Integrated Visual Field Testing and Remapping for Hemianopia using Virtual Reality

by Nik Harith Sharifuddin (2673209B)

WORLD CHANGING GLASGOW

A WORLD TOP 100 UNIVERSITY

Outline

- Introduction, motivations & aims
- Design
- Implementation
- User Evaluation
- Results & Discussion
- Conclusion

Introduction

Introduction

- **Hemianopia** is a cortical visual impairment that limits the visual field of a person to half of it, either to the right or left side
- Commonly caused by **stroke**, it reduces spatial awareness, making tasks such as navigation, reading and driving difficult.

Motivations:

- The devices used to assess hemianopia are often **expensive** and requires an expert.
- Common ways alleviate the effects of hemianopia uses methods such as Fresnel glasses are often **ineffective** in improving the condition in the long run.
- Virtual Reality (VR) technology, packed with **numerous features** such as eye-tracking and camera passthrough has become more **accessible** and have the potential to be utilised to help people with hemianopia.

University of Glasgow

Outline

Aims:

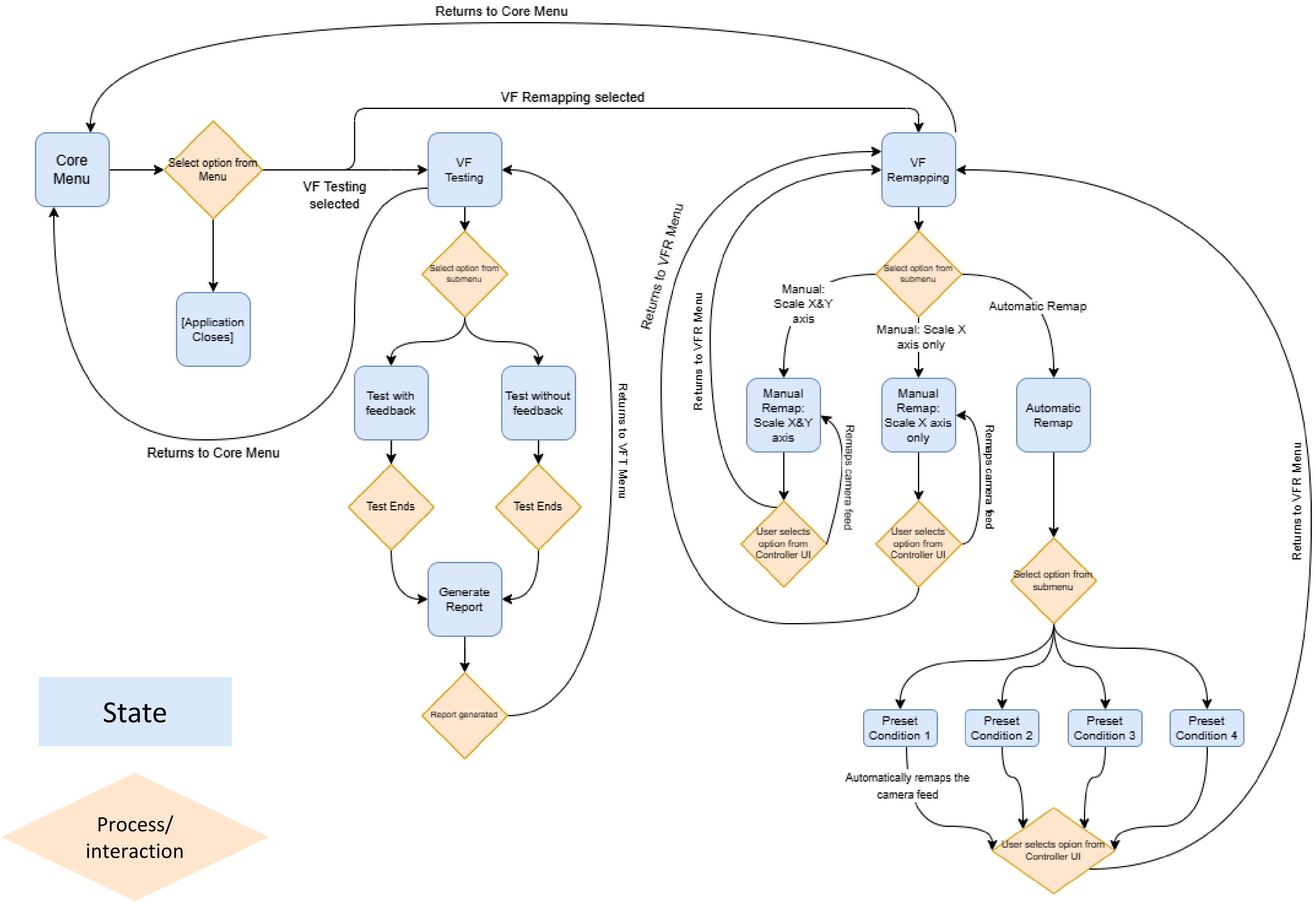
- This project continues the efforts of **previous students** developing the various ways to detect the visual impairment, specifically hemianopia, and aid people that has it using VR technology.
- Those projects however were developed separately and are **disjoint** from each other at the moment.
- There is a need to **integrate** them into a single application so that functionalities could be combined and complement each other's functionalities.
- The application would also be extensible, and new features may be added easily.

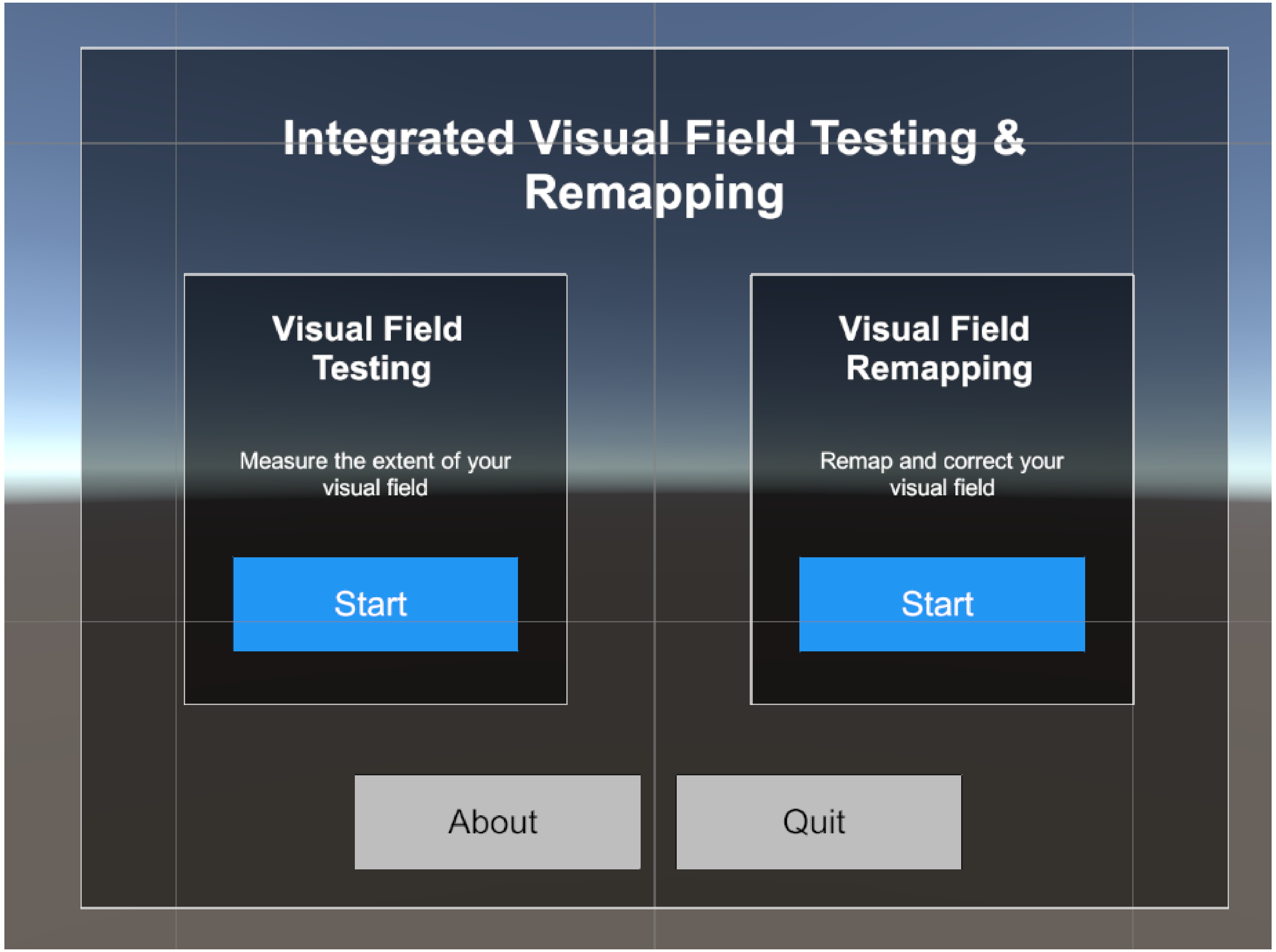
- This is a collaboration project, where I mostly focused on the integration part and Visual Field Remapping (VFR), while Caleb, worked on the Visual Field Testing (VFT) module.

Design

Application workflow

- A **common menu** (CORE menu) encompasses both modules we want to integrate.
- Both modules retains their functionalities from previous projects.
- Users can access both modules **without leaving the application**.





Common user interface

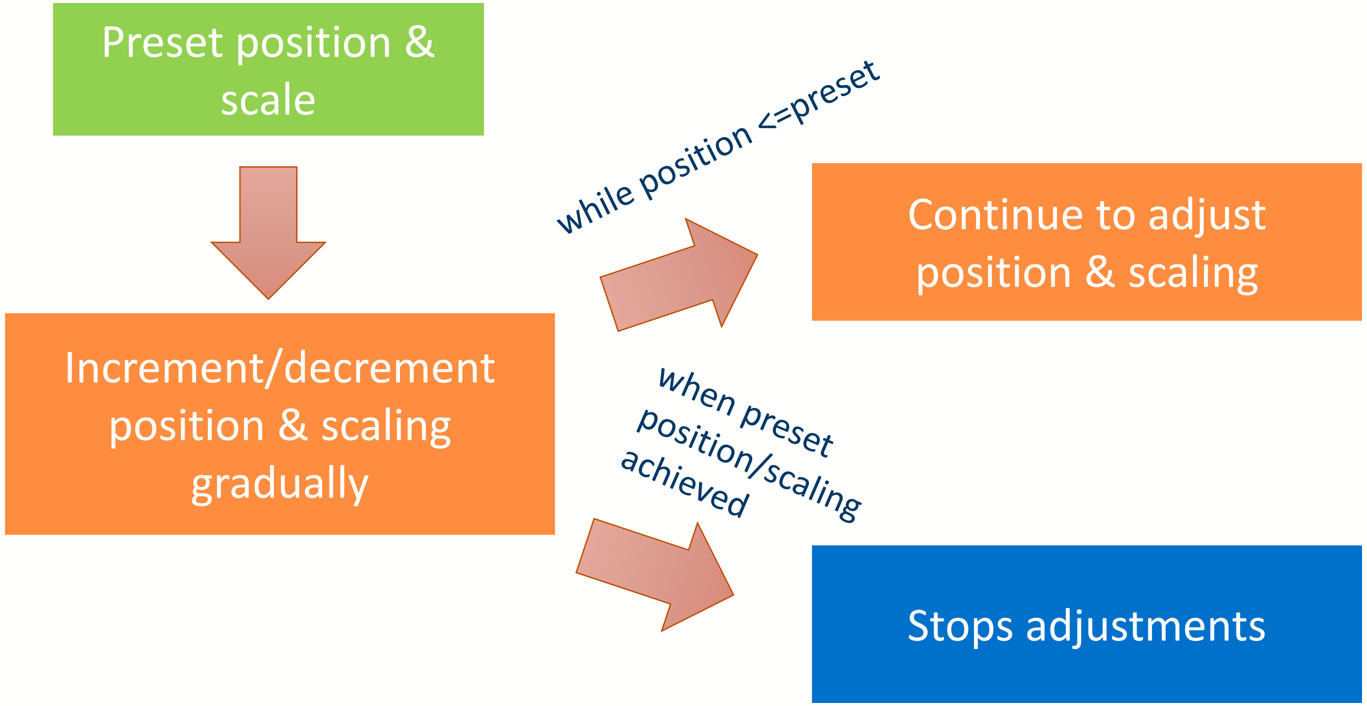
- Sample of common user interface to be used in the final application
- Used across all menu in the application

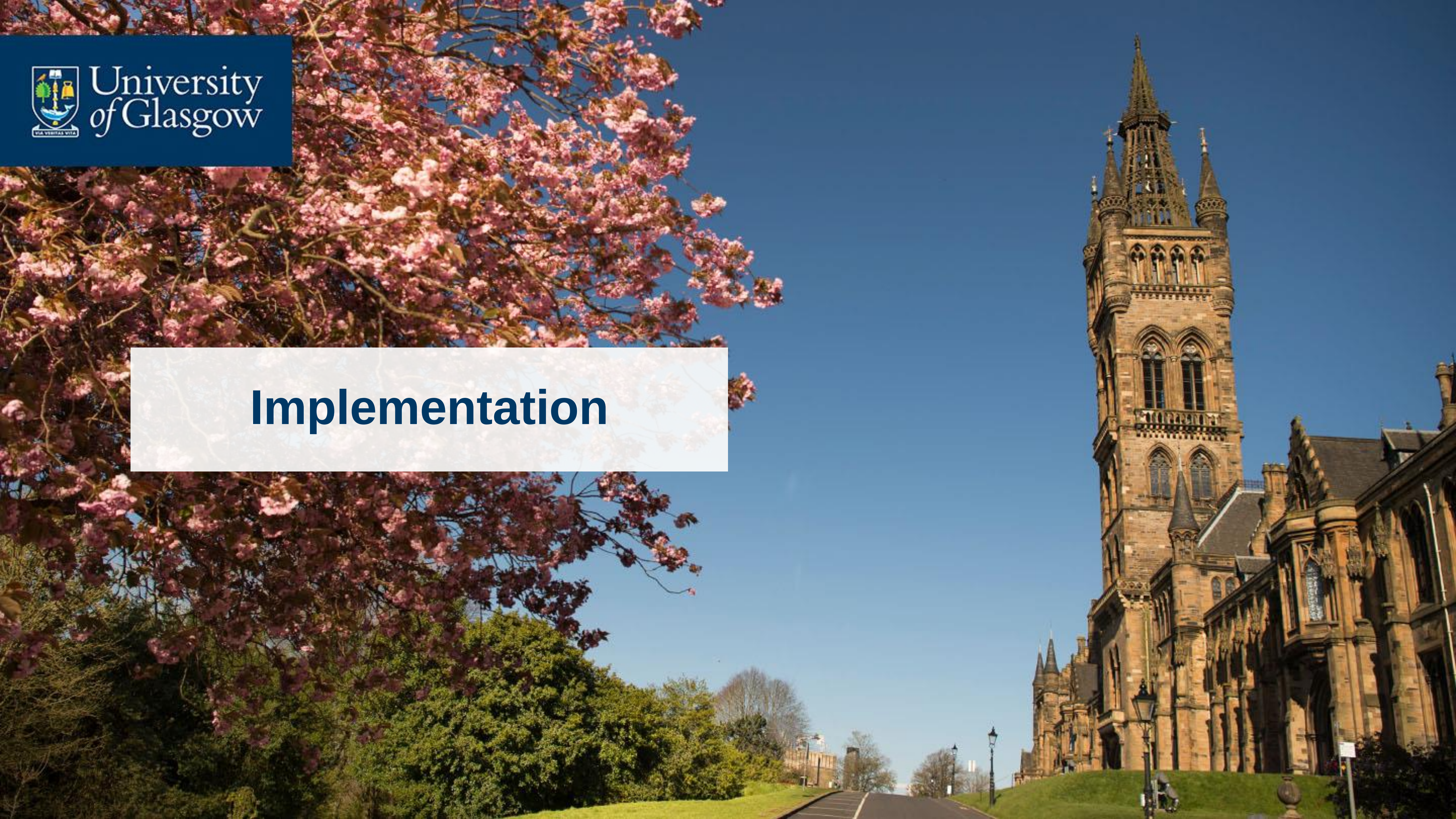
Automatic remapping

- A new feature implemented to **improvise** the process of remapping the live camera feed
- Intended to reduce cybersickness and increase accuracy and efficiency

```
Scale Adjustment;
while currentScale ≥ targetScale do
    currentScale ← Vector3(scaleFactor, scaleFactor, scaleFactor) * Time;
end
Position Adjustment;
while currentPosition ≥ targetPosition do
    currentPosition ← Vector3(position Factor, 0, 0);
end
```

Algorithm 1: Automatic remapping algorithm





Implementation

Hardware

- HTC Vive Pro Eye headset
- PC workstation:
  - Intel Core i9-12900K, 3.2GHz
  - 64GB of RAM
  - NVIDIA GeForce RTX 3080 Ti

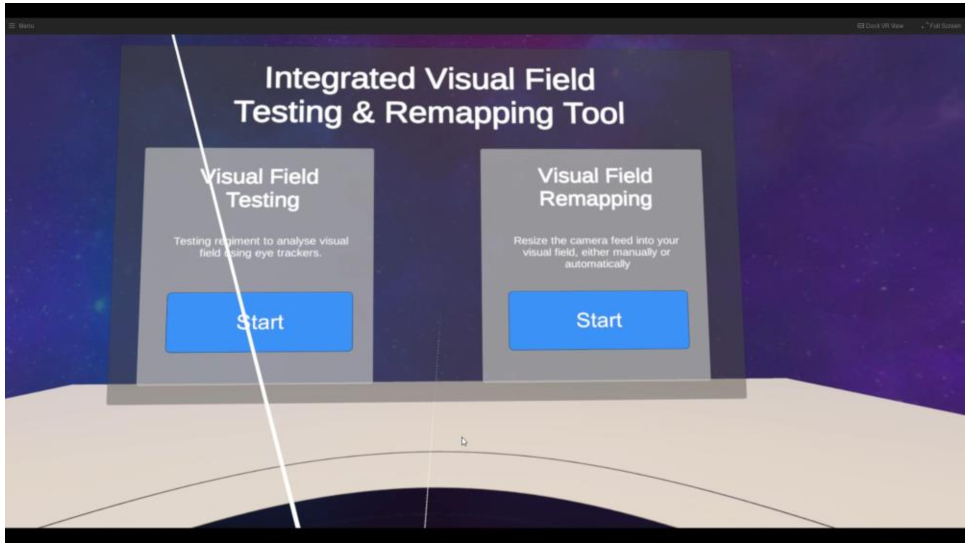
Software

Software we used:

- Unity v2019.4.35f1
- SteamVR
- SRWorks SDK
- SRAnipal
- Unity scripting was done in C#
- Parts of the VFT module uses Python for eye tracking data processing and reporting

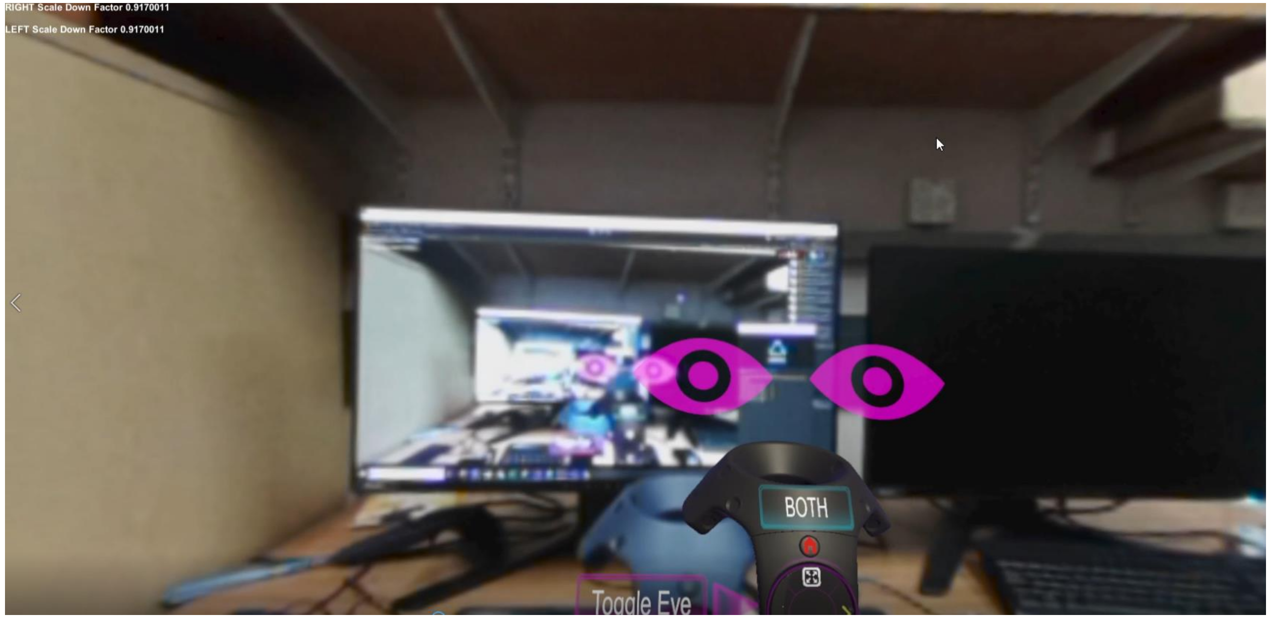
Unity

python™



- Application now integrates both VF Testing and Remapping modules
- Common user interface developed with consistent themes and colours across all menus

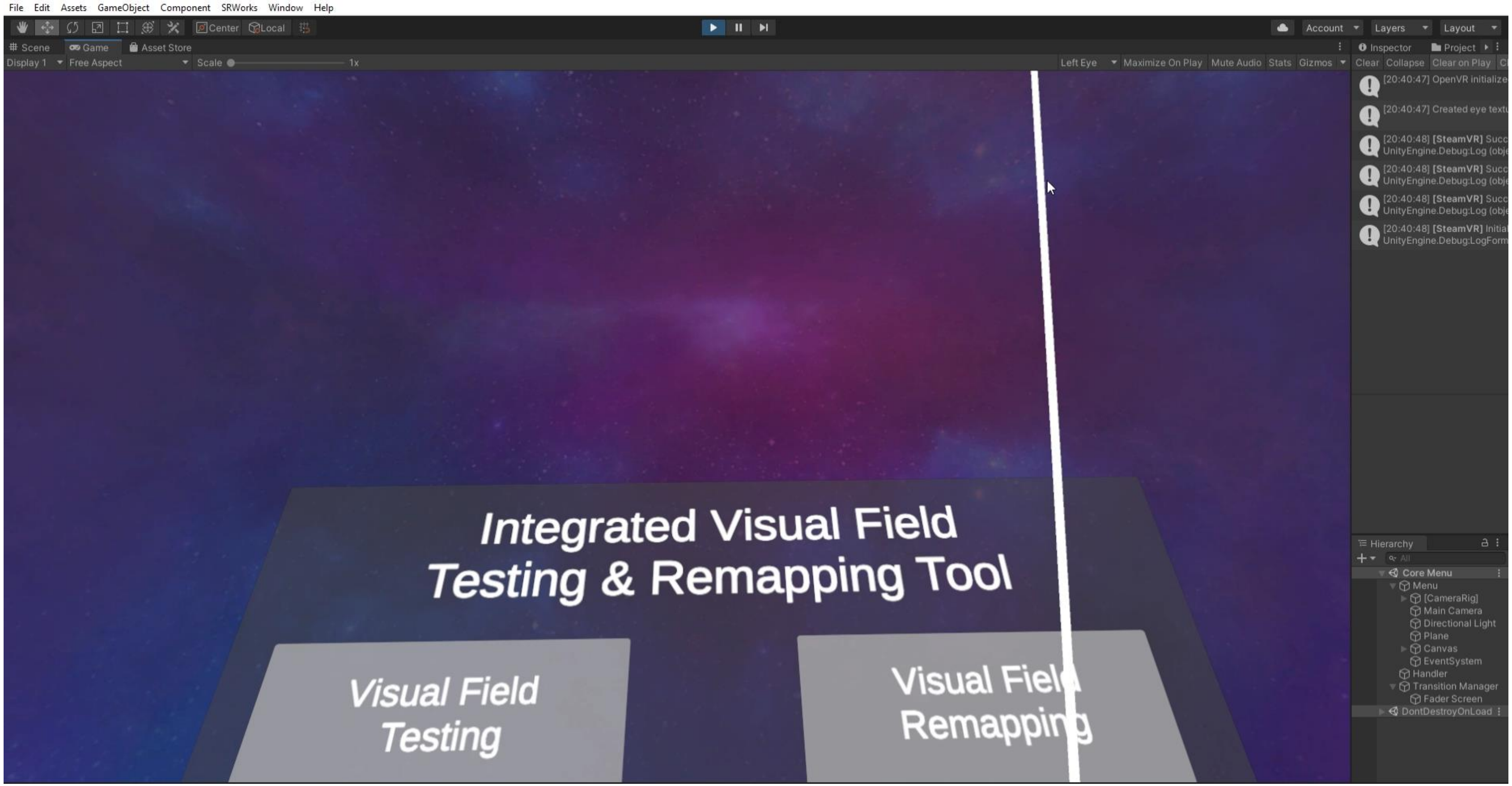
```
float scaleSpeed = 0.1f;
float posSpeed = 0.001f;

...
// Right-side Hemianopia autoremap
public void smoothHHRight() {
   // Run coroutine for both scaling and positioning
   StartCoroutine(scaleAdjustmentHHRight());
   StartCoroutine(posAdjustmentHHRight());
}

IEnumerator scaleAdjustmentHHRight() {
   while (leftEyeCamera.transform.localScale.x >= 0.6f) {
      leftEyeCamera.transform.localScale -= new Vector3(scaleSpeed,
          scaleSpeed,scaleSpeed) * Time.deltaTime;
      rightEyeCamera.transform.localScale -= new Vector3(scaleSpeed,
          scaleSpeed,scaleSpeed) * Time.deltaTime;

      yield return null;
   }
}

IEnumerator posAdjustmentHHRight() {
   while (leftEyeCamera.transform.localPosition.x >= targetPos) {

      Vector3 leftPos = leftEyeCamera.transform.localPosition;
      Vector3 rightPos = rightEyeCamera.transform.localPosition;

      // Subtract from the x component
      leftPos -= new Vector3(posSpeed, 0f, 0f);
      rightPos -= new Vector3(posSpeed, 0f, 0f);

      // Assign the updated position back
      leftEyeCamera.transform.localPosition = leftPos;
      rightEyeCamera.transform.localPosition = rightPos;

      yield return null;
   }
}
```



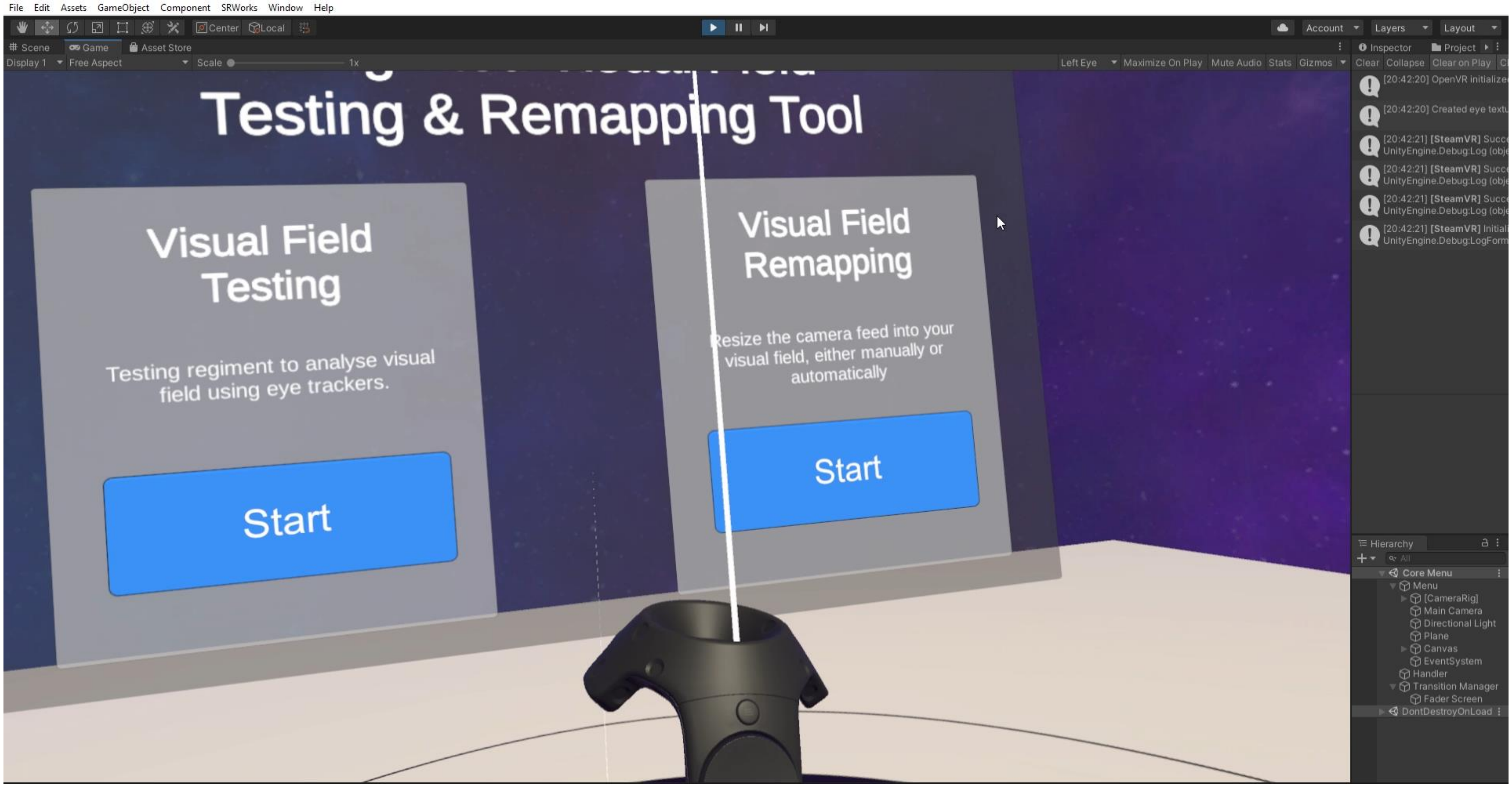
- Automatic remapping was achieved by using presets to transform the position and scale of the live camera passthrough
- We defined presets of position and location for certain hemianopia conditions;

- Homonymous hemianopia, Right
- Homonymous hemianopia, Left
- Quadrantanopia, Right Temporal
- Quadrantanopia, Left Parietal

Application Demo



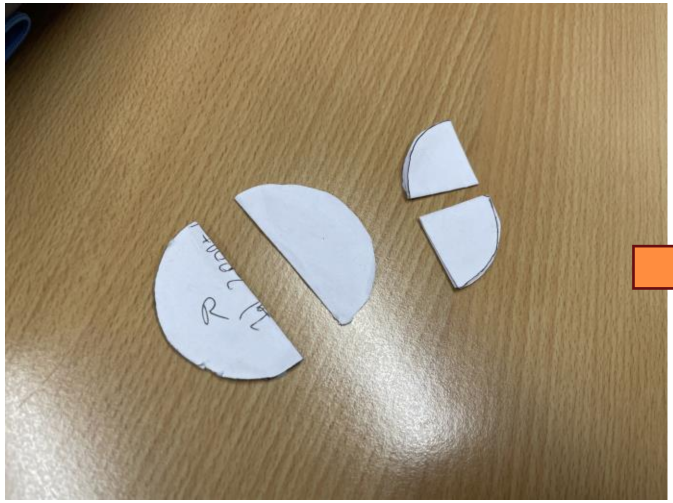
VF Testing Demo



VF Remapping Demo

Evaluation & Results

- The evaluation chose to focus on analysing the **effectiveness of automatic remapping**, while also assessing the usability of the application.
- Participants were simulated with **partial blindness** and were asked to **identify** some objects on a table in camera passthrough mode.
- Then, they would try to **remap their blocked vision** in manual mode, or they would select the **automatic remap option** so they could see all objects on the table.
- We assessed 4 conditions:
  - Homonymous hemianopia, Right
  - Homonymous hemianopia, Left
  - Quadrantanopia, Right Temporal
  - Quadrantanopia, Left Parietal
- A survey consisting of **SSQ** and **NASA-TLX** questionnaires were filled in after each conditions

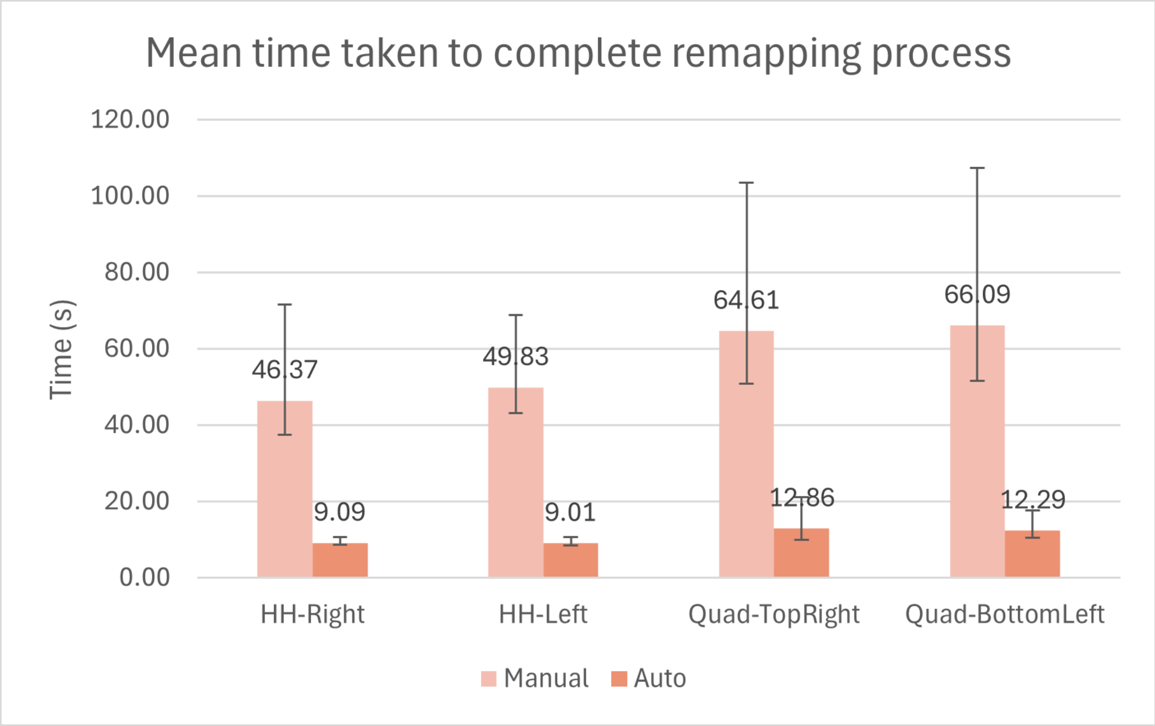


Hypothesis:

*"Automatic remapping results in better **efficiency**, **accuracy** and **comfort** compared to manual remapping"*

- **Efficiency**: measured by **time taken** to remap in each of the two modes.
- **Accuracy**: measured by difference between the **final remapped position** and **scaling** of the live camera passthrough between both modes.
- **Comfort**: measured by **SSQ** and **NASA-TLX** scores, with lower the scores indicate better results.

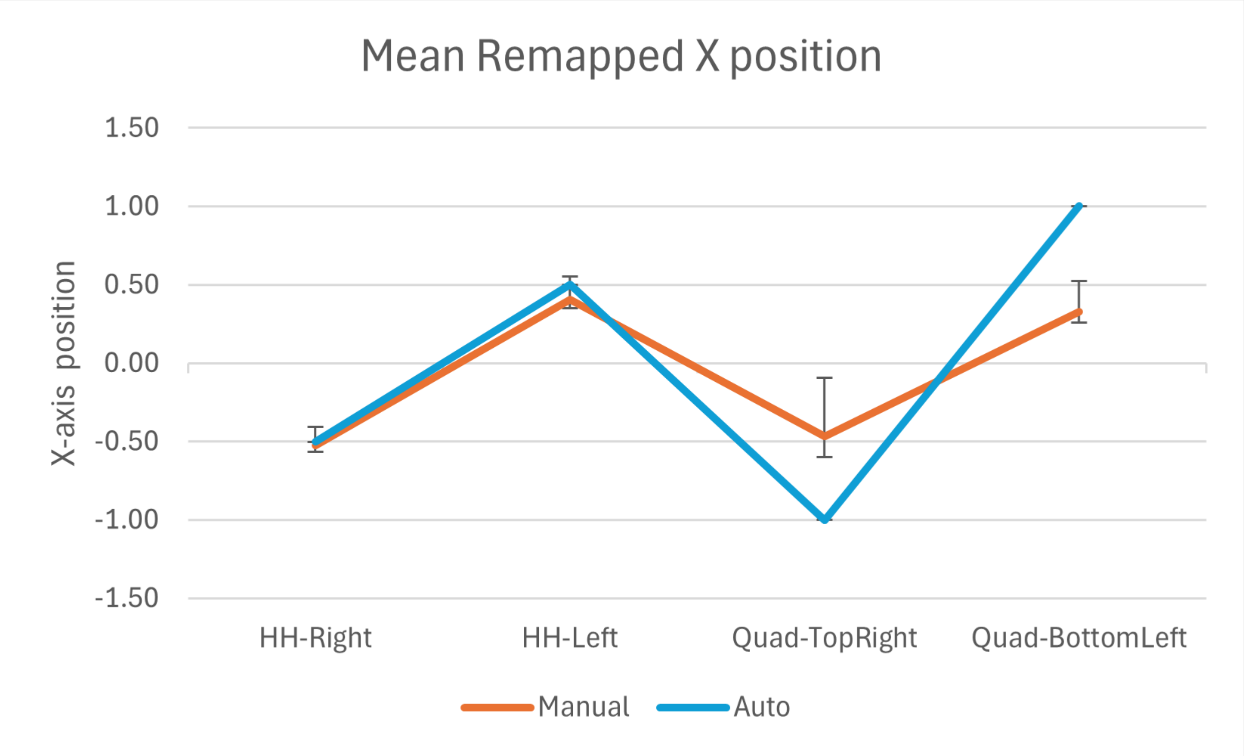
Results: **Mean time taken to remap**



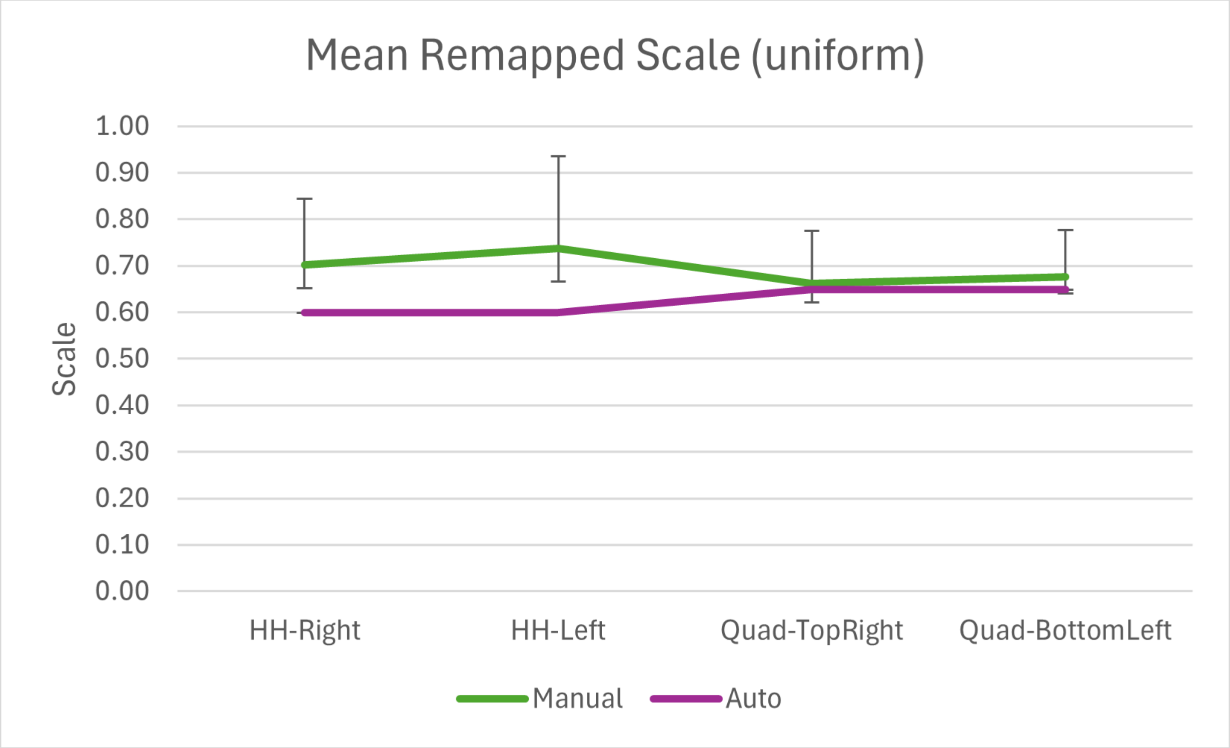
- Mean time taken to remap shows that automatic remapping outperforms manual remapping, mainly due to its fixed preset nature.
- In both modes, quadrantanopia conditions takes longer to remap. This may be caused by the asymmetrical shape of the simulated blindness required the user to spend more time finding the right position/scaling.

Results: **Mean remapped X-axis position**

- Mean final remapped position (taken with X-axis) reveals that there are little differences between both remapping modes for homonymous hemianopia conditions, supporting our hypothesis.
- Quadrantanopia conditions have big differences (~ ± 0.5) between the two modes, suggesting unsuitability of the preset settings.

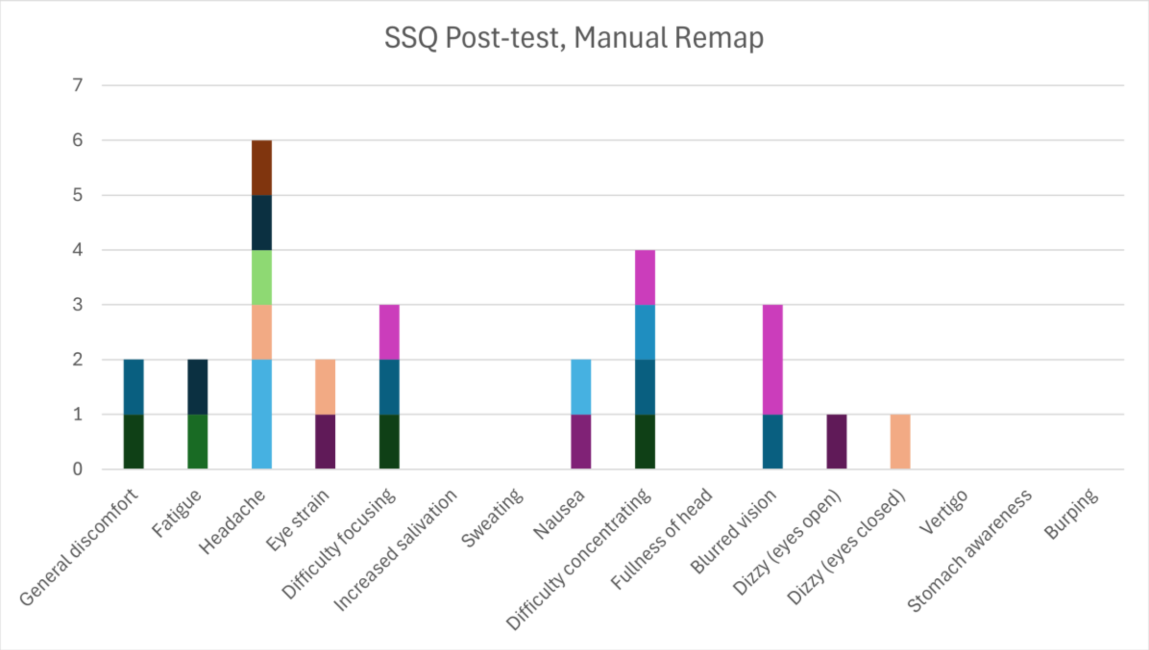


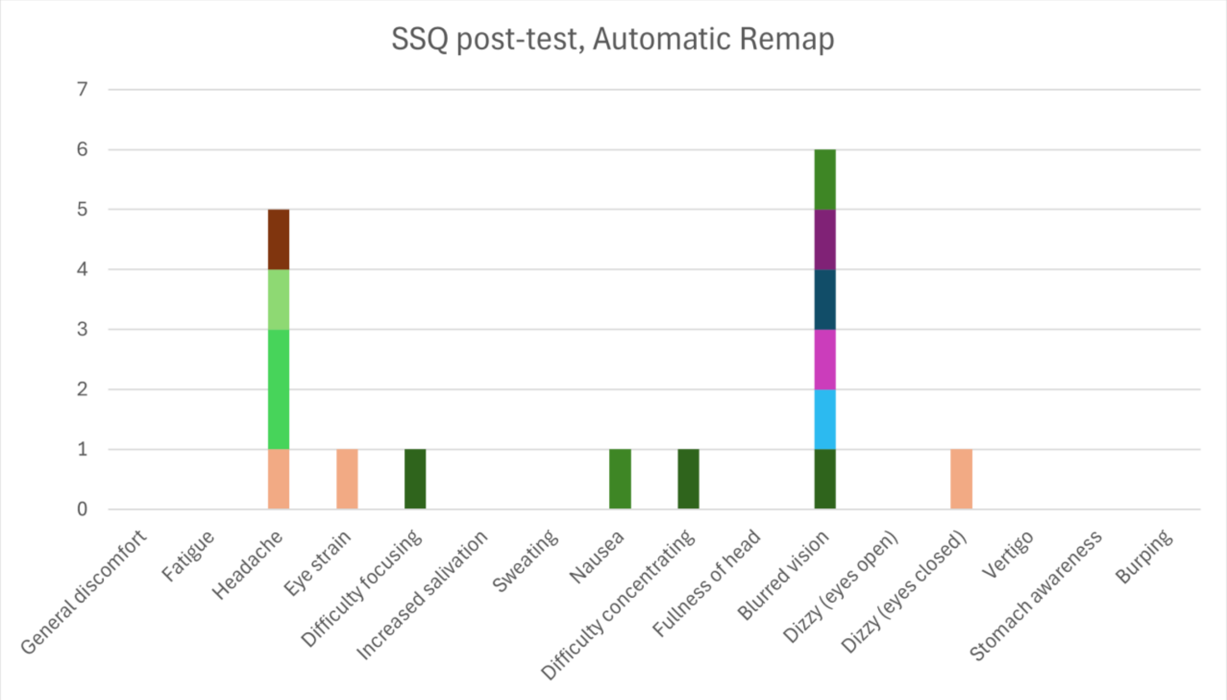
Results: **Mean final remapped scale**



- Mean final remapped scale (uniform on X & Y axis) reveals the inverse of the previous remapped final position results.
- Here, the differences between both remapping modes for quadrantanopia conditions are miniscule, supporting our hypothesis.
- Homonymous hemianopia conditions this time have differences around 0.10 between the two modes, suggesting the preset settings needs more improvement to adapt to these conditions.

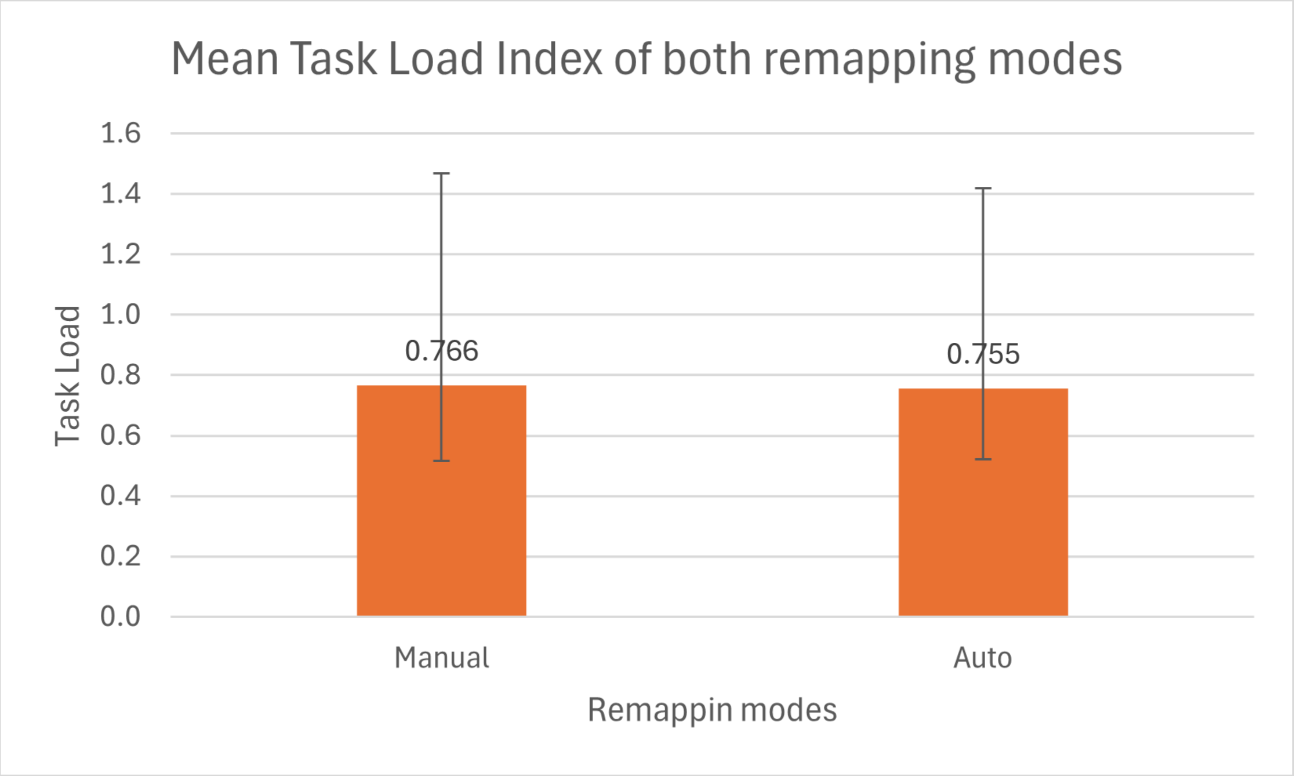
Results: **SSQ**





- There are less instances of simulator sickness reported after the automatic remapping was performed compared to manual remapping mode.
- Possibly due to the simpler process of automatic remapping requiring participants to exert less effort to complete the task.

Results: **NASA-TLX**



- The mean task load for both remapping modes are not that different, meaning the perceived workload for both modes are the same.
- This suggests that participants experienced the same amount of perceived comfort between both modes.

Results: **General Usability**

- **Half** of the participants ***(n=8)*** prefers automatic remapping while the other half prefers manual remapping, a perfect half split.
- Participants felt the program was **user friendly and intuitive**. They also felt it would be useful for people with visual impairment.
- Some participants felt there are still works to be done to improve the user interface, citing the need to adapt to the general public *especially* people with visual impairment.

Limitations

- Sample size is only *(n=8)* participants, limiting the validity and generalisability of the results experiment.
- Some parts of the experiment were difficult to control despite best efforts made to control them, such as the viewing angle of the participant while seated during the experiment due to different heights of the participant.

University of Glasgow

Conclusion

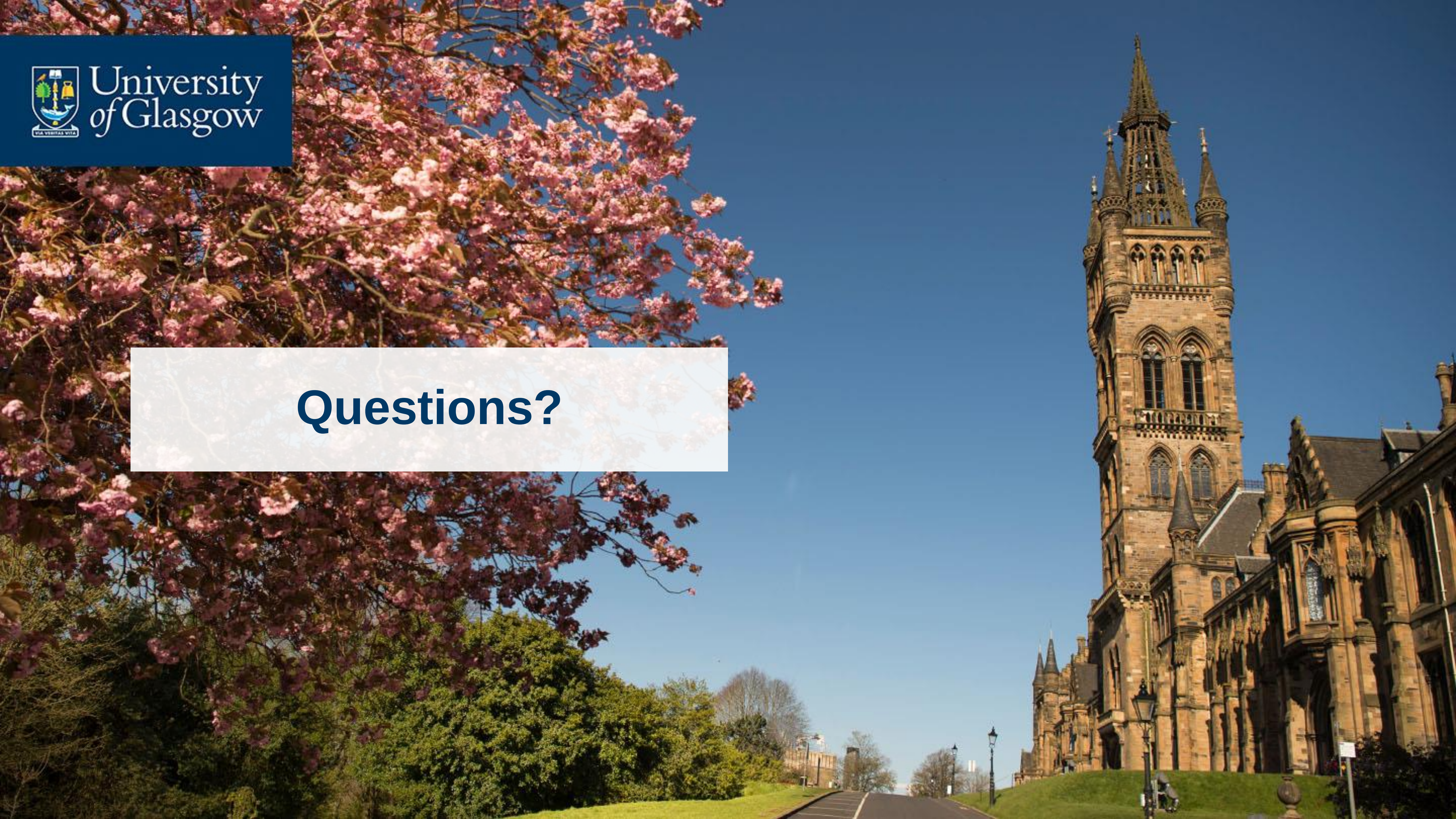
Conclusion

- This project achieved the goal to **integrate** both of the Visual Field Testing and Remapping modules into one single VR application.
- This will simplify and streamline the process of adding new feature in future development.
- A new feature to improve the comfort of remapping the visual field was also developed.
- Results from user evaluation reveals that it is more efficient in terms of the time needed to do the remapping process
- However, further improvements are needed for the automatic remapping to be fully useful for those with Hemianopia.

Future works

- **Automatic remapping with VFT Data**
  - Personalised remapping procedure, rather than preset.
- **Porting to the latest version of Unity**
  - Use the latest version to access latest features and conform to latest standards
- **Portability of HMD**
  - Use newer and lighter HMDs for the application, increasing comfort.
- **Adaptive remapping**
  - Use eye-tracker to responsively adjust the remapped vision to where the user is seeing, rather than static

University of Glasgow



Questions?

Thank you!

I enjoyed working on this project and putting my effort into it. ☺

Nik Harith
2673209B