.3 Finding and reading articles

Finding articles and research was done by using google scholar and reading the Tobii XR library documentation. The project's industry contact also provided a link to their eye-tracking-based research. The learning section was also used from the Tobii XR Devzone website to get an introduction to eye tracking and VR. In the documentation of Tobii XR, they refer to some research articles that are related to both VR headsets and eye tracking in VR so some of these were also read. The articles were then exported to pdf and put into OneNote so that we could mark important sections and write comments.

When searching for articles on google scholar the given search words were used: Eye tracking, Virtual reality eye tracking, VR Eye tracking, Virtual reality pancake lens, Virtual reality cognitive load, Eye tracking training, and eye-tracking.
The search results were filtered by reading if the abstraction and introduction of the article sounded relevant to the discussion topic of this report.

3.4 Hardware setup

Two hardware setups were used during the testing. When we tested the simulator provided by the NCA a gaming laptop was used in combination with an HP reverb G2 headset that does not support eye-tracking. And the Tobii eye-tracking glasses came with software that made it possible to record and stream the data to a computer while it was recording.

In the eye-tracking demos, a Hp Reverb G2 Omnicept edition (Omnicept) VR headset is used in the combination with a gaming pc from one of the members in this project. The VR headset has integrated eye, heart rate, and facial tracking. The general spec of the computer is a GTX 1080 with an 8700k intel i7 processor in combination with 32 GB of ram. All the prototypes have been run on an SSD.

Two demos/prototypes were tested from the Tobii XR Devzone. These are the shop demo called “Commercial Analytics: Shoozies” (Tobii, 2022k) and the conveyor demo called “Quality Control Training: Monitor Scanning” (Tobii, 2022l). For the shop demo, a keyboard and mouse had to be used since the controllers could not be emulated correctly through Steam VR. But in the conveyor demo, the controllers could be emulated using Steam VR. Since the VR headset has eye tracking, it must be calibrated to get the best quality data. This was done with the included “Hp Omnicept Eye Tracking Calibration” that comes with the basic SDK of the headset. Each time the headset was taken off and on again by one of the members the calibration tool was used to get the most precise data.

3.5 Testing

3.5.1 Morild simulator and Tobii eye-tracking glasses

The simulator provided by NCA was tested for one week during this project to get an understanding of how the simulator works, and what they wanted to observe with eye tracking in the simulator. While the group members tested this simulator, they sat in a chair to not induce any cybersickness from head motion and moving outside of the simulated seat. The members then used the chair to turn their heads so that the experience was as close to the simulation as possible. When a user enters this simulator, they can choose between different ships and coastal areas. When a ship is chosen like a tugboat the user is placed on its bridge where they then can steer the boat around. The environment of the simulator can also be edited to simulate stronger currents and darker environments. Many different scenarios were tried to get a feeling for the simulator.

The eye-tracking glasses provided by NCA were also used the same week to get a better understanding of eye tracking. As the members tested the eye-tracking glasses a live feed was streamed to the local computer to test that the glasses were working correctly. Then they walked around doing different activities to record some video. After an hour of usage, the members rewatched some of the footage to see where they have been looking and how their eyes moved.

3.5.2 Testing Omnicept headset and demos from Tobii

When we got the VR headset, we also tried to apply for an academic license for the Omnicept headset but got an error showing that we needed to contact their support email since there are some "U.S legal restrictions" when we applied. Therefore, we didn't get access to the features like cognitive load and machine learning for this project. But instead, we used the examples provided by Tobii.

In the shop demo, the user is placed in a shoe store that contains many different shoes. Each shoe then records the time the shoe is looked at, the time until the first fixation, and the fixation count. Then the different metrics are displayed as an overlay over the shoe. Outside the store, there is a control panel that can be used to turn off options like data collection, product statistics, and shuffling the shoes. By this control panel, the most viewed shoes are also on display. This demo is an example of the object-based color-coding visualization method. Where the color range is from red to green. Where green is a high amount of fixation time and red is a little amount of fixation time.

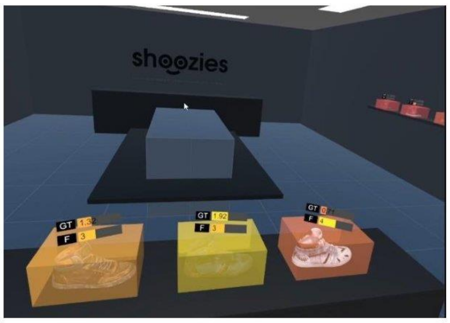
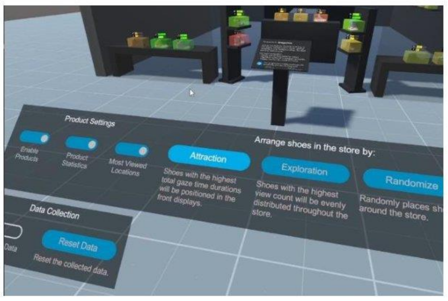


Figure 1 Picture of shoes that are in the shop demo. Here you can see the fixation count (F) and fixation duration (GT)



Shoes with the highest total gaze time durations will be positioned in the front displays.

Shoes with the highest view count will be evenly distributed throughout the store.

Randomly places sh around the store.

Reset the collected data.

Figure 2 Control panel of the shop demo

In the conveyor demo, the subject is placed at a conveyor belt where monitors are shipped out for them to control for dead pixels. During the control, the different sections of the monitor screen are divided into many boxes that the user must check with their eyes. The user must then look for dead pixels in the most efficient pattern possible. At the end of the quality control, the subject must remember how many dead pixels they found and give up an answer. When the answer is given the scan path that the subject used is displayed above included with the scan time, scan distance, missed zones, and dead pixels for the monitor. This way the subject can then optimize their viewing pattern for the next monitor and try to get as efficient as possible.

4.0 Result

4.1 Headsets with integrated eye tracking

There are mainly four big companies that have VR headsets with integrated eye tracking. The companies are HP, HTC, Pico, and now also Meta. One of the first headsets that were released with integrated eye tracking is the HTC Vive pro eye which was released in June 2019. (Brown, 2022d) Pico also came with their first eye-tracking headset Pico Neo 2 Eye on the 27th of May 2021. (Brown, 2022a) The headset offered a screen of 2048x2160 per eye with a refresh rate of 75hz. (Tobii, 2022i) Pico followed up with Pico neo 3 pro eye on Mai 10th of the next year. (Brown, 2022b) This headset offered less resolution than its predecessor at 1832x1920 per eye but had a higher refresh rate at 90hz and had the newer Qualcomm Snapdragon XR2 chipset. (Pico, 2022)

HP reverb G2 Omnicept edition was released on May 1st, 2021, and has an IPD of 60 to 68mm. (Brown, 2022c) This headset also supports heart tracking and some facial tracking with its mouth camera. The resolution of this screen per eye is also better than Pico neo 2 and 3 with 2160x2160 pixels per eye with a 90hz refresh rate. (Hp, 2022a) The headset SDK also offers real-time sensory data analysis that can show the cognitive load of the user and provide feedback on what the user can improve in real-time. The only data that can be accessed without the SDK are the eye gaze, pupillometry, and heart rate data. (Hp, 2022b)

After the rapid development of eye-tracking headsets in recent years, Meta also joined the crowd with their Meta Quest Pro headset which was announced on the 12th of October 2022. (Painter, 2022) This headset is standalone and has multiple innovations like the new pancake lenses, self-tracking controllers, and the new Qualcomm Snapdragon XR2+ chipset. The estimated battery life of the headset is one or two hours but can also be used with a cable directly to a computer. It also supports full-color passthrough to make the AR experience more immersive, so this headset is not purely VR-focused. (Meta, 2022a) The headset supports full facial and eye-tracking but does not come with full light blockers so some light might leak through the sides while using the headset in its original state. Light blockers can be bought separately at a later occasion. (Meta, 2022b) Another new feature it supports is to adjust the distance between the user's eyes and the headset's lenses. Making more space for people with glasses. It also has a bigger IPD than normal which can be between 55 and 75 mm. (Meta, 2022a)

Pico also joined Meta in 2022 by announcing the Pico 4 enterprise edition. This headset is a standalone headset that has the same lenses as Quest Pro and supports the same tracking features. It has a battery life of 2-3 hours and a resolution of 2160x2160 per eye with a refresh rate of 90hz. It does not have the newer chipset that the Meta Quest Pro has but has the older chipset that Pico Neo 3 used. (Brown, 2022e) Since the focus of the headset is enterprise customers, they also offer to make a custom app store for each enterprise where apps made by the enterprise itself can be located. (Truly, 2022)

4.2 VR hardware and foveated rendering

Currently, the manufacturer and developer of these chipsets for XR applications is QUALCOMM with their XR 2 and XR 2+ gen 1 chipsets. These chipsets are tailor-made for VR and have features that increase the efficiency of the hardware. Both chipsets support 8K video at 60 fps, 7 concurrent tracking cameras, XR with AI, Voice UI, and context awareness. (Qualcomm, 2022b) The combination of XR and AI is done to increase the performance of the hardware and decrease power usage.

The main difference between XR 2 and XR 2+ gen 1 (XR2+) is the claimed thermal performance of the XR 2+ chipset. (Qualcomm, 2022b) According to Qualcomm, the newer chipset has its thermal performance is improved by 30%. The chipset also has better support for multitasking like hand tracking, eye tracking, and video passthrough. The new chipset also has WIFI 6 support that later can decrease the latency from wireless sessions to a computer.

As the resolution of the screen increases in a VR headset, the hardware must become more complex and powerful, therefore increasing the power usage of the headset. For a wired headset, this is not the biggest problem but for a standalone headset, this will decrease the battery time, and overall better performance has never hurt anybody. That's why Nvidia has implemented foveated rendering in their "Variable Rate Super Supersampling" (VRSS) solution. The first VRSS version only supported static foveated rendering, But VRSS 2 supports dynamic foveated rendering. (Palswamy and Pang, 2021) Their VRSS 2 implementation has two modes. The adaptive mode grows the center of the visual field when the GPU has free resources to give greater detail to a larger area. Always on mode is when the visual center is a fixed size.

4.3 Libraries, software, and APIs

Tobii XR is a software development kit (SDK) that offers libraries and tools to access eye-tracking data more easily in Unity. It supports many headsets like Hp Reverb G2, Pico Neo 2 Eye, and Pico Neo 3 Pro eye. The SDK itself also offers many examples that show how to implement features like moving eyes based on the viewer's eye location and eye tracking. It also has a learning section in the documentation where they introduce topics like how the eye works or how to visualize and analyze the gathered data. (Tobii, 2022m) The only data the basic Tobii XR library provides is the combined gaze ray, left and right blinking convergence, and a timestamp. (Tobii, 2022h)

Tobii Ocumen is another service that has Tobii XR SDK integrated. This service offers more advanced data like left and right eye gaze and pupil diameter. It also offers solutions for recording sessions, python APIs to access the data, and Ocumen studio integration. (Tobii, 2022h)

4.4 Studies on eye tracking

A study done by Kang et al. (2022) looked at how eye tracking can detect the situational awareness (SA) of an operator or trainee during an oil rig simulation. This study showed that a person with a high SA created a better visual pattern than a person with a lower SA. This was also visualized with a visual scan path cluster diagram. A more random pattern was also detected when a phenomenon was detected.

The review written by Briest et al (2008) found that a longer fixation or shorter fixation amount might also be connected to the sleepiness of the subject using eye-tracking. This review also points out that there is a longer blink duration time when the subject gets sleepier. And the general awareness goes down as the person is less awake.

«Eye tracking in VR» (Clay et al., 2019) is another paper that looked at how eye-tracking data could be related to the recollection of the houses in a built city. When the subjects were shown a picture of a house, they rated it from 1 to 5 if they could find the house again and if they remembered the house. They found a connection between the viewing time of the house and the ability to remember details from it and where it was located. The study also found that subjects moved their eyes more when they explored the environment, but also noted that this might be because of the way a subject navigates in the virtual environment. Since the head is used to point in a certain direction to move in many VR applications.

Another article used eye tracking to look at how well the interface was designed for certain instruments on boats. (Hareide and Ostnes, 2017) The authors were looking at how the participants used the instruments and their graphical user interface (GUI). They also looked at the dwell time and lookbacks at the AOIs to determine how well the information was conveyed to the participants and if they got confused by the interface. One problem that they encountered was light pollution that made the recording unusable when it was too dark outside.

A similar study looked at how a set of trainees compared against experts in the usage of navigational instruments. (Oguz and Omer, 2019) And found that the experts have a better and more concentrated viewing pattern than the novices. Often the novices looked at other items like menus when it was not a part of the exercise while the experts just focused on the important parts of the instruments.

One feature that the HP Reverb G2 has that makes it stand out is the cognitive load and machine learning combination. (Hp, 2022a) Since it makes use of the included eye-tracking camera and tracks the movement of the pupil to measure the cognitive load of the subject. A paper review (Leroy et al., 2022) identifies mainly six features that are usually used to measure cognitive load. That is blinks, pupil diameter, fixation frequency, point of gaze accuracy, nystagmus, and saccades. The review also found that most experimental studies had an increase in pupil dilation the higher the cognitive load, and more blinks the higher the visual fatigue. But at the same time noting that visual fatigue and cognitive load both are affected by multiple parameters that they also share.

Another study looking at cognitive load on a train station during normal navigation (Armougum et al., 2019) found that experts felt more frustration than novices since they were unable to navigate as normal because of the observer. Since their traveling habits were on display during the testing. They also found that the participant's cognitive load was consistent with a real-world environment.

A review of eye-tracking in training situations done by Rosch and Vogel-Walcutt (Rosch & Vogel-Walcutt, 2013) also suggested that an adaptive system could be included to bring feedback to the user based on their cognitive load. This way hoping to maximize the learning outcome of the session. Much like what HP has done with their Omnicept edition headset. (Hp, 2022a) Where live feedback can be given through machine learning and cognitive load monitoring to improve the performance and learning outcome.

4.5 Visual models used by papers

Here we go over the different visual models that are used in studies and look at the different ways the models have been applied. Also, we summarize how this can be enhanced with VR representations that Tobii XR Devzone has shown.

4.5.1 Heat maps

When it comes to the statistical models used by the studies the most popular one is the heat map. A heatmap is a general statistical model that shows where the user has spent the most time and is often colored from a blue to red tone where blue colors are less viewed locations and red colors are more viewed locations. The more fixation time the brighter the color. (Tobii, 2022m) A heatmap can give a general notion of what the user is interested in but at the same time regions that are viewed for a long time can indicate confusion.

An example of a heatmap is in the study done by Clay et al. (2019). Where the distance is the third variable, this way making a 3D heatmap instead of the regular 2D heatmap. Since the data was scattered because of the distance variable they used the natural logarithm of distance to get a more collective data set that is more readable. They also used 2D heatmaps to visualize the gaze behavior while the subject was standing still.

Another example is the usage of heatmaps by Oguz & Omer (2019). Here they displayed the heatmap on top of a picture of the different instruments like a radar. This gives a better visual representation of where the fixations were on the instruments and how the experts view patterns looked versus the novices.

In VR these heatmaps can be enhanced since they can be displayed on top of the observed objects in the virtual environment. (Tobii, 2022m) This way the perspective can be changed since the camera can be moved around, and the heat map can be viewed in a 3D environment instead of a 2D picture or graph.

4.5.2 Perception maps

Perception maps and focus maps are the same things. This is a heat map that is overlayed on top of an image where the most viewed locations are in more detail than the less viewed locations. (Tobii, 2022m) The locations that do not have any fixations and therefore no fixation durations are not displayed and are blacked out. And an example of this perception map can be found used by Hareide and Ostnes (2017) who applied the perception map over a picture of the bridge on the boat that was used to record the data. This model like the heat map gives a good representation of what the user looked at since all the non-interesting details are blocked out. This map can also be implemented in VR. (Tobii, 2022m) So that the environment that was observed is visible to the observing camera.

4.5.3 Gaze plots

A gaze plot is a representation of the location, time spent, and order that an item was looked. (Tobii, 2022m) The number on a point represents the order that it was looked at and the size of the dot represents the time or duration spent looking at it. This is often used as a visual representation on top of a picture. Gaze plots can also be represented in the virtual world by plotting the gaze points directly onto the objects themselves. (Tobii, 2022m)

4.5.4 Other visualization methods in VR

Other methods of visualization are also introduced by the Tobii XR library. There are a total of six methods that Tobii lists, where heatmap, perception map, and gaze plots are three of them. (Tobii, 2022m)

The fourth one is correlated objects, where each object records from what and to what object the focus changes. And counts the number of times the objects have been looked at in sequential order. This way the correlation can say something about how strong the relation between those objects is. (Tobii, 2022m)

With normal eye-tracking glasses like the Tobii Pro Glasses 2, the session is recorded with a camera on the front of the glasses. (Tobii, 2016) But in a VR session, it could theoretically be recorded and replayed by the subject in the world they are. Instead of recording what the subject sees, we can record the entire situation. This way shows how the person looks around based on their eye tracking data. This is what Tobii calls Replay is the fifth method. (Tobii, 2022m) So instead of asking questions during a session, the replay can be used to recollect what the subject was feeling at that time.

The last method is called object-based color coding. In this method, the objects are color-coded from one standard color that represents the object when it hasn’t been looked at. In Tobii's example, the standard color is red, and it turns yellow to green the more you look at it. This gives a quick visual representation of the time each object has been looked at. (Tobii, 2022m)

Other methods like gaze point visualization are used by Clay et al. (2019) The data is only taken from one subject but illustrated as rounded points in the world. Each rounded point represents a gaze vector hitting something and the base color is blue for close and red for further away.

4.6 Problems with VR and eye tracking

The main problems found in the study done by Clay et al. (2019) are cybersickness and continuous calibration (Clay et al., 2019) During the study, they tried to reduce the effects of cybersickness by having smaller colliders on items to increase performance. But even though the performance was optimized some users still got cybersickness. In these users, they saw that the navigation behavior changed to counteract the cybersickness.

Since the accuracy of the eye-tracking data is paramount the study had to recalibrate the headset often to get accurate data. Clay et al. (2019) also pointed out that methods for autocalibration exist like statistical auto-calibration (Guenter & Tripathi, 2017). More device manufacturers must test this auto calibration and check if it's accurate.

Another example of problems that could occur during eye-tracking in VR is that the subject needs glasses. An example of this was when a group member tried to use glasses/VR lenses while testing the demos in Tobii XR. This resulted in worse tracking and therefore the subject had to take off their glasses and have a worse overall experience instead. This is one of the several problems that the paper made by Carter and Luke (2020) points out can happen during eye-tracking. Other problems pointed out by this paper include hard lenses, strong eye prescriptions, and makeup or eyelashes darkened by makeup.

Batteries are also a factor to consider when picking an eye-tracking headset. The portability and wire-free experience of a standalone headset are great, but when the headset gets more features the battery time will go down because of an increase in computational tasks. And batteries are also known to decrease in performance over the years. Previous generations of Metas standalone headsets like Quest 2 have been reported to still discharge while plugged into a computer. (Hector, 2022)

5.0 Discussion

5.1 Eye-tracking headsets comparison

We will use the Pico 4 Enterprise (P4E), Meta Quest Pro (MQP), and Omnicept as discussion headsets since they are the newest with the "best" hardware available currently.

Even though a standalone headset has its advantages over a weird headset like portability and no wires required they still have one large con which is the battery. Since the battery time can be limiting under longer sessions. The meta MQP has self-tracking controllers that are new inventions, but the battery time is between one and two hours. While the P4E has a battery life of two or three hours but has an older chipset and is not an AR headset. We cannot confirm if the P4E or MQP drains when they are connected to a computer with a cable, but since other models before them like Pico 3 and Quest 2 have done it, we can assume that it's a problem here too since the headsets themselves are more advanced than their predecessors with eye-tracking.

At the same time, the problem with battery time is avoided by the older Omnicept headset. But this headset supports a smaller IPD than the other two headsets, is wired, and has the older Fresnel lenses that have worse color distortion and image quality compared to the newer pancake lenses. Even though the lenses themselves are important the resolution and contrast of the screen should be considered. We have not gone into detail on the different screens, but the resolution is better on the Omnicept and P4E compared to the MQP. But it doesn't help if the screen is of a higher resolution if the lens itself is not of good quality.

The newer self-tracking controllers of the MQP must also be taken into consideration. Since two main advantages over the other models and their controllers. The first one is that the headset does not need a clear line of sight to the controllers so that their position of them can be more extreme, like behind you. The second reason is that the controllers themselves are rechargeable and do the tracking themselves.

5.2 How can eye-tracking help trainees

Since one of the main points of the simulator provided by Morild is that it's supposed to be portable and easy to use we look at the different perspectives and how the different perspectives of eye-tracking in VR might complicate it.

Even though the eye-tracking data can be visualized in the virtual world after a training session the question is if it's a good idea to show the data directly to the trainee. Depending on the statistical and technical knowledge of the trainee the visualizations of the eye-tracking data might be confusing since they are probably new to the technology and do not understand the data directly. Even the task of using the simulator might feel complicated to a subject. Also, if the subject must put on a headset and look at the visualizations in the virtual world it might be more confusing than helping since it adds another step that makes the process more complex. At the same time, visualization methods should be analyzed by a person with more knowledge to find patterns and not a novice user. Or statistical techniques should be used to show the most relevant information directly to the user. But at the same time, the potential of showing a novice an expert's viewing pattern has shown to be an effective learning tool. Since they can optimize their viewing pattern based on the expert's performance.

An adaptive training solution could also be made to increase the learning of the subject in real-time. If the subject looks at the wrong things for a certain amount of time a notification could come up with suggested actions. But this also adds another complex layer of code that must run in the background and could affect the runtime of the application. But at the same time might help novices and experts quicker since they can correct their behavior faster.

Other factors like how often the headset should be recalibrated to ensure good data should also be considered. Since if the user must recalibrate every 10 minutes it might get annoying after a while if the session is long. But at the same time, the importance of good data recording might outweigh the annoyance factor.

5.3 Project reflection and execution

Even though we are most happy about how the project worked out there are still some aspects that could be done better the next time around. Like finding relevant research and using better reading techniques earlier. Since this field is quite new for both group members the accompanying research was overwhelming in the beginning. This happened because we forgot to find out the basics before we jumped into the research. So that the first week of research was heavy since we learned the basics through the research that we read. Next time around it would be a good idea to find out what eye-tracking is before we investigate how it works in VR and what it can be used for. This way we also optimize the amount of research that we could read since we have a better understanding from the beginning.

Also, next time we would write more notes and summarize the research earlier and introduce a mind map to find connections between the different research earlier. This would have improved the workload and efficiency of writing the report. Halfway through the project, we started to do this and summarize the research, but it should have been done from the beginning. Next time around we could also not use that many sources from one source and try to distribute the load over more reports and companies. But this was done since Tobii had the easiest-to-understand articles.

6.0 Conclusion

When choosing a VR headset with integrated eye-tracking for training the length of the session, lenses, screen quality, and extra features should be considered. And how the different pros and cons of the different headsets might impact the user experience. For longer sessions, the Pico 4 Enterprise is a good choice since it has longer battery life. But at the same time, the Quest Pro is a good choice if the sessions are shorter.

At the same time, the complexity of the solution and ease of use should be considered to create a good user experience and get the most out of eye-tracking. We believe that eye-tracking is a great tool that can be used to improve the behavior and performance of a trainee if it’s presented understandably. Even though the data can be visualized in the virtual environment it must be in an informative format. Also, if the data can be saved a comparison can be made where the subject can see their progress and improvements over time.

Sources

Lawlor, P. (2016) Keeping the virtual world stable in VR*, QnQ Blog,* June 28. Available at: https://www.qualcomm.com/news/onq/2016/06/keeping-virtual-world-stable-vr (Accessed at: 5 December 2022)
Lawlor, P. (2016) Augmented reality is coming, and mobile technologies are leading the way*, QnQ Blog,* December 11. Available at: https://www.qualcomm.com/news/onq/2016/12/augmented-reality-and-mobile-technologies-are-leading-way (Accessed: 5 Descember 2022)
Tobii (2022a) *The eye,* Available at:

https://developer.tobii.com/xr/learn/eye-behavior/the-eye/ (Accessed: 6 November 2022)

Tobii (2022b) *Eye movements*, Available at:
https://developer.tobii.com/xr/learn/eye-behavior/eye-movements/ (Accessed: 6 November 2022)
Tobii (2022c) *What is eye tracking?,* Available at:
https://www.tobii.com/learn-and-support/get-started/what-is-eye-tracking (Accessed: 6 November 2022)
Tobii (2022d) *Principles of Foveation,* Available at:
https://developer.tobii.com/xr/learn/foveation/principles/ (Accessed: 10 November 2022)
Tobii (2022e) *Benefits and Costs,* Available at:
https://developer.tobii.com/xr/learn/foveation/rendering/benefits-costs/#cost (Accessed: 10 November 2022)
Tobii (2022f) *Essential Concepts,* Available at:
https://developer.tobii.com/xr/learn/foveation/rendering/essential-concepts/ (Accessed: 10 November 2022)
Tobii (2022g) *Tobii XR Devzone.* Available at:

https://developer.tobii.com/xr/ (Accessed: 06 November 2022)

Tobii (2022h) *Tobii Ocumen*. Available at:

https://developer.tobii.com/xr/solutions/tobii-ocumen/ (Accessed: 06 November 2022)

Tobii (2022i) *Pico neo 2 Eye Powerful. Intuitive. Insightful.* Available at:

https://www.tobii.com/products/integration/xr-headsets/device-integrations/pico-neo-2-eye (Accessed: 6 November)

Tobii (2016) *Tobii Pro Glasses 2 – Wearable Eye Tracker for Human Behavior Research.* Available at:

https://www.youtube.com/watch?v=3hcQYN0t-VM (Accessed: 5 December 2022)

Tobii (2022j) *Metrics.* Available at:

https://developer.tobii.com/xr/learn/analytics/fundamentals/metrics/ (Accessed: 28 November 2022)

Tobii (2022k) *Commercial Analytics: Shoozies.* Available at:

https://developer.tobii.com/xr/explore/commercial-analytics/ (Accessed: 28 October 2022)

Tobii (2022l) *Quality Control Training: Monitor Scanning.* Available at:

https://developer.tobii.com/xr/explore/quality-control/ (Accessed: 28 October 2022)

Tobii (2022m) *Visualizations*. Available at:

https://developer.tobii.com/xr/learn/analytics/fundamentals/visualizations/ (Accessed: 27 November 2022)

Brown, R. (2022a) *Pico Neo 2 Eye.* Available at:

https://vr-compare.com/headset/piconeo2eye (Accessed: 6 November 2022)

Brown, R. (2022b) *Pico Neo 3 Pro Eye.* Available at:

https://vr-compare.com/headset/piconeo3proeye (Accessed: 6 November 2022)

Brown, R. (2022c) *Hp Reverb G2 Omnicept Edition.* Available at:

https://vr-compare.com/headset/hpreverbg2omniceptedition (Accessed: 10 November 2022)

Brown, R. (2022d) *HTC Vive Pro Eye.* Available at:

https://vr-compare.com/headset/htcviveproeye (Accessed: 10 November 2022)

Brown, R. (2022e) *Pico 4 Enterprise.* Available at:

https://vr-compare.com/headset/pico4enterprise (Accessed: 10 November 2022)

Meta (2022a) *Dette er meta quest pro,* Available at:
https://www.oculus.com/blog/meta-quest-pro-price-release-date/ (Accessed: 6 November 2022)
Meta (2022b) *Meta Quest Pro full light blocker,* Available at:

https://www.meta.com/no/en/quest/accessories/quest-pro-full-light-blocker/ (Accessed: 9 December 2022)

Hp (2022a) *Hp Omnicept & Hp Reverb G2 Omnicept Edition.* Available at:

https://www.hp.com/us-en/vr/reverb-g2-vr-headset-omnicept-edition.html?jumpid=va_cdbrzpgewu#tab=hardware (Accessed: 6 November 2022)

Hp (2022b) *Omnicept.* Available at:

https://developers.hp.com/omnicept (Accessed: 6 November 2022)

Qualcomm (2022a) *Snapdragon XR2+ Gen1 Platform.* Available at:

https://www.qualcomm.com/products/application/xr-vr-ar/snapdragon-xr2-plus-gen-1-platform (Accessed: 7 December 2022)

Qualcomm (2022b) *Snapdragon XR2 5G Platform.* Available at:

https://www.qualcomm.com/products/application/xr-vr-ar/snapdragon-xr2-5g-platform : (Accessed: 7 December 2022)

Truly, A. (2022) The new Pico 4 Enterprise outdoes the Quest pro in one key area, *Digitaltrends.* Available at:

https://www.digitaltrends.com/computing/pico-4-enterprise-challenges-meta-quest-pro : (Accessed: 10 December 2022)

Pico (2022) *Neo 3 pro – Neo 3 pro eye.* Available at:

https://www.picoxr.com/us/neo3.html (Accessed: 10 November 2022)

Painter, L. (2022) *Meta Quest Pro: All you need to know about the VR/AR headset.* Available at:

https://www.techadvisor.com/article/743826/meta-quest-pro.html (Accessed: 10 November)

CAO (2022) *Interpupilary Distance (PD)*. Available at:

https://opto.ca/eye-health-library/interpupillary-distance-pd (Accessed: 7 December 2022)

NHI (2022) *Reisesyke*. Available at:

https://nhi.no/livsstil/reise/reisesyke/ (Accessed: 10 December 2022)

Arm, (2022) *Glossary gaming engine,* Available at:
https://www.arm.com/glossary/gaming-engines
Kernel, L. (2021) *What is a Chipset?* Available at
https://www.hp.com/us-en/shop/tech-takes/what-is-a-chipset (Accessed: 7 December 2022)
Specsavers (2022) *Hva er slitne øyne / astenopi?* Available at:

https://www.specsavers.no/hjelp-og-ofte-stilte-sporsmal/hva-er-slitne-oyne-astenopi (Accessed: 8 Desember 2022)

Leland, T. (2017) Augmented reality is coming, and mobile technologies are leading the way*, QnQ Blog,* May 30. Available at:
https://www.qualcomm.com/news/onq/2017/05/extended-reality-convergence (Accessed: 20 November 2022)
Unity (2022) *Introduction to unity asset store*. Available at:

https://unity.com/pages/introduction-to-asset-store#resources-beginners (Accessed: 20 November 2022)

Wu, S., Hsiang, E. He, Z. Zhan, T. & Xiaong, J. (2021) Augmented reality and virtual reality displays: emerging technologies and future perspectives, *Light: Science & Applications, 10, 216,* Available at:

https://doi.org/10.1038/s41377-021-00658-8 (Accessed: 01 November 2022)
Hareide, O.S., & Ostnes, R. (2017) "Maritime Usability Study by Analysing Eye Tracking Data," *Journal of Navigation*. Cambridge University Press, 70(5), pp. 927–943. doi: 10.1017/S0373463317000182. (Accessed: 1 November 2022)

Palswamy, D., & Pang, P. (2021) Technical Blog, *Dellevering Dynamic Forveated Rendering with NIVIDIA VRSS 2.* Available at:

https://developer.nvidia.com/blog/delivering-dynamic-foveated-rendering-with-nvidia-vrss-2/ (Accessed: 8 December 2022)

Clay, V., König, P., & König, S. (2019) Eye tracking in Virtual Reality, *Journal of eye movement research, 12(1),* Available at:

https://doi.org/10.16910/jemr.12.1.3 (Accessed: 10 October 2022)

Briest, S., Galley, N. , Galley, L., & Schleicher, R. (2008) Blinks and saccades as *R*indicators of fatigue in sleepiness warnings: looking tired?, *Ergonomics*, 51:7, 982-1010, Available at:

https://doi.org/10.1080/00140130701817062 (Accessed: 20 November 2022)

Kang, Z., Jeon, J., & Salehi, S. (2020). Eye tracking data analytics in virtual reality training: Application in Deepwater Horizon oil drilling operation. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *64*(1), 821 - 825. Available at:
https://doi.org/10.1177/1071181320641191 (Accessed: 10 October 2022)

Chang, E., Kim, H.T., & Yoo, B. (2020) Virtual Reality Sickness: A Review of Causes and Measurements, International Journal of Human–Computer Interaction, 36:17, 1658-1682, Available at:

https://doi.org/10.1080/10447318.2020.1778351 (Accessed: 1 December 2022)

Carter, B.T., Luke, SG. (2020) Best practices in eye tracking research, *International Journal of Psychophysiology,* 155, 49-62. Available at:

https://doi.org/10.1016/j.ijpsycho.2020.05.010 (Accessed: 1 December 2022)

Leroy, L., Lourdeaux, D., Philippe, S., & Souchet, A.D. (2022) Measuring Visual Fatiuge and Cognitive Load via Eye Tracking while Learning with Virtual Reality Head-Mounted Displays: A Review, *International Journal of Human-Computer interaction,* 38(9), 801-824, Available at:

https://doi.org/10.1080/10447318.2021.1976509 (Accessed: 07 Desember2022)

Armougum, A., Orriols, E., Gaston-Bellegarde, A., Marle, C.J.L, Pilino, P. (2019) Virtual reality: A new method to investigate congintive load during navigation, *Journal of Environmental Pshychology*, 65. Available at:

https://doi.org/10.1016/j.jenvp.2019.101338 (Accessed: 10 November 2022)

Rosch, J.L., & Vogel-Walcutt, J.J. (2013) A review of eye-tracking applications as tools for training, *Cognition, Technology & Work*, 15, 313-327. Available at:

https://doi.org/10.1007/s10111-012-0234-7 (Accessed: 10 November 2022)

Oguz, A., Omer, A. (2019) Use of Eye Tracking For Assessment of Electronic Navigation Competency in Maritime Training, *Journal of Eye Movement Research,* 12(3), Available at:

https://doi.org/10.16910/jemr.12.3.2 (Accessed: 20 November 2022)

Guenter, B., & Tripathi, S. (2017) A Statistical Approach to Continuous Self-Calibrating Eye Gaze Tracking for Head-Mounted Virtual Reality Systems, *IEEE Winter Conference on Applications of Computer Vision*, Santa Rosa, 24-31 March 2017, Santa Rosa: IEEE: 862-870. Available at:

https://ieeexplore.ieee.org/abstract/document/7926684/citations#citations (Accessed: 1 December 2022)

Hector, H. (2022) How to extend battery life on your Oculus Quest 2. *Techradar,* April 16. Available at:

https://www.techradar.com/how-to/how-to-extend-battery-life-on-your-oculus-quest-2 (Accessed: 2022)

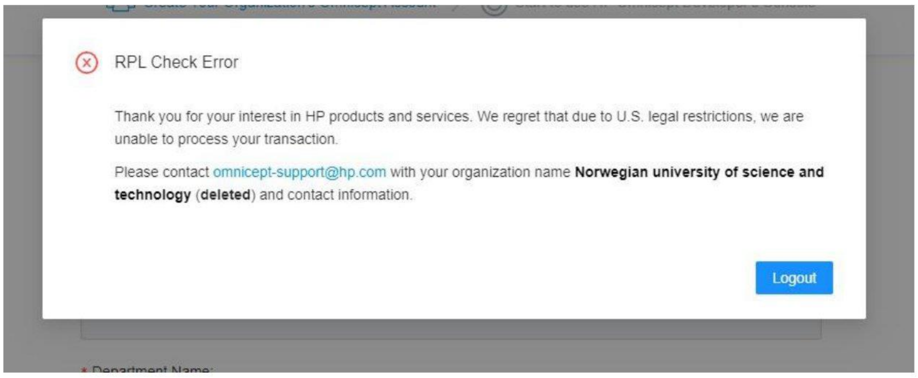
Sweller, J. (1988) Cognitive load During Problem Solving: Effects on Learning, *Cognitive load during problem solving: Effects on learning*, 12(2), 257-285. Available at:

https://doi.org/10.1016/0364-0213(88)90023-7 (Accessed: 20 November 2022)

Attachments

Statusrapport uke 43

Denne uken ble headsettet satt i fokus. Først prøvde vi å få taki rådata fra headsettet i seg selv. Dette ble gjort med en debug konsoll hvor man kan velge hvilke data man vil ha, problemet med dette er at dataene kommer så fort opp at vi ikke rekker å se dem. Vi vet heller ikke om dataene er i sanntid siden en konsoll er treg. Konsollens data ble også førsøkt tatt opp med OBS. Dette også viste at dataene bare flyr av gårde. Samtidig prøvde vi å få taki SDKen til headsettet igjennom HP sine nettsider. Dette var ikke hovedfokuset som vi så etter men det ble gjort. Etter hvert fikk vi beskjed om at organisasjonen som vi skulle bruke var NTNU. Når dette ble prøvd fikk vi opp denne feilemeldingen:


RPL Check Error

Thank you for your interest in HP products and services. We regret that due to U.S. legal restrictions, we are unable to process your transaction.

Please contact omnicept-support@hp.com with your organization name **Norwegian university of science and technology** (**deleted**) and contact information.

Dette førte til at vi droppet og få taki SDK'en til HP siden det bare var et «sidequest». Dermed fokuserte vi mer på Tobii XR sin dokumentasjon og eksempler. Etter møtet på Tirsdag 25.10.2022 sa Sveinung at han ønsket en kjapp demo på hva som var mulig innen eyetracking som kan vise til andre i Kystverket hvordan det fungerer. Dermed blir fokuset denne uken satt på å se litt på Unity og dokumentasjonen til Tobii på hvordan man gjøre det. Et verktøy som kan hjelpe oss mye heter Tobii Ocumen. Dette har blant annet recording muligheter som kan vise de på kystverket bedre hva vi kan gjøre med denne teknologien.

Tobii installasjon på unity: https://developer.tobii.com/xr/develop/unity/getting-started/omnicept/

Tobii ocumen : https://developer.tobii.com/xr/solutions/tobii-ocumen/

Statusrapport uke 44

I denne uken har vi fått en mail ifra Tobii om forespørselen på at vi ville ha en akademisk lisens. Siden de har et problem med nettsiden sin, måtte vi skrive grunnen igjen til hvorfor vi ønsket en lisens. Etter det har vi kontaktet veileder for å få malen til en bachelor oppgave så prosjektet kanskje kan videreføres.

Samtidig ble det ikke levert en statusrapport 27. oktober som ble skrevet i dag. Etter det ble det også lest en artikkel fra Sveinung som er industrikontrakt siden han sendte en lenke til sin google scholar profil. Etter det ble det også lest en artikkel på hvordan pancake og frensel linser fungerer pluss en state-of-the-art rapport om VR som ga inspirasjon til hvordan vi kan formulere våres.

Statusrapport uke 45

Denne uken ble brukt til å lese noen artikler og finne mer forskning på cognitive load feltet. Samtidig fremover blir det skrevet stikkord til rapporten samtidig som at forskning blir lest så man lettere kan fylle inn alt som skal i rapporten.

Siden vi har sett i flere rapporter at cognitive load er noe som kan måles med pupilometry som VR headset støtter ble det også lest en artikkel som handler om cognitive load og hvordan dette har sammenheng med visual fatiuge. Dette kan også påvirke lærngsutbytte siden man blir foretere sliten og ikke har tilgjenglig noe mer ressurser til læring. Samtidig etter det ble det lest et reivew av hvordan eyetracking kan bli brukt som et verktøy for læring. Denne fokuserte også på hvordan cognitive load har påvirkning på hvordan folk lærer.

Cognitive load

https://www.tandfonline.com/doi/full/10.1080/10447318.2021.1976509

Eyetracking as tools for interaction.

https://link.springer.com/article/10.1007/s10111-012-0234-7

Statusrapport uke 46, 47 og 48

Siden det har vært eksamens tid har vi summert opp arbeidet i en status rapport.

Mye av tiden har gått til å fokusere på hvordan vi skulle skrive rapporten og summere opp forskning som vi har lest. Samtidig måtte vi lese litt om igjen det som vi allerede hadde lest for å finne koblinger som vi kunne ta med i rapporten. Vi har også lest følgende forskning:

https://www.tandfonline.com/doi/pdf/10.1080/00140130701817062?needAccess=true

https://ieeexplore.ieee.org/abstract/document/7926684?casa_token=RKkl8goxN3oAAAAA:PPGHQwQZFdTKzUzhZqqLMu7itMU_nJSn7ISMvqd-YXanpy0IJ6LUfBYrifVEUgqbStb-x6pc5IU

https://reader.elsevier.com/reader/sd/pii/S0167876020301458?token=B36B755EC7DDE27EFB6162F31AF8A473A68529A72224F1B0759C8E2232B87433C4451AC6841E2C4303BFA8AA81086A4A&originRegion=eu-west-1&originCreation=20221205115128

https://www.tandfonline.com/doi/full/10.1080/10447318.2020.1778351

Statusrapport uke 49

Denne uken skal gå til å ferdigstille rapporten basert på det vi har lest og skrevet stikkord om. Hittil har vi et utkast som ble sendt til industrikontakten og han sa vi var godt på vei. Dermed gikk vi full guffe og ferdigstilte rapproten.

Repositories

VR Project:

https://github.com/Bacheloroppgave-Kystverket/Unity-VR-Eye-Tracking-Demo

Backend:

https://github.com/Bacheloroppgave-Kystverket/Backend

Frontend:

https://github.com/Bacheloroppgave-Kystverket/Frontend