UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Renee Kroon

Application for Psychophysics Experiments in Virtual Reality

Bachelor's Thesis (9 ECTS)

Supervisors: Jaan Aru, PhD
Madis Vasser, MSc

Tartu 2019

Application for Psychophysics Experiments in Virtual Reality

Abstract:

The goal of this thesis is to develop an application that allows running psychological experiments in virtual reality or in a first-person perspective on a normal computer monitor. Taking inspiration from Psychlab, the application is meant to be used to bring human test subjects and artificial intelligence agents to a more similar experimenting environment, or comparing the results of the same experiment done on humans with and without virtual reality. The application is created using Unity and four classic psychological experiments are implemented. The thesis gives an overview of how the application functions, and gives background information about similar programs and implemented experiments. A pilot study is conducted to validate the usability of the application. Possible improvements, like adding machine learning support, are discussed.

Keywords: virtual reality, three-dimensional graphics, Unity, psychophysics, cognition, vision

CERCS: P175, Informatics, systems theory

Rakendus psühholoogiliste katsete läbiviimiseks virtuaalreaalsuses

Lühikokkuvõte: Bakalaureusetöö eesmärk on luua rakendus, mis võimaldab viia läbi psühholoogilisi katseid virtuaalreaalsuses või esimeses-isikus perspektiivis tavalistel arvutimonitoridel. Loodud rakenduse inspiratsiooniks on Psychlab. Rakendus võimaldab teha katseid inimeste ja tehisintellektiga varasemast sarnasemas katsetuskeskkonnas, ning võrrelda inim-katseisikuid virtuaalreaalsuses ja ilma virtuaalreaalsuseta timgimustes. Rakendus on loodud kasutades Unity mängumootorit ja sisaldab nelja klassikalist psühholoogia eksperimenti. Bakalaureusetöö annab ülevaate sellest kuidas rakendus töötab ning taustainfot sarnaste programmide ja implementeeritud katsete kohta. Rakenduse kasutatavuse valideerimiseks on tehtud prooviuuring. Arutatakse võimalikke edasiarendusvõimalusi, näiteks loodud rakendusele masinõppe toe lisamine.

Võtmesõnad: Virtuaalreaalsus, ruumiline graafika, Unity, psühhofüüsika, tunnetus, nägemine

CERCS: P175, Informaatika, süsteemiteooria

Table of Contents

Introduction

Machine learning based artificial intelligence has seen a lot of attention in recent years. From self-driving cars to mobile apps, many companies and individuals have been trying out different real-world uses for artificial intelligence [1]. Thanks to a multitude of freely available software and libraries[1], it is possible for any individual to make use of the latest technologies when working with artificial intelligence.

Complex neural networks that are often used in artificial intelligence and machine learning are also a simplified model of the human brain. This means that observing artificial intelligence could improve the understanding of how the human brain works. On the other hand, discoveries from the field of neuroscience could help to improve artificial intelligence development. A lot about how the brain functions has been learned through different psychological experiments, but very few of these experiments have been attempted with artificial intelligence.

Similarly to artificial intelligence, virtual reality has become more widespread in recent years. Although the prices of virtual reality headsets have dropped slightly, virtual reality is still something most people don't have access to. Doing psychological experiments in virtual reality could yield slightly different results compared to doing experiments with study subjects sitting in front of a computer screen in a laboratory.

The goal of this thesis is to develop an application that allows running classical psychological experiments in both virtual reality and on normal computer monitors in a first-person perspective, as well as have a viable way to integrate machine learning into the application so that the artificial intelligence agents could perform the exact same experiments. It should also be usable by people with little computer science background.

The program is largely influenced by Psychlab [2]. Psychlab allows both humans and artificial intelligence agents to perform the same psychological experiments, and compare

[1] https://en.wikipedia.org/wiki/Machine_learning#Software

them to each other. In Psychlab experiments are done on a computer monitor using a computer mouse. If the human test subject would use virtual reality for the experiment he or she would be in a more similar environment to the artificial intelligence agent that is always immersed in a virtual environment, and it would also remove potentially distracting real-world variables. Due to the old engine of Psychlab however, it was determined that a new modern environment should be developed. Additionally four experiments were implemented for the newly created application and some testing was done to assure that usable data can be gathered using the application.

The thesis consists of four chapters. The first chapter introduces similar programs and discusses their advantages and disadvantages. The second chapter gives an overview of the technologies that were used to develop the new application. The third chapter explains how the created application works, its main features, what the process of running an experiment is like and how new experiments can be added. This chapter also describes the four experiments that were implemented as a part of this thesis. The fourth chapter describes the testing that was done with the created application, analyzes its results, and describes what improvements could be made to the application.

1 Similar Programs

Many programs exist that allow running psychological experiments with human test subjects. However almost none of these have integrated support for artificial intelligence to learn and perform the same psychological experiments. Most of the experiments also consist of showing 2D images to the test subject, putting the artificial intelligence in a slightly different environment because it is unable to look around like a human test subject in the laboratory could, and is able to send responses without the need for an external interface like a computer keyboard or a mouse. Two more popular existing applications were analyzed to assess if it would be feasible to simply add virtual reality support or integrate machine learning into them.

1.1 Psychlab

Psychlab is a 3D first-person cognitive psychology and psychophysics testing environment developed bt DeepMind [2]. It can be used to run psychological experiments on both humans and machine learning agents [2]. In Psychlab the player character stands in front of a virtual screen surrounded by walls (see figure 1). The player can look around the environment but cannot move from the place in front of the screen. Different experiments, referred to as tasks, can be run by showing different images and buttons on the virtual screen [2]. The test subject is expected to react to shown images by moving the mouse so that the player character is looking directly at a specific button on the virtual screen (see figure 1). Psychlab has multiple example tasks. New tasks can be added through an API and are written in lua scripting language [2].

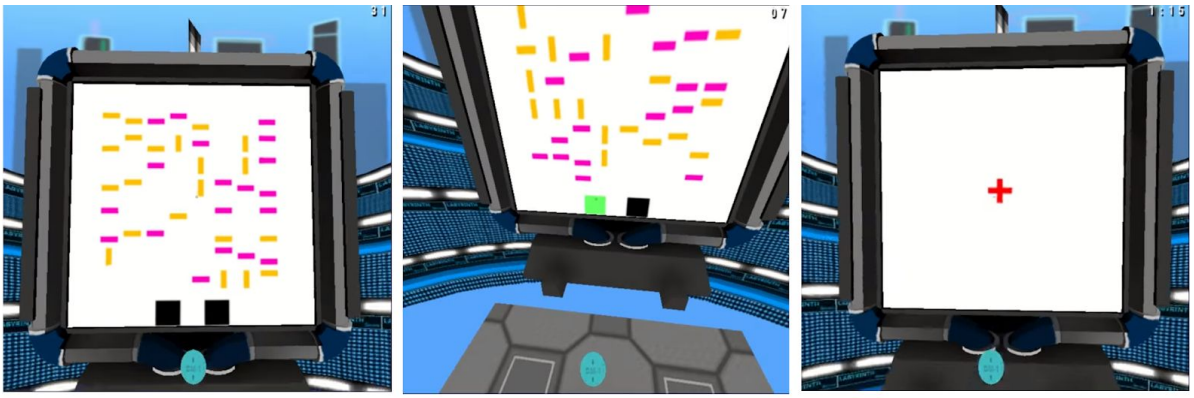


Figure 1. Screenshots taken during a Psychlab task: what the virtual screen looks like when an image is shown (left), what looking at a button on the virtual screen looks like (middle), and the screen with the red cross that starts the trial on it (right) [2]

Psychlab and its source code was made available to the public in early 2018 [2]. Psychlab is built upon DeepMind Lab, a platform made by DeepMind for researching and developing artificial intelligence behaviour in 3D worlds [3]. DeepMind Lab was released in 2016 and runs on a slightly modified ioquake3 engine [3]. Machine learning agents in DeepMind Lab can learn to navigate a 3D environment based only on visual input and optionally rewards [3].

Psychlab allows both humans and machine learning agents to learn and perform different psychological experiments. Deep reinforcement learning is integrated in DeepMind Lab and is used for the machine learning agents in Psychlab [2]. The machine learning agents perform the tasks by looking at buttons on the virtual screen in the same way as humans would.

Because humans and machine learning agents are in a somewhat similar environment when doing Psychlab tasks, it is also possible to directly compare them to each other. For better comparison the machine learning agent and human test subject could be put into the same environment. Virtual reality could be used to make the human test subject see the same that the machine learning agent does, and would make the gap between what the two experience smaller. It would also make responding to shown images more similar, as the human test subject would no longer have to use the mouse to look at a button and could look at it by simply moving their head.

1.2 PEBL

PEBL (Psychology Experiment Building Language) is an open source cross-platform software for creating and running psychological experiments [4]. PEBL comes with a test battery[2] of over 90 experiments. New experiments can be created using a specially made programming language [4]. PEBL has been widely used in different publications and theses[3]. PEBL was first released in 2003, the test battery was initially released in 2006 [4]. Both the program itself and the test battery have been improved multiple times since their initial releases.

PEBL experiments are all two dimensional, PEBL does not have support for creating three dimensional virtual objects and using them in experiments. PEBL also has a complex GUI which allows to navigate the test battery, set options for the program and configure parameters for chosen experiment (see figure 2). When an experiment is launched, at first instruction text is shown. For some experiments the instructions are followed by examples.

[2] http://pebl.sourceforge.net/wiki/index.php?title=PEBL_Test_Battery
[3] http://pebl.sourceforge.net/wiki/index.php/Publications_citing_PEBL

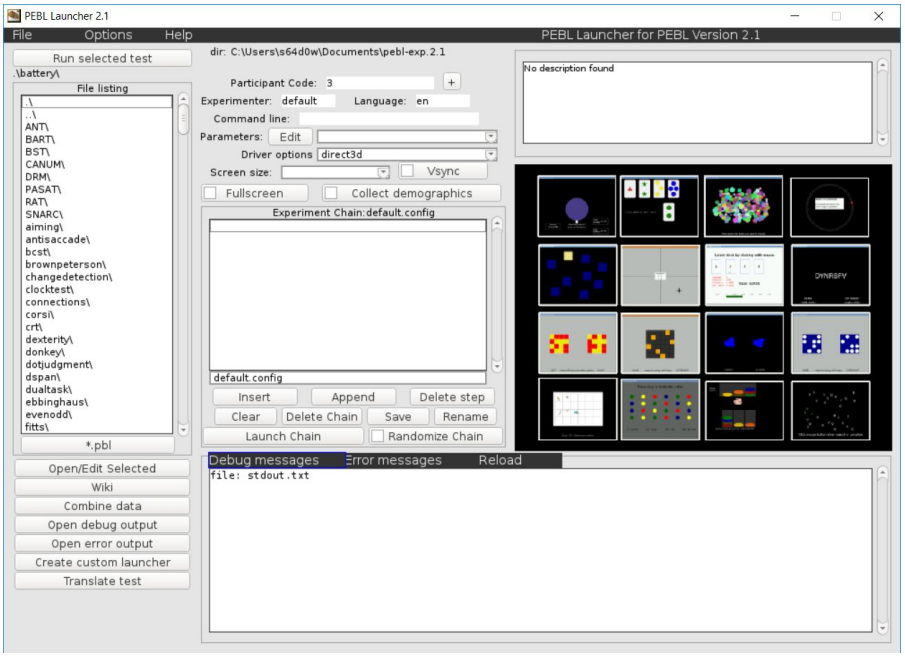


Figure 2. Screenshot of the PEBL Launcher version 2.1 [4]. The test battery can be seen on the left. Program options can be specified in the middle. Previews of some experiments (or the chosen experiment) can be seen on the right.

1.3 Problems with Similar Programs

One way to make it possible to do Psychlab tasks in virtual reality would have been to add virtual reality support to Psychlab's engine. Psychlab runs on ioquake3[4] engine that is an actively maintained and improved version of Quake III Arena engine's source code[5] [5]. Quake III Arena[6] was released in 1999 and as such its engine, being nearly two decades old, has many graphical limitations [6]. If virtual reality support was added to the engine, these limitations could ultimately lead to an uncomfortable or unrealistic virtual reality experience. Compiling and running Psychlab is also a significantly more difficult task than compiling or running an application created in Unity, furthermore Psychlab does not have a menu, so for changing tasks or different parameters, launch options have to be used. Since one of the goals of this thesis was to create a program that would be easily usable even by experimenters with little computer science background, something more simple to use than what Psychlab currently offers would have been desirable.

[4] https://github.com/ioquake/ioq3/
[5] https://github.com/id-Software/Quake-III-Arena
[6] https://store.steampowered.com/app/2200/Quake_III_Arena/

In addition to built-in virtual reality support, Unity also provides a plugin[7] which allows machine learning agents to be trained and observed in environments created in Unity. These agents can also process visual data, making them as capable as machine learning agents in DeepMind Lab [6].

For these reasons the option of re-creating the Psychlab testing environment and some of the example tasks in Unity was chosen. The new testing environment created in Unity is similar to the testing environment in Psychlab but still has slight differences due to the different game engine used and differences in object sizes and placements. In addition, the experiments that were re-created were not made exactly as they were in Psychlab. Due to these differences results of a task in Psychlab may differ from the results of a replicated task in the application created in Unity. The created application can still be used to observe and compare differences of completing tasks in virtual reality and completing tasks using a computer mouse and a monitor.

Because PEBL has been built for running psychological experiments with human participants, adding support for machine learning might prove to be a difficult task. Since the experiments can only be two dimensional, virtual reality support for PEBL would not be practical. Because of these reasons PEBL was not used as an example for the created application.

[7] https://github.com/Unity-Technologies/ml-agents

2 Used Technologies

2.1 Unity

The application was developed using Unity version 2019.3.0a6[8]. Unity is a multi-platform game engine that can be used to create 2D and 3D applications, animations or other projects [7]. A 3D or 2D scene in Unity consists of GameObjects that can have different components attached to them which give them functionality [8]. The behaviour of GameObjects can be fully customized using scripts that are written in the C# programming language [9]. Unity editor makes developing applications easier by providing a GUI to build and debug 3D and 2D scenes with, and allowing to run an application immediately after modifying it, without having to compile it separately [10].

Unity also has built-in support for virtual reality, but for a virtual reality device to work appropriate runtimes still have to be installed on the computer [11]. Unity is one of the most popular platforms for creating virtual reality and augmented reality applications, claiming that 60% of all such content is created using Unity [7].

2.2 Unity Assets

Unity Asset Store[9] gives its users the opportunity to upload or download different asset packs designed to be used with Unity. For this application two asset packs were used: the Standard Assets asset pack and the Oculus Integration asset pack. The asset packs were used to simplify complex input handling and making sure it works without bugs and is performance friendly. Both asset packs can be downloaded for free from the Unity Asset Store

The Standard Assets[10] asset pack contains multiple different character controller scripts that can be used in Unity. The asset pack is published by Unity Technologies. For this application the FPSContoller was used from the asset pack. The FPSController script was slightly

[8] https://unity3d.com/unity/alpha/2019.3.0a6
[9] https://assetstore.unity.com/
[10] https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-32351

modified to repurpose moving around using movement keys (arrow keys and WASD keys) to adjusting distance from the virtual screen instead. A lot of files and scripts from Standard Assets that were unnecessary for the created application were not included in the final build.

The Oculus Integration[11] asset pack, published by Oculus, contains multiple scripts and some resources that enhance Unity's built-in virtual reality support for Oculus Rift. For this application the OVRPlayerContoller was used from the asset pack. Like the FPSController, slight modifications were also made to the OVRPlayerContoller to prevent free movement around the scene and restrict the movement keys to only adjust the distance from the virtual screen. A lot of content in this asset pack was also not included in the application because it was not necessary.

2.3 Oculus Rift

Oculus Rift Consumer Version 1 (CV1) is a virtual reality headset released in 2016. It has two 2160x1200 resolution displays that run at 90Hz, meaning that rendering games in the Rift's headset takes about three times as much graphics processing power as rendering games on a full HD 60Hz monitor [12]. Because of this, a more powerful computer with a dedicated GPU is required to use Oculus Rift [12]. Maintaining a stable frame rate is very important in virtual reality as frame drops can cause a lot of discomfort in users [13]. This means that sometimes expensive graphical effects have to be toned down or removed to meet the frame rate threshold.

Oculus Rift works with Unity by default, and to make developing applications with the Rift easier Oculus recommends using the Oculus Integration asset pack [14]. The Rift's positional tracking system was used for the created application to enable looking around and selecting response buttons. Oculus also produces immersive controllers called Oculus Touch, which allow people wearing the Rift to use their hands in virtual reality [15], but these were not used for the created application.

[11] https://assetstore.unity.com/packages/tools/integration/oculus-integration-82022

Oculus Rift CV1 was discontinued in 2019, succeeded by Oculus Rift S, but it is still one of the most popular VR headsets, is supported by Oculus and can be used with the same applications that Oculus Rift S can be used with [16].

3 Application Overview

The created application has the ability to run simple psychological experiments. Experiments can be run either in first-person perspective on a computer monitor or in immersive virtual reality. As the goal of this application is to re-create Psychlab's testing environment, most of the features, including the experiments, are replicated from Psychlab. "Tasks" in Psychlab are referred to as "experiments" in the created application.

The application can be launched through Unity editor or it can be compiled and launched as a standalone program. When launching through Unity editor and using Oculus Rift for virtual reality there can be issues with the virtual reality headset disconnecting from the application due to the way Oculus runtime is set up.

3.1 Main Menu

When the application is launched, first the main menu is shown (see figure 3). In the main menu it is possible to choose between different options and set necessary parameters for the environment the experiment will take place in and the experiment itself. The options that can always be modified are the experiment that will be launched, whether virtual reality will be enabled, whether to use the UI timer or the in-world one (see figure 6), whether to save experiment results to a file, and whether the experiment will be limited by time or by the amount of trials completed and set the limit.

Depending on the experiment currently selected from the "Experiment" dropdown, more options are shown that affect the specific experiment only. These options will be changed every time a new value is selected from the "Experiment" dropdown. When the "Start" button is pressed the chosen experiment is started by loading either the VRRoom or FirstPersonRoom scene. Alternatively pressing either "enter" or "space" will also start the experiment, but can cause issues when certain input fields or checkboxes are active. It is possible to return to the main menu at any time when an experiment is active by pressing the

"escape" key. However this will cancel the experiment that was running and will not save any results to log files.

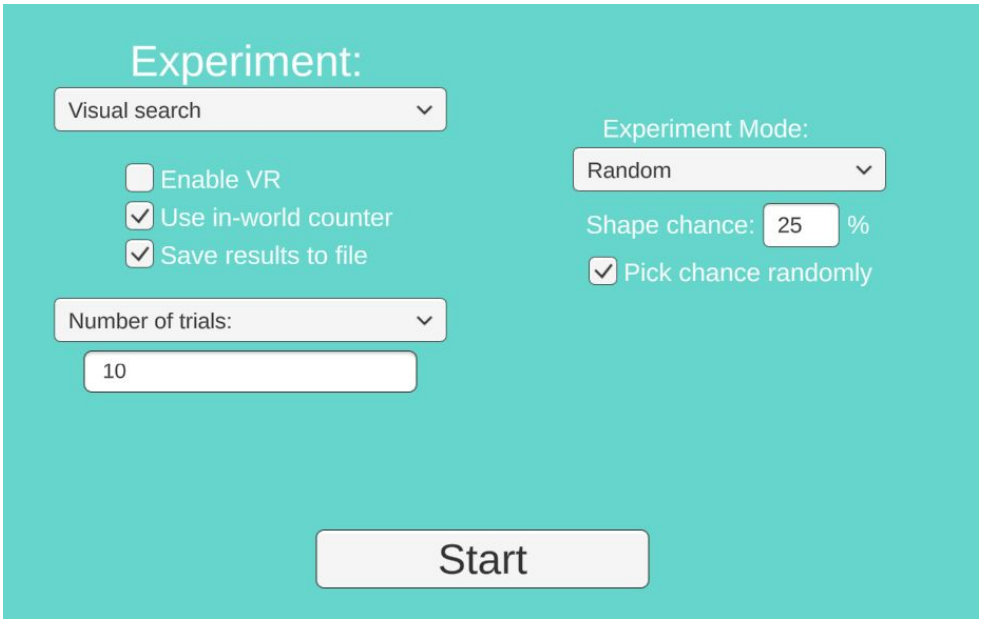


Figure 3. All the buttons and input fields in the main menu, with the visual search experiment selected

The main menu also supports saving and loading the selected options using the Unity PlayerPrefs[12] module. Every time the main menu scene is loaded or a new value from the "Experiment" dropdown is selected, previously entered values are loaded from the PlayerPrefs files, if they exist. New values are saved to the PlayerPrefs files every time the "Start" button is pressed. The main menu is not implemented in virtual reality and cannot be seen through the virtual reality headset or interacted with using the controllers.

3.2 Experiment Scene

Experiments are done in either the FirstPersonRoom scene or the VRRoom scene (see figure 4). The main difference between these two scenes are the character controller they use. FirstPersonRoom uses the FPSController which enables the use of mouse to look around the scene in a first-person environment. VRRoom uses the OVRPlayerController, which enables

[12] https://docs.unity3d.com/ScriptReference/PlayerPrefs.html

the use of a virtual reality headset to look around the scene. The character controllers are also used to adjust the distance from the Screen object. VRRoom also has reduced scale to make it look more believable in virtual reality.

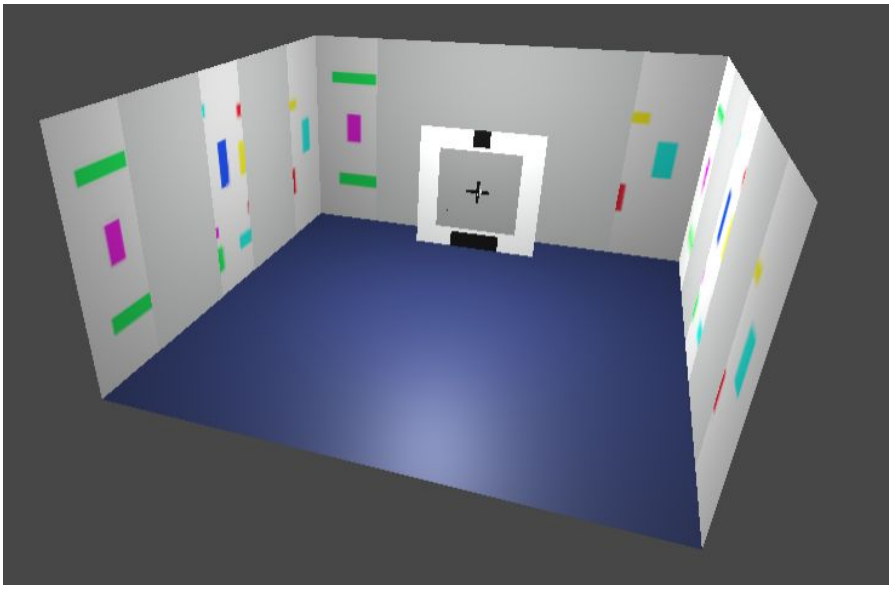


Figure 4. The experiment scene as can be seen in Unity editor. The fourth wall and the ceiling are not visible from this angle due to the way quads are rendered in Unity.

FirstPersonRoom and VRRoom both contain the Room prefab asset. Room prefab asset contains the floor, ceiling and walls that make up the room that the character controller is in and the test subject can see. The walls also have areas with moving textures. The moving textures were implemented to confuse the test subject or potentially artificial intelligence, but are hidden in some areas to reduce possible motion sickness in virtual reality (figure 5). The Room prefab asset also contains a light source.

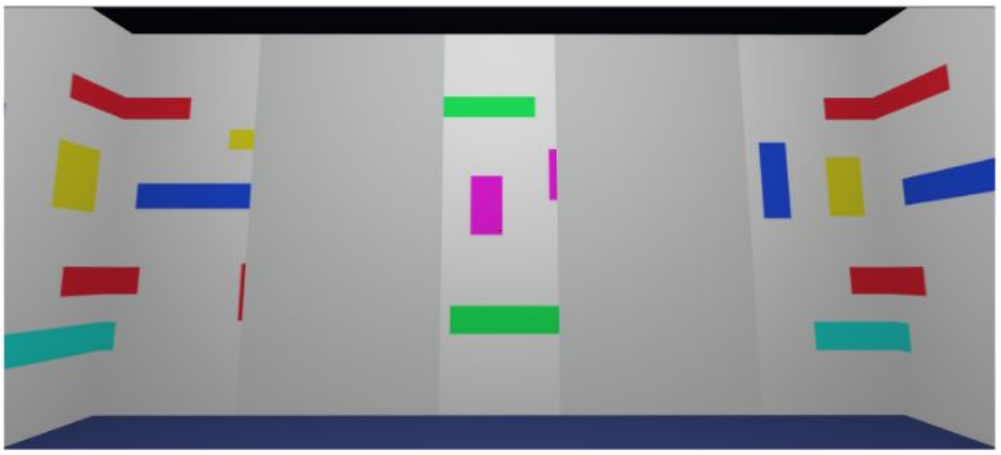


Figure 5. Moving textures that move counterclockwise (from right to left) around the room. The darker gray areas are areas that do not show moving textures.

FirstPersonRoom and VRRoom both also contain the Screen prefab asset. Screen prefab asset contains the response buttons that the test subject will have to look at during the experiment, the area where the images will appear during the experiment, moving circle objects that are used in the multiple object tracking experiment, a canvas where the screen timer (see figure 6) and the experiment-over-text (see figure 8) is shown, and a red cross that is used to trigger the start of each trial. The model for the red cross was made in Blender[13]. The Screen prefab also has the ScreenController script attached to it, which provides most of the functionality for running the experiment.

FirstPersonRoom and VRRoom both also contain a UI canvas and the CrossHair object. The UI canvas is used to render the UI timer (see figure 6). Since the canvas is rendered in screen space it cannot be used in virtual reality. It is not used in the VRRoom but is still left in the scene because the ScreenController script requires it as a reference.

[13] https://www.blender.org/

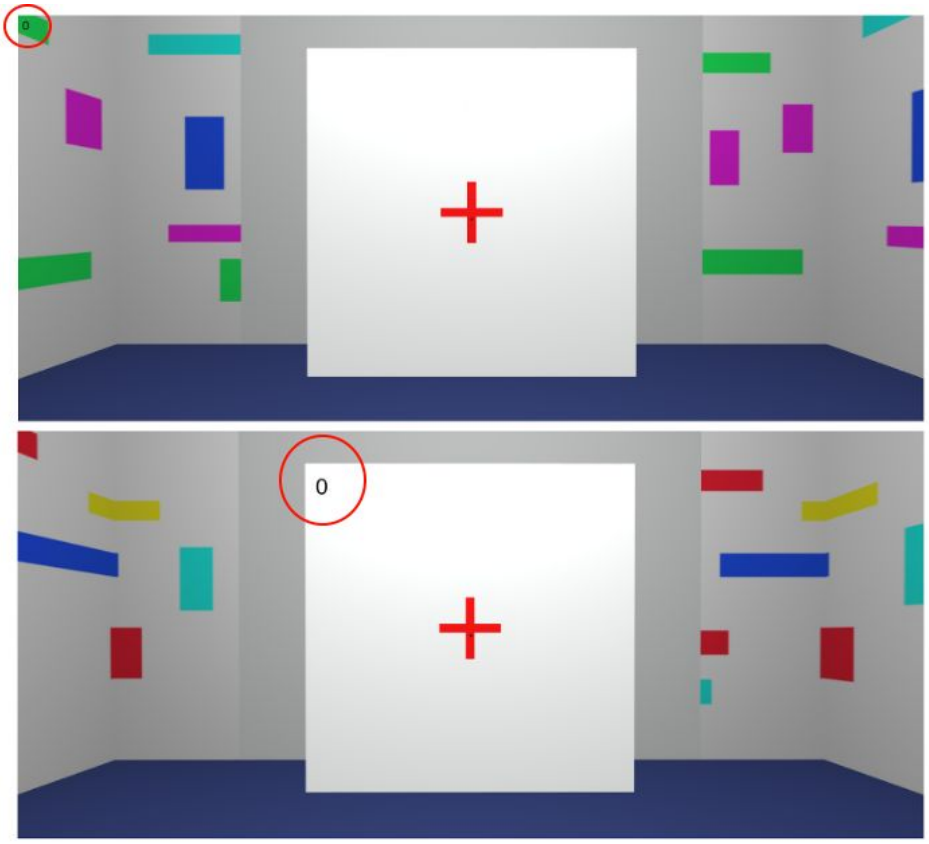


Figure 6. UI timer (top) and screen timer (bottom), highlighted with red circles in this figure for better visibility. UI timer is drawn on the top-left of the UI layer and is always in a fixed position. Screen timer is attached to the Screen object.

The CrossHair object is a small black square that is used to help the test subject determine the object the camera is looking at. The position of the CrossHair is determined by where a raycast from the camera center transform hits an object, and is rotated to face the same way as the object the raycast is hitting. This is implemented mainly for virtual reality, as virtual reality does not allow for a screen space canvas and placing an always visible unmoving object stuck in front of the test subject's field of view could be considered a bad practice [17]. The CrossHair object has the LookAtRaycast script attached to it, which is responsible for calculating the position of the object as well as notifying the response buttons when the test subject looks directly at them.

3.3 Running an Experiment

After selecting an experiment, setting desired parameters and pressing the “Start” button in the main menu, the scene where the experiment will happen is loaded. Upon loading the test subject is positioned looking at the center of the Screen object. The distance from the Screen object can be adjusted with the arrow keys or “w” and “a” keys. A new instance of the chosen experiment controller script is created. It will provide the application with either new images each trial or control the moving objects, and will also provide the expected response to the trial. To start the first trial the test subject has to look at one of the buttons on the screen and then look away from it. Figure 7 outlines a simplified summary of how an experiment would run for the test subjects and the experimenters.

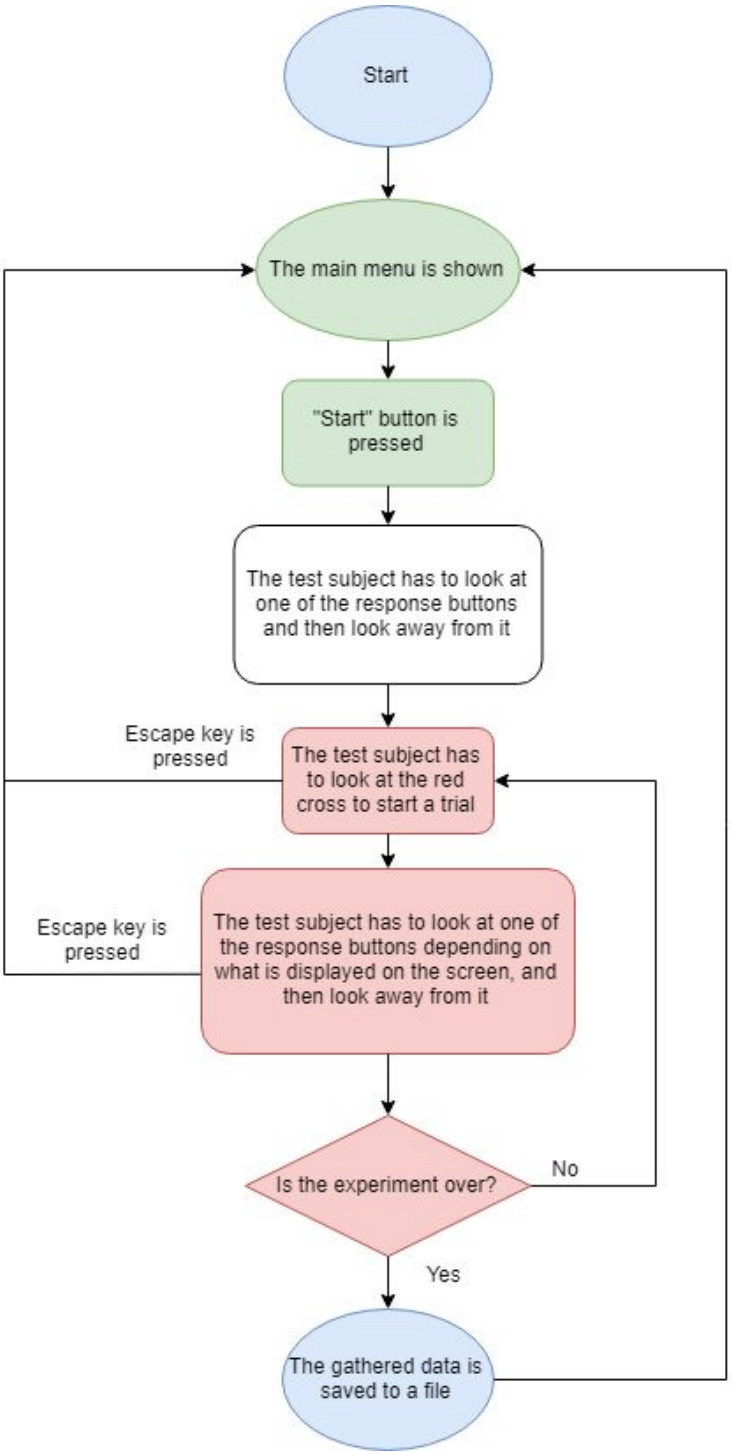


Figure 7. A flow chart of running an experiment. Steps related to the main menu are highlighted in green. Steps that are a part of each trial are highlighted in red.

Every trial starts with a blank screen and a red cross in the middle of it (see figure 4). When the test subject looks at the center area of the red cross either an image is shown on the screen or moving objects are activated for a duration. If this was the first trial it also starts a timer

that counts seconds until the end of the experiment if “Experiment length in seconds” was selected in the main menu. The image that will be shown is selected by the experiment controller. Response buttons appear as soon as the image does, or after a certain amount of time if the experiment contains moving objects.

When the test subject looks at a response button it turns green (see figure 8). The response button that was selected, duration it took the test subject to look at the button from when the response buttons appeared, the correct answer and the trial number are then saved to memory, and later written to a log file. Other data from the trial can also be logged. For that the data has to be added to the dictionary named logData in the experiment controller. Correct answer retrieved from the experiment controller script should correspond to one of the response buttons. After the test subject looks away from the green response button the response buttons and image or moving objects disappear, the trial count is incremented and the next trial starts.

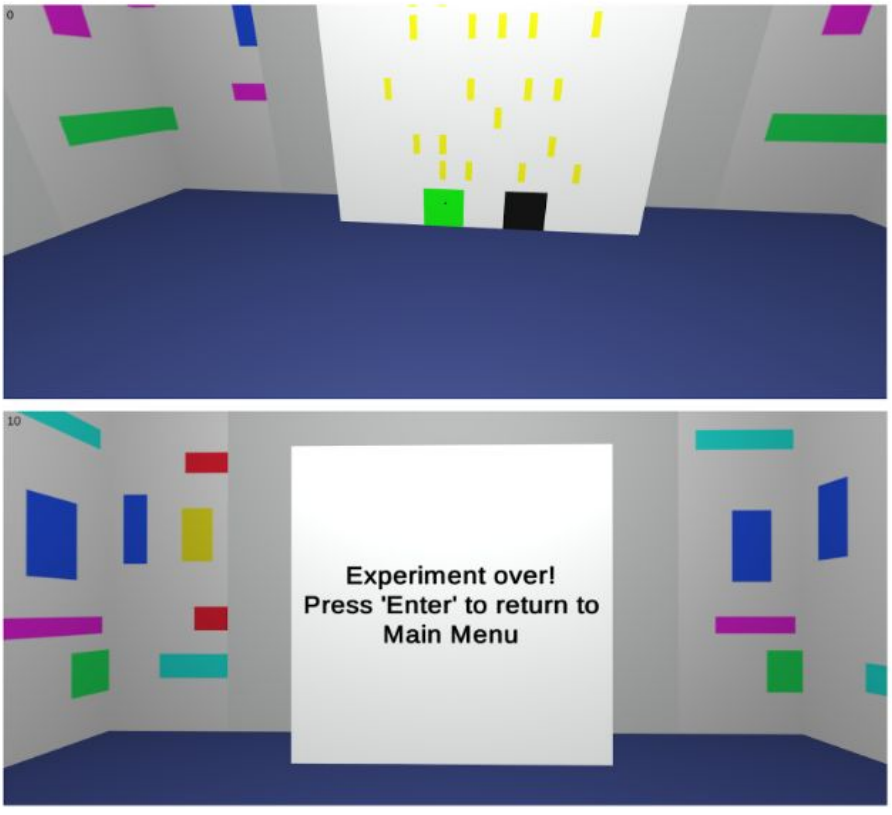


Figure 8. The response button that is looked at turns green (top). Text is shown to notify the test subject that the experiment has ended (bottom).

The experiment end condition is different depending on what was chosen from the main menu. If “Number of trials” was chosen, the experiment ends after a specified amount of trials are completed. If “Experiment length in seconds” was selected, the experiment ends after the timer that started when the first trial started reaches a specified amount of seconds. The experiment can be ended early by pressing the “escape” key, however the data gathered during the experiment will not be saved to a file in that case.

When the experiment ends, the Screen object is made blank again and the words “Experiment over! Press ‘Enter’ to return to Main Menu” appear on the screen (see figure 8). The test subject can then press “enter” or “space” to return to the main menu. If saving results was enabled a CSV file is generated that contains the test subject’s responses, correct responses, response durations and trial numbers that were saved to memory during the experiment. Anything saved to the logData dictionary is also written to the CSV file. The file is saved in a folder named “Logs” in the application directory, and is given a name that contains the current timestamp and experiment name. Example files can be seen in the “Study” folder of the application’s source code. A new experiment can be started from the main menu, the application does not have to be restarted.

The mechanic of having to look away from a response button that has turned green to end the trial is borrowed from Psychlab. Although it might seem counter intuitive it can give the test subject time to reflect on their decision. The requirement to look at a response button to start the experiment is a workaround to a problem where the test subject’s character loaded into the world looking at the center of the screen, immediately looked at the center of the red cross and started the first trial. This is not ideal as the test subject would not be able to get an understanding of the surroundings and react to the image on the screen in time. Otherwise looking at the response button before the start of the experiment is not recorded or required for anything else. Other possible workarounds would be loading the test subject’s character into the world not looking at the center of the screen or starting the experiment after a delay.

3.4 Implemented Experiments

A total of four experiments were added to the finished application in an attempt to re-create some of the tasks in Psychlab. These were the visual search experiment, the continuous recognition experiment, the arbitrary visuomotor mapping experiment and the multiple object tracking experiment.

3.4.1 Visual Search Experiment

The first experiment that was added was the visual search experiment (figure 9). Humans are able to categorize objects when performing visual search, and focus their attention at specific categories of visual objects [18]. Attention can be guided using different attributes, like object color, shape or orientation for example [18]. This makes it efficient to find objects that have characteristics which make them stand out from the rest. However grouping by too many features can cause attention to be overloaded [19]. When this happens attention has to be directed at each object of some attribute group individually [19]. In classical theories these two visual search modes are called serial search and parallel search [20]. Parallel search happens faster because all visual objects are processed simultaneously, whereas serial search requires attention to be allocated one object at a time until a target is found, resulting in slower search times [20]. The implemented visual search experiment has been designed so that some of the trials require serial search and others can be done with parallel search.

In the implemented experiment the test subject has to look for a target amongst distractors. The target always has a specific shape and color, a vertical magenta bar in this implementation. Each trial the test subject has to identify whether the target is present or not. If the target is present the test subject should look at the button on the bottom-right of the screen, otherwise the test subject should look at the button on the bottom-left of the screen. The target and distractors appear on an 8 by 8 grid. The chance that a grid tile contains a distractor can be set from the main menu. The chance that the target appears is 50% and it appears on a randomly selected grid tile. The visual search experiment also has three different modes to control what kind of distractors can appear and an option to randomly choose a mode every trial.

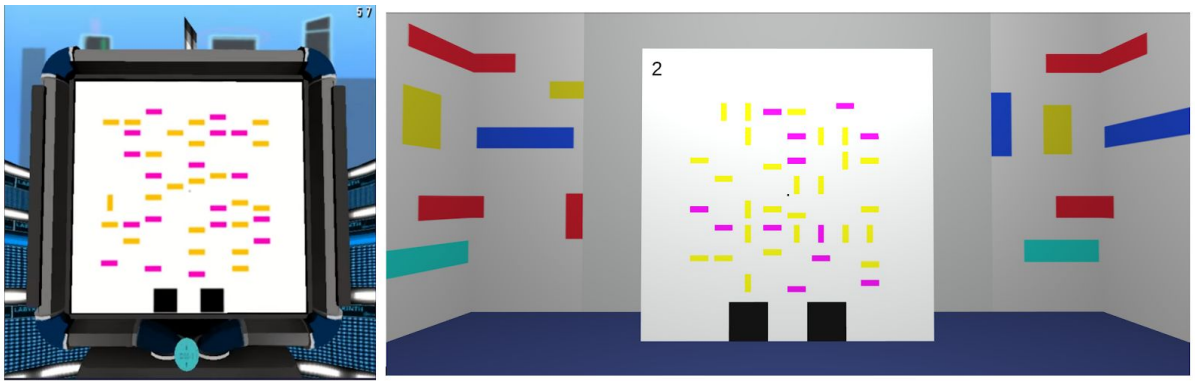


Figure 9. Visual search in Psychlab [2] (left) and in the created application (right)

3.4.2 Continuous Recognition Experiment

Another experiment that was implemented is the continuous recognition experiment (figure 10). The human memory can remember thousands of images with surprisingly high level of detail [21]. Showing an image once for even a couple of seconds is enough for humans to be able to later tell it apart from an another very similar image that was not shown [21]. The implemented experiment can be used to measure the test subject's ability to recall which images have been shown during the experiment by showing both new images and images that have already been shown.

In this experiment an image is shown to the test subject every trial. If the test subject has not seen the image before they should look at the button on the left, otherwise they should look at the button on the right. The chance that a new image is shown can be configured in the main menu. The images that are used in the experiment are located in the "Resources/Images" folder, all the images in that folder are loaded when the experiment scene is loaded. The created application has a set of 50 random images taken from Unsplash[14], retrieved via Lorem Picsum[15] [16]. These images are meant for testing the application and are not meant for actual experimentation.

[14] https://unsplash.com/
[15] https://picsum.photos/
[16] https://picsum.photos/200

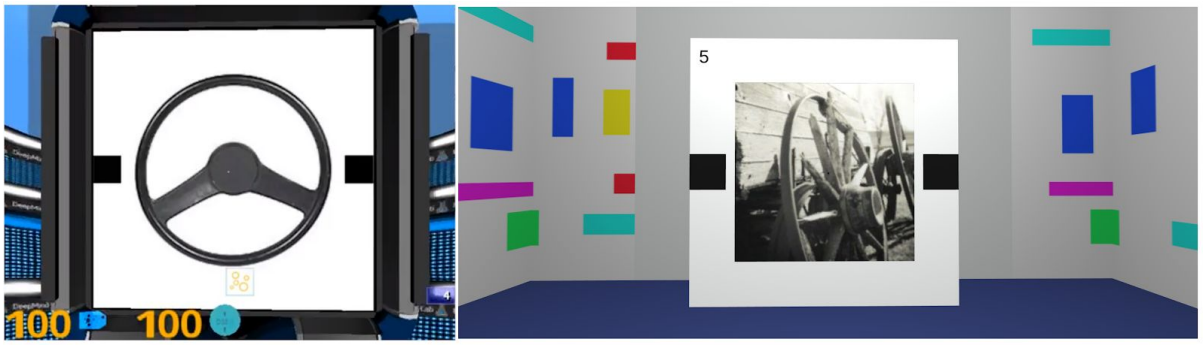


Figure 10. Continuous recognition in Psychlab [2] (left) and in the created application (right)

3.4.3 Arbitrary Visuomotor Mapping Experiment

As an extension to the continuous recognition experiment, the arbitrary visuomotor mapping experiment was implemented (figure 11). Visuomotor mapping refers to transforming visual information into physical motor movement [22]. In standard visuomotor mapping the visual object is usually the target of the action or its location is used for calculating motor coordinates [22]. In arbitrary visuomotor mapping the location of the visual object does not have a direct relationship with the motor action [22]. Instead other characteristics of the visual object are used for instructing movement [22]. In the implemented experiment the test subject has to look at a specific response button depending on the image shown on the screen, thus requiring arbitrary visuomotor mapping to be made.

Similarly to the continuous recognition experiment, this experiment also shows the test subject an image each trial. When an image is shown for the first time, one of the four response buttons is colored green and the others red. The test subject has to remember which button was green and select that button every time this image is displayed in the later trials. When the image is displayed again the response buttons will all be of the same color. The chance that a new image is shown can be configured in the main menu. Because the response button needs to change depending on the image that is shown, a special button feature code was added to the application. This allows the experiment controller to relay information to the response buttons. The arbitrary visuomotor mapping experiment uses the same images as the continuous recognition experiment.

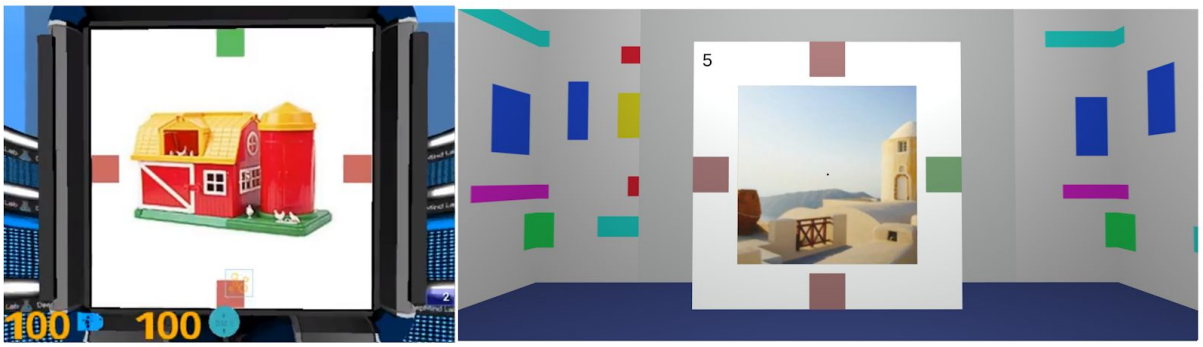


Figure 11. Arbitrary visuomotor mapping in Psychlab [2] (left) and in the created application (right)

3.4.4 Multiple Object Tracking Experiment

To show that experiments with moving details can also be made, the multiple object tracking experiment was implemented (figure 12). Humans have the ability to visually track multiple moving objects in parallel [23]. Most people are able to simultaneously track up to five objects amongst a group of ten identical objects [23]. The implemented experiment can be used to measure the test subject's ability to identify target objects amongst distractors and then visually track them for a certain period of time.

In the beginning of each trial in the multiple object tracking experiment the test subject is shown up to ten circles. The amount of circles is chosen randomly but is always an even number. Half of these circles will be colored green, others will be colored black. After one second the green circles will also be colored black and all circles start moving. The test subject has to memorize and track the green circles. After some time the circles will stop moving and one of them will be colored blue. If the blue circle was initially colored green the test subject has to look at the button on the bottom-right on the screen, otherwise the test subject should look at the button on the bottom-left. The speed that the circles move at and the duration they move for can be configured in the main menu.

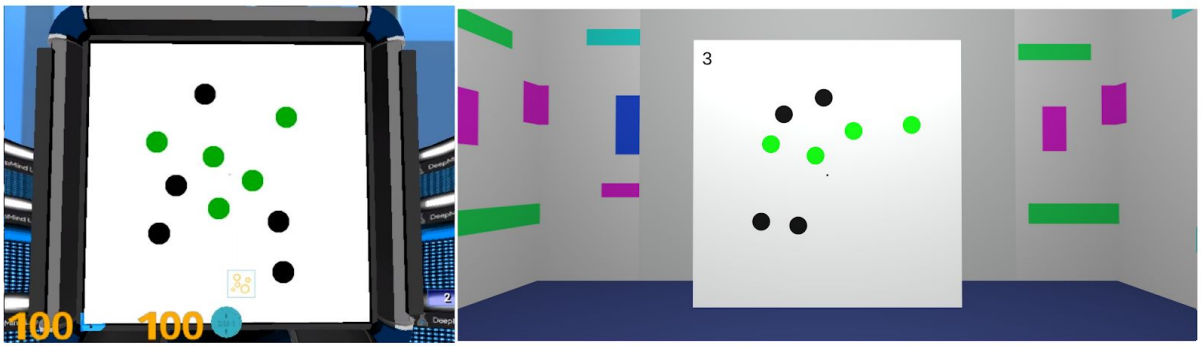


Figure 12. multiple object tracking in Psychlab [2] (left) and in the created application (right). The circles will move around during the experiment.

To display the moving circles on the screen circular gameobjects were added as a part of the Screen prefab. The circle model was created in Blender. Each circle has a MovingCircle script attached to it, which controls the movement of the circle. To avoid overlap with other circles and circles moving off the screen, each circle calculates the distance from every other circle and if it is too close to another circle or too far away from its origin point a new random direction is assigned. The new direction is assigned so that the circle would be either moving towards the center of the screen again or away from another circle that is too close.

The experiment controller for the object tracking experiment decides how many circles will be active in a trial and their initial colors, and also provides circles references to access other circles. Because of the addition of moving objects and a slightly different behaviour of response buttons, the multiple object tracking experiment is more hard-coded than other implemented experiments, and adding new experiments with moving details could be significantly more difficult than adding new experiments without moving details.

3.5 Adding an Experiment

It is fairly easy to add new experiments:

1. Create a new script with a class that extends the Experiment abstract class. This is the script that controls the experiment. It must implement the GetNextTexture method. This method is called every trial and is used to either retrieve the image shown on

screen for that trial or start the moving objects. The method should also set the correctAnswer variable to the expected answer for current trial: 0 if left button should be selected, 1 if right button should be selected, 2 if top button should be selected, 3 if bottom button should be selected. The new script must also have a constructor method that can be used to initialize an object of it. If additional experiment related data has to be logged a new entry can be added to the logData dictionary in the new script and updated every trial.

2. The experiment needs to be initialized in the Start method of the ScreenController script if it is selected. For this a new else if statement needs to be added at the end of the method similar to the ones used for the other experiments. The buttons that will be used and their positions can also be set here.
3. Optionally a new selection can be added to the experiment dropdown menu and new UI elements for experiment related options under the RightMenu in the main menu. For the added menu items to work changes have to be made in the Menu script.

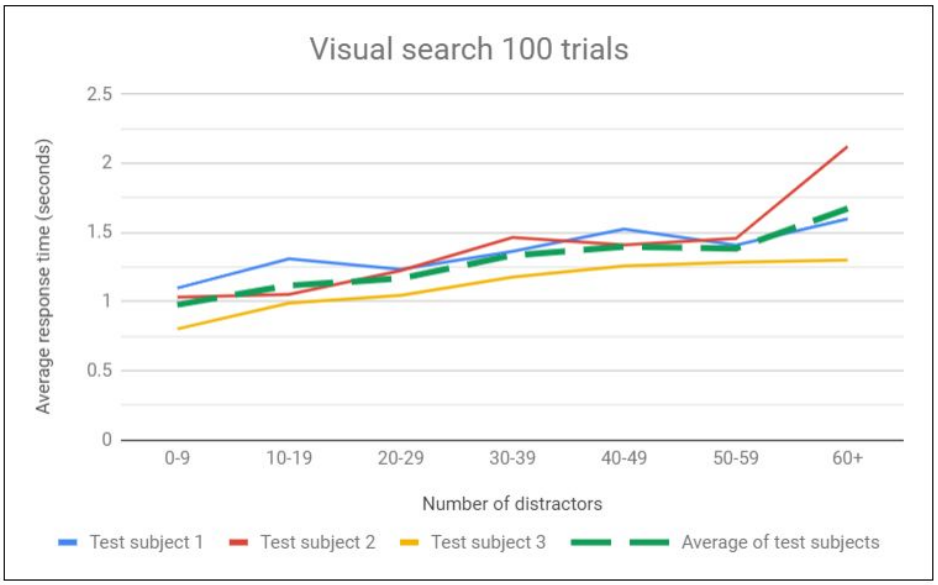
An ExampleExperiment script is added along other experiment scripts and the example experiment is also included in the ScreenController script. The example experiment showcases steps 1 and 2 described above. After changes have been made the application either has to be compiled again or launched through the Unity editor. If the experiment needs to use new response buttons or moving objects they have to be added through Unity, changes need to be made to multiple scripts and they have to be referenced where necessary.

4 Testing

4.1 Pilot Study

To test the usability of the application two pilot studies with human test subjects were conducted. Because the main feature of the application is the ability to run experiments in virtual reality, both studies were done in virtual reality using the Oculus Rift CV1 headset.

For the first study the visual search experiment was used. Three test subjects participated, completing 100 trials each. From experiment options “Different shape and color” experiment mode was selected and “Pick chance randomly” was ticked. Because of this every trial had a random amount of distractors and the data from the log file was later sorted by the amount of distractors. The measured response times from each trial were grouped by the distractor amount and then averaged. A graph showing the data can be seen in figure 13.



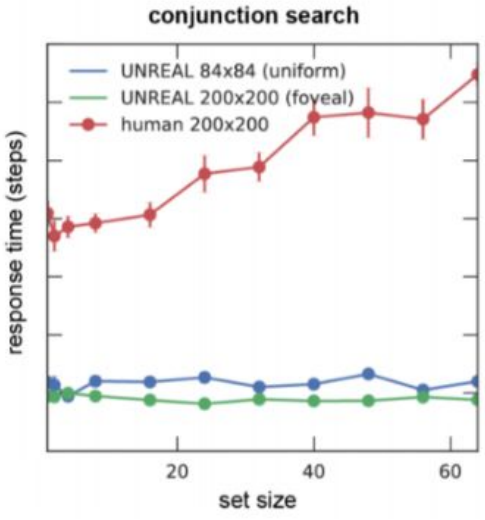


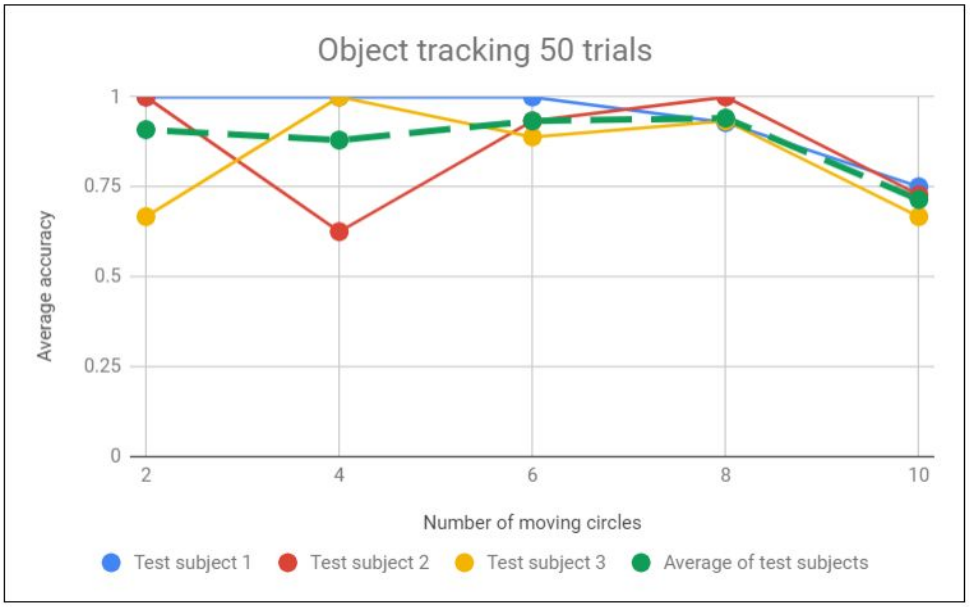
Figure 13. Data from the 100 human trials of the visual search experiment done in virtual reality (top) and results of a similar experiment published by Psychlab [2] (bottom). The "conjunction search" in Psychlab corresponds to "Different shape and color" option in the created application.

As can be seen from figure 13, the response times did increase as more distractors appeared on the screen. This was the expected result and is similar to the results published by Psychlab (see figure 13). The amount of mistakes made by test subjects was very small, with two to three mistakes made during the 100 trials. It should be noted that the image shown to the test subject might have actually contained one less distractor than was recorded. This was caused

by the fact that the target could sometimes appear in a position that already contained a distractor and would then overwrite it.

For the second study the multiple object tracking experiment was used. The study was done with the same three test subjects, this time completing 50 trials each as trials in this experiment take significantly longer than trials in the visual search experiment. From experient options speed of the moving circles was set to 7 and trial length was set to 3. It should be noted that the speed at which the circles move feels significantly faster in virtual reality. The gathered data was grouped by the amount of circles shown and test subject's accuracy was calculated for each group.

The graph of the data, seen in figure 14, shows a clear drop in accuracy when the moving objects count increased to ten. This is similar to what was shown by Psychlab (see figure 14). At least one test subject verbally reported that when doing a trial with very few objects it was easy to not pay enough attention at the start and forget which circles were colored green. This can be seen in the graph, as two out of three test subjects often had difficulty during seemingly simple trials. The amount of moving circles shown was chosen randomly at the start of each trial and no adaptive difficulty procedures were used.



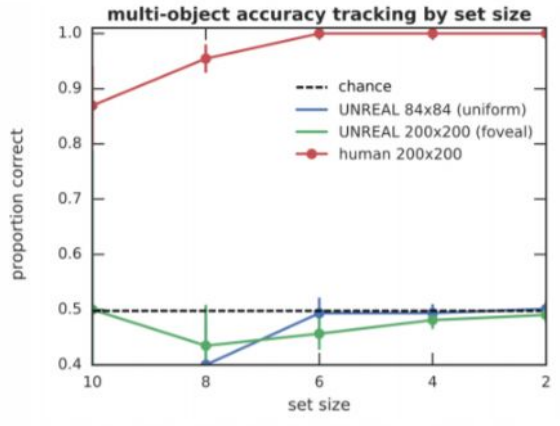


Figure 14. Data from the 50 human trials of the object tracking experiment done in virtual reality (top) and results of a similar experiment published by Psychlab [2] (bottom). Notice that the x axis is reversed on the bottom graph.

None of the test subjects reported sickness or disorientation caused by the use of virtual reality, although one subject reported slight neck strain after the experiments, suggesting that perhaps the screen in virtual reality could be smaller to minimize big head movements. The mechanic of having to look away from a response button to end a trial turned out to be confusing to some test subjects, as they would just keep the CrossHair object over the green response button waiting for the red cross to appear at the center of the screen. The log files with the data that was gathered during the study are available in the "Study" folder of the application's source code.

The main goal of the pilot study was to test if the gathered data is useful and can be analyzed, and whether the test subjects find the experiments in virtual reality comfortable or not. Because of this, the amount of data gathered was very small and it should not be used to draw any definitive conclusions about psychophysics. To find out whether or how human responses to these experiments differ when done in virtual reality versus when done on a normal computer monitor and with a mouse, a more extensive study with more test subjects should be done.

4.2 Future Improvements

The main feature that exists in Psychlab but is missing in the created application is the ability to run experiments with machine learning agents. This feature was not implemented due to the limited time frame of the current thesis. The simplest way to implement this feature would most likely be to use the Unity ML-Agents Toolkit[17]. As mentioned earlier, machine learning agents from this toolkit are able to process visual data as input [6], making them suitable to be trained on implemented experiments. The machine learning agents in Psychlab were successfully able to learn and perform the visual search experiment, however were unable to successfully perform the multiple object tracking experiment [2]. Information about machine learning agents attempting either continuous recognition experiment or arbitrary visuomotor mapping experiment was not included in the research paper detailing Psychlab [2].

In addition to adding machine learning support, more experiments could also be added to the application. Psychlab has a total of eight example tasks, four of which were re-created in this application given the scope of the thesis. Perhaps GUIs and user experience could also be improved, as these were not the main focus during the development of the application. Currently the test subjects have to be explained or shown how to complete the experiments and which response buttons to look at, so showing some kind of explanatory text or image before the first trial of an experiment could be preferable, similar to how it was done in

[17] https://github.com/Unity-Technologies/ml-agents

PEBL. PEBL could also serve as an example of how to design a GUI with more features and better usability, like showing a preview of the selected experiment (see figure 2). As the mechanic of looking away from response buttons to end a trial turned out to be confusing for some test subjects, making the trial end immediately after looking at a response button might make experiments more intuitive.

Conclusion

During the work for the thesis a Unity application that can run psychological experiments in both virtual reality and in a first-person perspective on normal computer monitors was developed. Two already existing psychological testing environments were analyzed and Psychlab was chosen as an example. Four of the experiments from psychlab were re-created and implemented into the newly created application. To make the application easier to use a main menu was added, that allows to change between experiments and change different options.

The thesis also details the technologies used to develop the application and gives an in-depth overview of how the application works. Explanations about the implemented experiments and their backgrounds were also given. A short guide on adding new experiments to the application was included.

A pilot study to test the usability of the application was conducted. The study showed that usable information can be gathered using the created application, but also found some minor flaws in the design of it. Possible future improvements were suggested, most importantly integrating machine learning into the application and testing how it behaves when performing experiments.

Even though humans and artificial intelligence agents can perform the exact same psychological experiments is some already existing programs, they have to do it in different environments. This thesis is hopefully a small step towards a more comparable study methodology and other researchers can use and build on the open source code to gain new insights on both biological and artificial brains.

References

[1] Adams R. L. 10 Powerful Examples Of Artificial Intelligence In Use Today. https://www.forbes.com/sites/robertadams/2017/01/10/10-powerful-examples-of-artificial-intelligence-in-use-today/#53423c2b420d (13.08.2019)

[2] Leibo J. Z., d'Autume C. M., Zoran D., Amos D., Beattie C., Anderson K., Castañeda A. G., Sanchez M., Green S., Gruslys A., Legg S., Hassabis D., Botvinick M. M. Psychlab: A Psychology Laboratory for Deep Reinforcement Learning Agents. *arXiv*, 2018. https://arxiv.org/pdf/1801.08116.pdf (12.08.2019)

[3] Beattie C., Leibo J. Z., Teplyashin D., Ward T., Wainwright M., Küttler H., Lefrancq A., Green S., Valdés V., Sadik A., Schrittwieser J., Anderson K., York S., Cant M., Cain A., Bolton A., Gaffney S., King H., Hassabis D., Legg S., Petersen S. DeepMind Lab. *arXiv*, 2016. https://arxiv.org/pdf/1612.03801.pdf (12.08.2019)

[4] Mueller S. T., Piper B. J. The Psychology Experiment Building Language (PEBL) and PEBL Test Battery. *Journal of Neuroscience Methods*, 2014, No 222, pp 250-259.

[5] Homepage of ioquake3. https://ioquake3.org/ (13.08.2019)

[6] Juliani A., Berges V.-P., Vckay E., Gao Y., Henry H., Mattar M., Lange D. Unity: A General Platform for Intelligent Agents. *arXiv*, 2018. https://arxiv.org/pdf/1809.02627.pdf (12.08.2019)

[7] Public Relations. https://unity3d.com/public-relations (13.08.2019)

[8] GameObject. Unity Documentation. https://docs.unity3d.com/Manual/class-GameObject.html (13.08.2019)

[9] Creating and Using Scripts. Unity Documentation https://docs.unity3d.com/Manual/CreatingAndUsingScripts.html (13.08.2019)

[10] A feature-rich and highly flexible editor. https://unity3d.com/unity/editor (13.08.2019)

[11] VR overview. Unity Documentation. https://docs.unity3d.com/Manual/VROverview.html (13.08.2019)

[12] Binstock A. Powering the Rift. https://www.oculus.com/blog/powering-the-rift/ (13.08.2019)

[13] Rendering. https://developer.oculus.com/design/latest/concepts/bp-rendering/ (13.08.2019)

[14] Oculus Unity Getting Started Guide.
https://developer.oculus.com/documentation/unity/latest/concepts/ (13.08.2019)
[15] Oculus Touch Launches Today!.
https://www.oculus.com/blog/oculus-touch-launches-today/ (13.08.2019)
[16] Announcing Oculus Rift S, Our New PC VR Headset Launching Spring 2019 for $399.
https://www.oculus.com/blog/announcing-oculus-rift-s-our-new-pc-vr-headset-launching-spring-2019/ (13.08.2019)
[17] Vision. https://developer.oculus.com/design/latest/concepts/bp-vision/ (13.08.2019)
[18] Wolfe J. M., Horowitz T. S. Five factors that guide attention in visual search. *Nature Human Behaviour*, 2017, No 1, 0058.
[19] Treisman A. M., Gelade G. A Feature-Integration Theory of Attention. *Cognitive Psychology*, 1980, No 12, pp 97-136.
[20] Moran R., Zehetleitner M., Liesefeld H. R., Müller H. J, Usher M. Serial vs. parallel models of attention in visual search: accounting for benchmark RT-distributions. *Psychonomic Bulletin & Review*, 2016, No 23, pp 1300-1315.
[21] Brady T. F., Konkle T., Alvarez G. A., Oliva A. Visual long-term memory has a massive storage capacity for object details. *PNAS*, 2008, No 105 (38), pp 14325-14329.
[22] Murray E. A., Bussey T. J., Wise S. P. Role of prefrontal cortex in a network for arbitrary visuomotor mapping. *Experimental Brain Research*, 2000, No 133, pp 114-129.
[23] Pylyshyn Z. W., Storm R. W. Tracking multiple independent targets: evidence for a parallel tracking mechanism. *Spatial Vision*, 1988, No 3, pp 179-197.

Appendix

I. Source Code

The source code for the created application can be accessed on GitHub at

https://github.com/reneekroon/PsychlabU

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, Renee Kroon,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

Application for Psychophysics Experiments in Virtual Reality

supervised by Jaan Aru and Madis Vasser.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Renee Kroon
14/08/2019