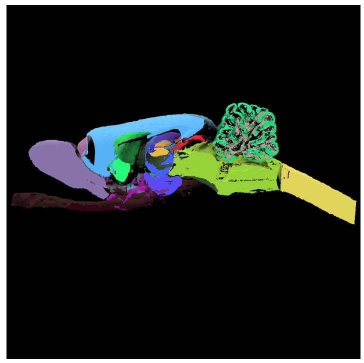
NTNU – Trondheim
Norwegian University of
Science and Technology



Towards a 3D model of the rat brain in AR as an educational tool

Ole V. Ravna

Autumn 2020

PROJECT REPORT
TDT4501 - Specialization Project
Department of Computer Science
Norwegian University of Science and Technology

Supervisor 1: Gabriel Kiss

Supervisor 2: Ekaterina Prasolova-Førland

Abstract

This study aims to explore whether Augmented Reality can be used as a tool for medical students learning neuroanatomy. A application, *Nevrolens*, was created with features simulating a conventional rat brain dissection. While the results from this study are limited, the application has shown promise in its use as a virtual simulation of a rat brain dissection.

Contents

Chapter 1

Introduction

1.1 Motivation

Augmented reality is a technology which has experienced great leaps in recent years, and this growth has inspired many visons of medical potentials for this young technology. Within medical education there are many fields where visual understanding is critical, one of being neuroanatomy. Neuroanatomy is a highly complex domain both visually and spatially, the ability to use the human senses in a real-world setting could result in greater intuition and understanding. With that in mind the use of augmented reality could be a natural way to virtualize the experience of a brain dissection, and further the unique capabilities of AR could enable innovative ways of learning. (Moro et al., 2017) shows the possibility of greater immersion and engagement while using augmented reality in teaching anatomy to medical students. This has also recently been shown with promising result by (Wish-Baratz et al., 2020), where COVID-lockdown required from-home teaching, and the use of HoloAnatomy, an anatomy application for the HoloLens, performed significantly better than even conventional in-class lectures.

1.2 Problem Formulation

The main problem with most academic implementations, like (Wish-Baratz et al., 2020), of AR in medical education is the use of head-mounted display (HMD) devices like the HoloLens 2 and Magic Leap, which in the near to mid-term future will have limited practical use in education, as a result of the high price-tag, combined with the still inadequate general use-case for these types of devices. This project will try to mend these challenges by having the lecturer using an HMD and having student view and interact with the lecture in an AR-based application running on their smartphone. This is possible because of the great leap in AR-performance seen in recent models of Android and especially iPhones, in combination with development platforms like Unity, Mixed Reality Toolkit and **??** which enables multiplatform development and real-time collaboration between devices. The aim of the project will be to create a seamless educational experience in Augmented Reality which can be valuable both on an HMD device and a modern smartphone. The focus will be on investigating its feasibility as an educational tool both in a lecture-type setting and for students to explore the brain anatomy independently.

1.3 Research Questions

What follow are the research questions which motivates this project:

Main-RQ: How can AR support teaching of neuroanatomy and dissection for medical students?

- **Sub-RQ1:** How should interaction in be implemented in AR to accommodate medical students and educators?
- **Sub-RQ2:** How will a collaborative experience shared between an HMD and a smartphone compare to accommodate medical users?
- **Sub-RQ3:** Can understanding be increased by integrating microscopical data into a macroscopical model?

1.4 Approach

Research method

The research questions were derived through discussing the needs of the intended users with neuroscientists at the Kavli Institute. It was then narrowed down by a literature review, finding a lack of satisfactory substitutions for real brain dissections and especially finding no

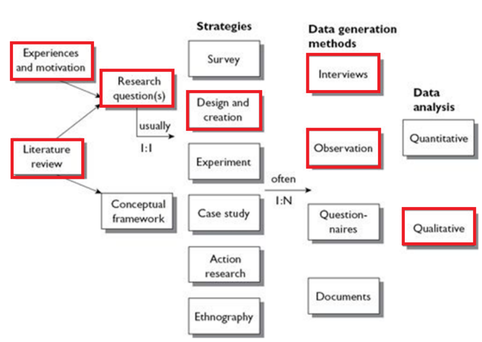


Figure 1.1: Model of the research process as illustrated in Oates (2006)

attempt at a practical multiplatform application for a more scalable use for students. The projects research question falls under the strategy of Design and Creation as the main goal is to develop a useful application for medical education. The focus on a smartphone solution was further motivated by the COVID-pandemic making from-home learning quite essential and making the passing around of HMD devices an unwanted scenario. As part of an agile software development model the gathering of qualitative data from observations and interviews within the scope of user testing will be essential.

1.5 Contributions

The research product resulting from this project will be a new computer-based software application using augmented reality and running on multiple platforms like HoloLens 1 and 2, Android and more. The aim will be to develop an application that can bridge the gap between expensive head mounted displays and everyday smartphones which you will find in the pocket of any student, and to use this as a collaborative tool for learning neuroanatomy. Throughout the development period we will consult with medical professionals and gather feedback from students on the usability of the application.

Chapter 2

Background

2.1 Augmented Reality

Augmented Reality (AR) describes the use of computer technology to generate an audio-visual experience combining real-world impressions with computer generated graphics, and, *essentially*, the ability to interact seamlessly within both domains. Ever since its infancy medical usage of AR technology has been envisioned as a great potential. The idea of x-ray vision is seen both in science fiction and in genuine research dating all the way back to the 1930s when H. Steinhaus explored ways to visualize metal pieces inside the body (Sielhorst et al., 2008). There is now substantial interest in the use of AR within a wide array of medical fields as well as in industry and education. As an emerging technology there is still much research needed, and great leaps in hardware, software and sensor capabilities are bound to happen in the near future. Already AR shows promising results in both surgical settings and in education (Singh and Kharb, 2013).

Disambiguation of some acronyms

As a new field, this field suffers from naming disagreements. This is a confusing reality which needs to be addressed. There are differing view of what each acronym refers to, and even what they stand for. I will overlook most of this discussion, and simply explain what is meant by each acronym in the scope of this project.

AR Augmented Reality, exercises which implement a see-through effect to display 3D visuals on top of the real-world. The idea of holograms is a good stand-in for the effect of AR. This is the term which will be most used in this project.

MR Mixed Reality, anything within the spectrum between reality and pure visual 3D graphics, which blends computer generated visuals and reality. While the term has been in use since it was coined by Milgram and Kishino (1994), it has in recent years been strongly associated with Microsoft, and in this project the term MR will generally only be used as a reference to Microsoft's products or concepts. The term is also used by some as a subset of AR, so in conclusion it is a somewhat controversial term.

XR Extended Reality, much like MR this includes the whole spectrum of experiences blurring the line between the real and the virtual. However it does not have the Microsoft taint, nor the confusion of that term. And thus it is a more acceptable term, and it is what will be used here to describe the spectrum.

VR Virtual Reality, is enclosed experiences which completely surrounds the user within a computer generated world. This is a generally uncontroversial term and will be used for applications running on devices like the Oculus Rift and the HTC Vive.

2.2 Graphics and Rendering

Models old

Three dimensional data can be stored and visualized in a number of ways, and the way a graphical application like Nevrolens does it is very different from the ways of medical applications. Within medicine volumetric data is common, as it is just as important what's inside the model as what's outside. In conventional graphics 3D models are built up of 2D polygons which added up forms a 3D structure, this reduces rendering time while keeping the outside structure of the model intact. Figure **??** show a model with about 15 thousand polygons.
The process of generating polygonal models from volumetric models is quite complex, Elden (2017) writes about this process in some detail, which can be found in Appendix B. This is the model I have used in Nevrolens though the model had to be simplified further to about 300,000 triangles, to run decently on the HoloLens 2.

Models

Colors

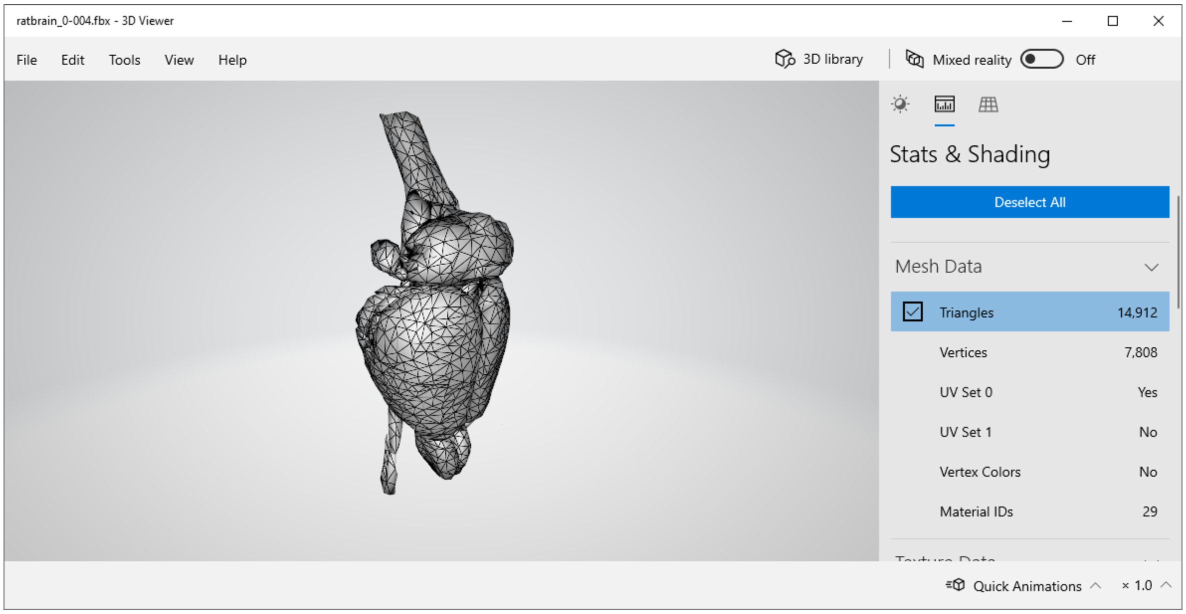


Figure 2.1: Wireframe view of the low resolution rat brain in Microsoft 3D Viewer. The triangles are clearly distinguishable, in total there is 14,912 triangles in this model.

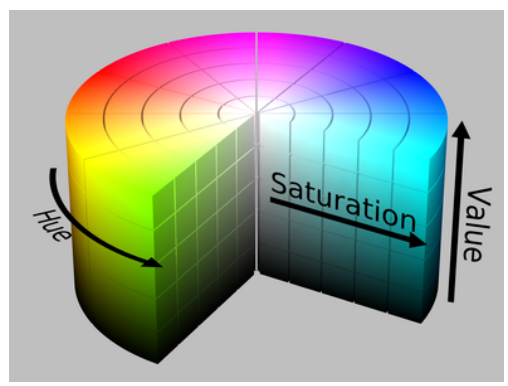


Figure 2.2: HSV color model represented cylindrically.

Within computer graphics colors are generally encoded by extracting the primary color values in separate channels, this is called RGB (Red, green, blue) and is the basis for most color models used on computers. This is naturally used widely in this project, and will not be explained further. There is however another less common color model used in this project which has some useful properties worth exploring. This is the HSV color model. The different channels are hue, saturation, and value. The hue is simply the color based on a traditional color wheel, this means that the color will periodically repeat starting with red on hue=0, green on hue=0.33 and green on hue=0.67. The saturation is how "colorful" the color is, there 0 is a gray-scale and 1 is completely colored. The value is sometimes also called light or brightness, where 0 is black and 1 is again completely colored.

Transformation

2.3 Neuroanatomy

The study of neuroanatomy is concerned with the structural organization of the nervous system. This primarily means the brain and its structures, and is what this project will focus exclusively on. Within the study of neuroanatomy, the use of macroscopical brain dissections have long been the conventual practice for teaching the organization of the structures in the brain. Requiring cadavers and the single use of their brain, this method is highly resource intensive and has limited scalability. In addition, there are deeply concerning ethical challenges with the use of animals in research.

The Waxholm Space Atlas of the Sprague Dawley Rat Brain

This project makes use of high-resolution 3D-models of a rat brain. This brain model has been captured and manually delineated[1] by a collaboration between research groups at the University of Oslo and NTNU, and is in fact a highly accurate volumetric representations of the rat brain. This model is an open access community resource, intended as a free tool for education and research[2]. Within the convectional rasterization rendering pipe-line of Nevrolens, a geometric asset derived from this volumetric model is naturally used.
The model is referred to as *The Waxholm Space Atlas of the Sprague Dawley Rat Brain*. That means a atlas of the *Sprague Dawley* rat breed defined in Waxholm Space. I will briefly explain what a brain atlas is and what Waxholm space is.

Brain Atlas

A brain atlas is a composite representation based on one or more datasets of a given brain. An atlas generally has the function of highlighting some specified aspects and relations in the brain, and is a convectional tool in neuroscientific research (Toga and Thompson, 2000). The convectional atlas is based on micrometer scale sliced sections in the brain, effectively creating two-dimensional layers through the brain. While functional, this "turns the brain into a book". Three-dimensional digital atlas are however relative newcomer on the neuro-imagery scene, by employing magnetic resonance imaging (MRI) and diffusion

[1] Delineation refers to the process of clearly defining different structures in the brain into separate namable parts.
[2] https://www.nitrc.org/projects/whs-sd-atlas

tensor imaging (DTI) the resulting atlases are complete volumetric representation of the subject brain (Papp et al., 2014).
This volumetric model is the basis for the delineated 3D-model used in Nevrolens.

Waxholm Space

Waxholm Space (WHS) is a vector space defined as a standard reference space for the mouse brain and the rat brain (Papp et al., 2013). Its use as a coordinate system simplifies interoperability across atlases. It was developed by International Neuroinformatics Coordinating Facility (INCF) for the mouse brain, and has further been implemented in the rat brain by Papp et al. (2014). The following is the formal definition of WHS:

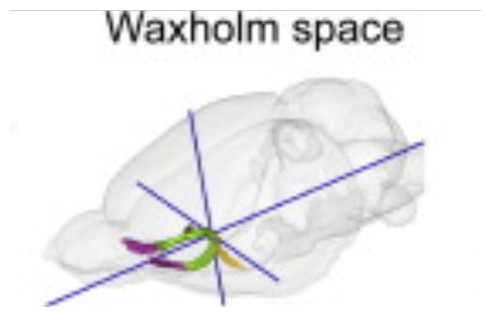


Figure 2.3: WHS (Papp et al., 2014)

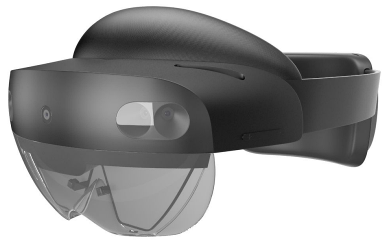
> *The coordinate system for WHS is defined as a continuous Cartesian system with the origin in the brain determined by*
>
> - *the anterior commissure (AC) at the intersection between the mid sagittal plane,*
> - *a coronal plane passing midway (rostro-caudal) through the anterior and posterior branches of AC, and*
> - *a horizontal plane passing midway through the most dorsal and ventral aspect of the AC.*
>
> Hawrylycz et al. (2011)

Figure 2.3 visualizes the axes through origin of WHS in the brain of a rat. Within the scope of this project WHS will be the local space of the rat brain model implemented in Nevrolens.

2.4 Equipment

HoloLens 2

HoloLens 2 is the second iteration of Microsoft immersive headset line. It uses an ARM-based computing unit, running a custom holographic version of Windows 10 for ARM.



This enables the HoloLens 2 to produce high quality graphics while being very power efficient. It was announced in early 2019, with a limited release on November 7, 2019. It is however jet, as of writing, not publicly available and could be considered a limited industrial product. As the technology stands today, HoloLens 2 is the most complete augmented reality device on the marked, with interaction features like hand tracking and eye tracking in combination with the most immersive display technology in any AR HMD. This makes it a natural device choice for developing AR applications with today. show the specifications for HoloLens 2. Very helpfully the HoloLens 2 has on-board screen capturing tools and the option to live preview the video feed from the Windows Device Portal. These features have helped greatly both in user testing and in demonstration the application.

2.5 Software Tools

Unity

Unity is a game engine for developing 2D and 3D games. It has grown to become the most popular game engine used by single developers and small development teams because of its ease of use and simple licensing terms for independent developers. Because of its popularity and ease of use Unity has become a platform for 3D development within more widespread fields than video gaming, such as engineering, moviemaking and architecture. Within this project the critical reason for choosing Unity for our 3D development is the exceptional support for the HoloLens product line. As seen in the section 2.5, Microsoft has poured resources into developing a "relatively" robust open framework for using Unity to develop for HoloLens. Alternatives to using Unity are slim, but one could be to use Unreal Engine, an 3D game engine with great support for VR and AR in general, however the support for HoloLens specific tools like Mixed Reality Toolkit is limited, Microsoft has a version of MRTK for Unreal, called MRTK-Unreal. It seems to be stale however, not having

Display	Optics: See-through holographic lenses (waveguides) Resolution: 2k 3:2 light engines Holographic density: >2.5k radiants (light points per radian) Eye-based rendering: Display optimization for 3D eye position
Sensors	Head tracking: 4 visible light cameras Eye tracking: 2 IR cameras Depth: 1-MP Time-of-Flight (ToF) depth sensor IMU: Accelerometer, gyroscope, magnetometer Camera: 8-MP stills, 1080p30 video
Audio and speech	Microphone array: 5 channels Speakers: Built-in spatial sound
Human understanding	Hand tracking: Two-handed fully articulated model, direct manipulation Eye tracking: Real-time tracking Voice: Command and control on-device; natural language with internet connectivity Windows Hello: Enterprise-grade security with iris recognition
Environment understanding	6DoF tracking: World-scale positional tracking Spatial Mapping: Real-time environment mesh Mixed Reality Capture: Mixed hologram and physical environment photos and videos
Compute and connectivity	SoC: Qualcomm Snapdragon 850 Compute Platform HPU: Second-generation custom-built holographic processing unit Memory: 4-GB LPDDR4x system DRAM Storage: 64-GB UFS 2.1 WiFi: Wi-Fi: Wi-Fi 5 (802.11ac 2x2) Bluetooth: 5 USB: USB Type-C

Figure 2.4: Specifications for the HoloLens 2

any updates in the last six months in the time of writing.

Mixed Reality Toolkit

Mixed Reality Toolkit (MRTK) is a open source, Microsoft-driven framework for Mixed Reality (MR) development. In practice it is Microsoft's SDK for HoloLens development, greatly simplifying development related to interaction, user interface and [...]. As it is a framework for MR in general, it supports other platforms like Android, iOS and VR devices such as HTC Vive and Oculus Rift with OpenVR. An alternative to using MRTK would be to us XRTK which is a community-driven fork of MRKT. Thought such a choice would be an exercise of free software principles, it also lends it self to better support for some devices, like the MagicLeap.

Blender

Blender is a 3D modeling application, it is free open source software and is has a wide set of functionalities for 3D modeling, animation, rendering and optimization. This was chosen because of its free and open availability.

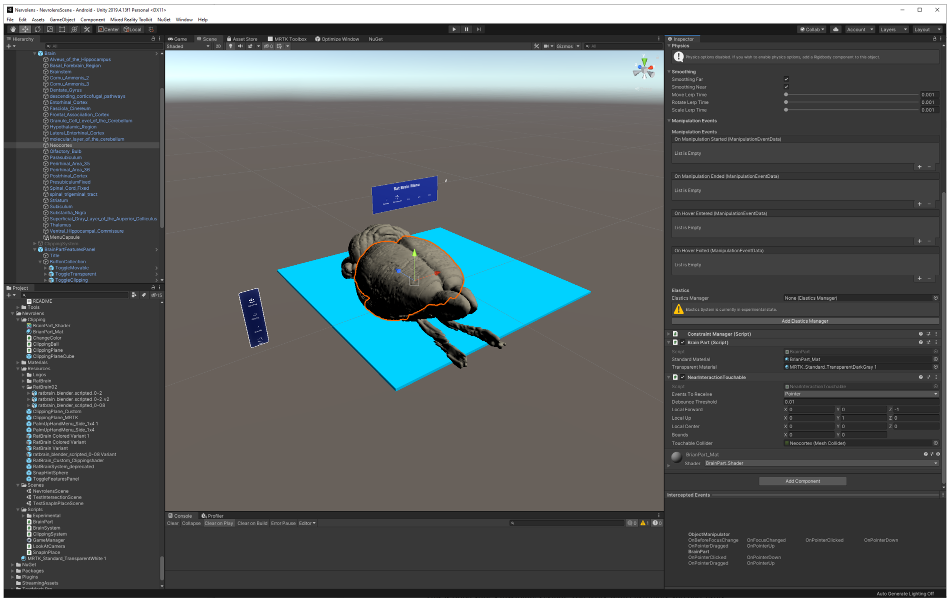


Figure 2.5: Unity 2019.4.13f1 running Nevrolens

Photon Unity Networking 2

Photon Unity Networking 2 (PUN) is the state of the art networking library for Unity. It can manage everything from voice chat to interaction over network.

Windows Device Portal

Windows Device Portal is an web-based application for managing devices running Window, like HoloLens. The HoloLens 2 hosts this application if it's connected to a network, so you can easily log in on the device and manage files, profile application or stream video from it.

Git

Git is a distributed version control system. Together with hosting on NTNUs self-managed GitLab it enables version control and cloud back-up of the project. While this is the most conventional version control system for any software project, using Git with Unity can be frustrating. Git is design for projects with only (mostly) small, human-readable text files, like a code base. A Unity project often as huge files, which Git does not support, and relevant changes can happen in binary files, which makes merging impossible. To mitigate this problems the use of Git LFS was needed. It is an extension for Git which enables storages of larger files. Together with enabling only human readable settings in Unity and a long gitignore file, Git with Unity was manageable. A good alternative to this use of Git is *Unity Collab*, which is Unity's answer to version control, it lack many features found in Git, but would probably be just fine for a project of this scope with a single programmer. However, I like Git very much and find the feature-set of Git to be very helpful.

GitKraken

GitKraken is a graphical Git management tool. What's more relevant here is its Kanban feature, or GitKraken Boards as their called. This enables synchronization of Kanban tickets and feature branches in Git, and generally makes my life easier.

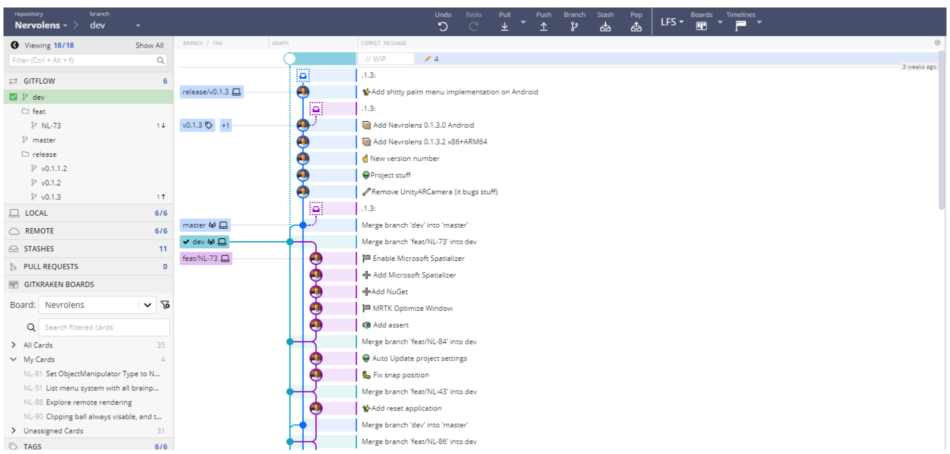


Figure 2.6: Git log in GitKraken, on the left you can see synchronized Kanban tickets

2.6 Related work

VRVisualizer

VRVisualizer is the name of the research product from the master thesis Elden (2017). It is a VR application running on the HTC Vive. While its features are similar to the goals of this project, the focus of Eldens project was to develop universal guidelines for scientific visualization in VR. That is far from this projects goals and thus its feature set, relating to interaction and exploration, is quite superficial.

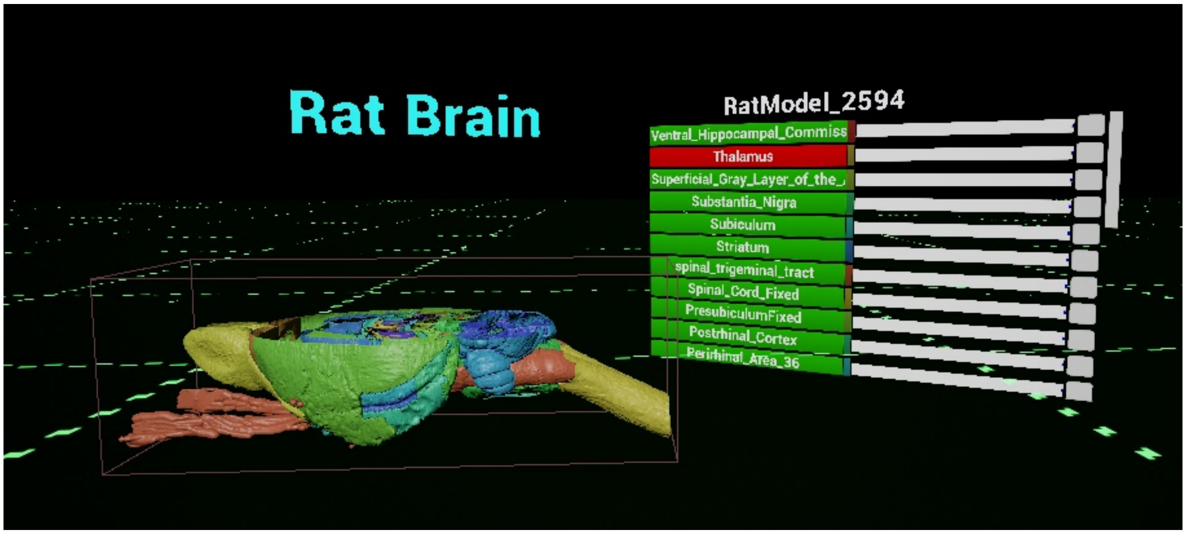


Figure 2.7: Cutting operation in VRVisualizer on HTC Vive

ITK-Snap

ITK-Snap is an open-source interactive computer application for navigating medical imagery. It is a convectional tool used to display, segment and label 3D neuroanatomical data. It supports a wide array of visual tools which helps in learning about the brain. However, the fact that it is purely displayed on a two dimensional monitor, means it limits the opportunity of spatial understanding.

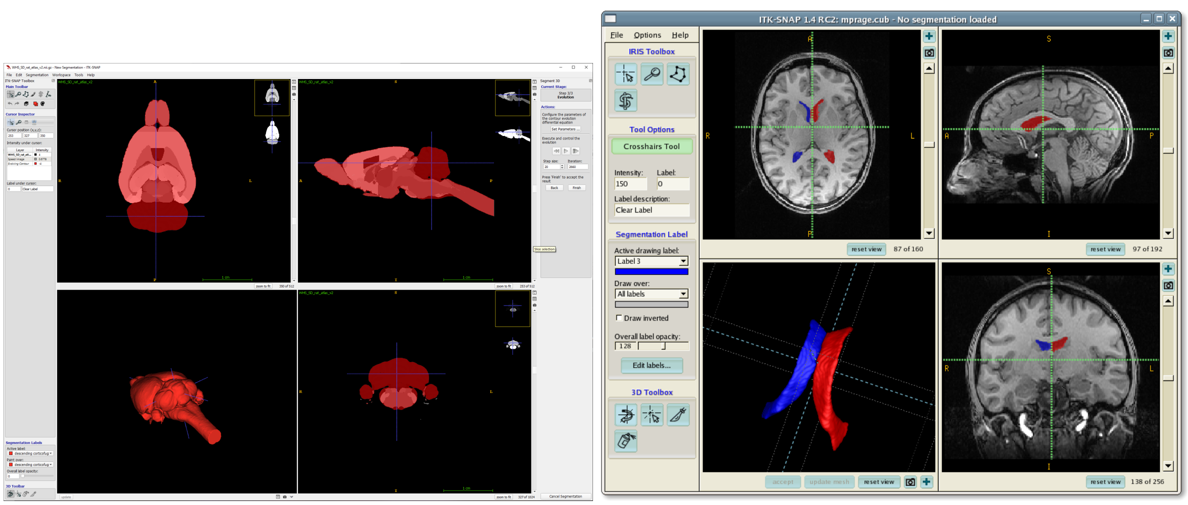


Figure 2.8: ITK-Snap, Waxholm Space Rat Brain (left), Human brain (right)

HoloAnatomy

HoloAnatomy is a application running on HoloLens, which aims to be a educational tool for anatomy including basic neuroanatomy. Wish-Baratz et al. (2020) has shown great promise in use of this application for education. While this application is really promising, it lacks some features we would like to focus on. There is no use of mobile devices for collaborative learning. And HoloAnatomy focuses on the macroscopic image of anatomy, and does not go into microscopic detail, which is something this project aims to do. Further, at NTNU and the Kavli Institute research and education using rat brains is normal an thus visualizing it it AR is also a focus.

Figure 2.9: HoloAnatomy

Chapter 3

Technical Design

This chapter will give a overview of the structure of the application as well as some choices taken when developing the research product, Nevrolens.

Unity Game structure

Within Unity a *Scene* consists of a *scene graph* which is a tree structure of `GameObjects`. By default a scene consist of a `Directional Light` lighting up the scene at its main light source and a `Main Camera` which is the view point of the running game. In addition, the MRTK library will add two objects to the scene graph, one called `MixedRealityToolkit` which contains configuration of the Mixed Reality features and systems. This is where input systems are defined and where control of spatial awareness and boundary detection is handled, in short all features and sensors of the HoloLens system or other AR system are defined and controlled here. The other object added by MRTK is the `MixedRealityPlayspace`, this encapsulates the `Main Camera`, but is lacking any useful documentation on what its purpose is. The name could be hinting at it being the parent of the *Playspace*, meaning all `GameObjects` in the game. However, even MRTK demos seem to ignore this object and thus it has not been used in this project either.

The functionality of the scene graph, other than organizing GameObjects, is that child objects inherit the position, rotation and scale of their parents, thus simplifying transformation of complex object. This naturally structures many systems, however in a AR application there can be many independent 3D objects floating in space. In addition, some objects are dependent not on their parent, but on a defined object `Transform`. Therefore, some organization is needed and some objects are placed by choice and convenience rather than any practical reason.

The main object of the application is the `BrainSystem`, this acts as the parent GameObject

for all objects defined by the application.
Thus, natural

Chapter 4

Development Process

4.1 Software Process

Even though I have developed the Nevrolens application by myself, I have strived to use best practices for a software development workflow. These practices have generally grown out of the the needs of a multi-developer setting, enabling simpler use of collaboration and version control. Though their value possibly increases exponentially by the number of team members, I have found value in the structure and clarity I find in the workflow.

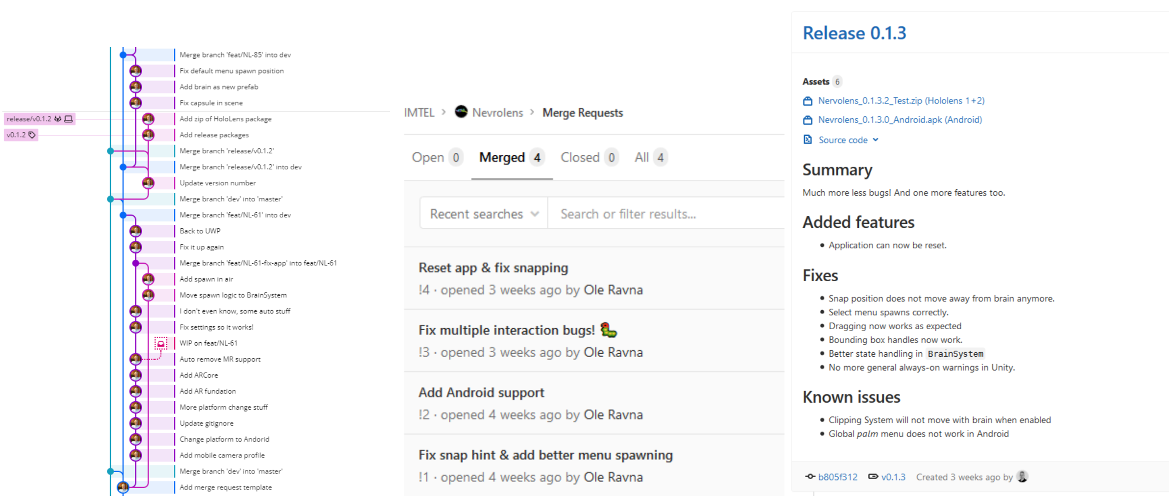


Figure 4.1: Feature branches, merge requests and releases.

My workflow is based on *Gitflow*, a workflow framework optimized for continuous software development. In short, this is just a very basic rule set for branch-naming and the sanctity of the master-branch (requiring merge requests of only product ready code), within the version control system Git. It does however act as a fundament which enables practices like rapid release cycles, because of the clearly define production ready state, and the integration with lean development technics like Kanban. This stems from the parallels between feature-branches in Gitflow and the *ticket* in Kanban. In practice, this means that tickets, with issues or new features for the app, are created on in the *Backlog* column of the Kanban board and are then moved to *Doing* column simultaneously as a feature-branch is created with the ID of the ticket, e.g. `feat/NL-42`. All of this is automated in the Git management tool GitKraken, which manages both the git-repo and the Kanban board.

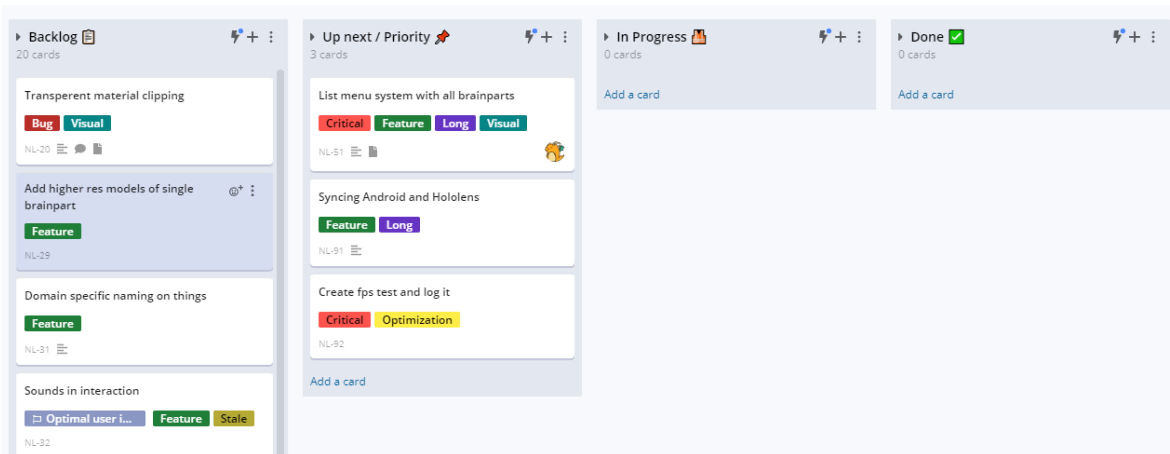


Figure 4.2: A snapshot of the Kanban board in GitKraken, after a development sprint, when completed tickets are archived (closed).
Note: *Priority* acts as pined tickets on *Backlog*, as the backlog tends to sizeable.

This workflow, by design, supports an agile development process. Agile approaches to software development are generally human-centered, valuing individuals and interactions over processes and tools[1], and focused on iterating rather than upfront planning. This is ideas which are beneficial for single-developer or small teams especially when developing for new platforms like the HoloLens 2. While the project aims for an agile approach, the sprint cycle core to the agile development, where stakeholders are involved for regular feedback, has, due to a number of factors like COVID-19, only really been done for one cycle. However, the steps taken for an agile workflow should enable more agile development for the master project.

[1] The Agile Manifesto https://agilemanifesto.org/

Chapter 5

Implementation

5.1 Requirements

The first meeting initializing the project took place at VRLab Dragvoll in early September, here I was introduced to the general background and the problem description of how Witter and others envisioned the use of AR for neuroanatomical education. It was explained how cadavers for education are difficult to acquire and therefore used quite sparingly. Another problem we discussed was related to the difference in medium between VR and AR. While the application VRVisualizer did have many of the features envisioned, and could have been a basis for further development. The fact that is was implemented in VR was problematic for the envisioned use cases. Being completely enclosed visually limits its use case in lectures and in any use case with collaboration in a physical space. Generally the loss of spatial awareness and eye contact as a result of using VR headsets was though of as an impediment for using VR for such an application. Thus, we had an outline of a neuroanatomical education tool in AR using the HoloLens 2 and concluded with some questions and requirements for the project:

1. Can the current VR dataset[1]be used in the HoloLens 2 AR environment?
2. If not, which steps need to be taken to use the segmented WHS rat brain to develop a suitable 3D model that can be used in AR?
3. Develop an optimal user interface for a single person to explore the rat brain as if the user is doing a dissection of a real brain.
4. Develop/test ways to make this a multiuser/shareable tool adequate in a teaching environment.

5. Explore ways to integrate microscopical data into the AR representation.

6. Describe/explore the feasibility to implement the system for Human neuroanatomy education.

Here items 1-4 were deemed critical for the project, while 5 and 6 were dependent on the progress made.
This meeting together with the list formed a clear problem description and can be seen as the initial discovery process of the project. Though the following period of exploring the newly arrived HoloLens 2 and its capabilities, we formed a set of *system requirements*. System requirements are descriptions of how a system should operate, what it should be able to do and the constraints of its operation. The requirements must reflect the stakeholders needs for the system (Sommerville, 2011). System requirements are generally split into functional requirements, which describe specifics of what the system (and its subsystems) should do, and non-functional requirements, which generally are descriptions of the user experience of the system as a whole. What follows are the system requirements decided on for the application:

Functional Requirements

1. **Implement a brain dissection tool in AR.**
   The app should render a brain at sufficient quality for educational use, and have the tools for creating a dissection experience in AR.

2. **The application must run in HoloLens 2 and at least one mobile platform**
   The ability to run a version for the app on multiple platforms is essential for the purpose of this project. While the main platforms are HoloLens and mobile, others may also be implemented in the future.

3. **Implement cross-platform collaboration over network**
   For the application to have value above a single user it is important that it can be used with a HoloLens and a more accessible platforms in a collaborative manner.

Non-Functional Requirements

1. **Medical students should find educational use for the app.**
   It is critical that there is educational value in the application.

[1] Referring to VRVisualizer by (Elden, 2017).

2. **The application should be usable without outside guidance.**
   The app should have a clear and understandable design, such that a new user should be able to navigate the app by them self, even with minimal experience with AR.

3. **All relevant usability criteria for a mixed reality app should be met.**
   We should work to not fall under the 'meets' criteria on any relevant metric in the App quality criteria[2]. This includes criteria on; FPS, spatial anchoring and view comfort.

5.2 Development phase

0th iteration: Initializing application, importing and simplifying brain model

The first phase of development started by acquiring the surface model of the WHS rat brain created by Elden (2017). This was done by simply moving the FBX files from the VRVisualizer application and to a new Unity project. After initializing MRTK by following their Getting Started documentation, the application was ready to deploy on the HoloLens 2. This resulted in a barely running application as the polygon count of the brain model was orders of magnitude larger than what is recommended to maintain adequate performance on the HoloLens 2, which is in the order of 100,000 polygons shown in Figure 5.1. The model used by Elden was scaled to run on workstation computer outputting to an HTC Vive, and thus his model was reduced from a original 16 million polygons to around 3 million. The HoloLens 2 runs all calculations on-device on a mobile ARM-based processing unit and naturally the brain model created for rendering on a dedicated workstation graphics card had to be further scaled down. This downscaling was first experimented with doing at run-time dynamically on-device using the library *UnityMeshSimplifier*. It was quickly de-

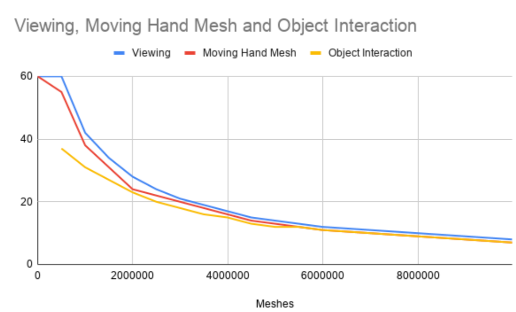


Figure 5.1: Figure showing frame rate as a function of polygon count on HoloLens 2.
Credit: Fologram

[2] https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/app-quality-criteria

termined that this was not a viable solution both because of untenable processing time, but also because the simplified result had a huge impact on quality of model, hinting at the performance optimization that had to be done on the simplifier algorithm to be able to execute at run-time. The next and final solution for downscaling was to use the *decimate* modifier in Blender. *Incremental decimation* is a mesh simplification algorithm which trades some speed for higher mesh quality, in contrast to the *vertex clustering* presumably used in UnityMeshSimplifier which prioritizes speed in such a way that topology is not preserved. Within Blender functionality for simple application of the modifier to all objects in a tree-structure was not found, or understood to exist, so a script applying the decimate modifier with a given ratio was written, see Listing 5.1. The `ratio` parameter is a value between 0 and 1, representing the scaling of the resulting mesh' polygon count.

```python
import bpy # importing the blender python library

def decimate(ratio, replace = True):
    # Finds all objects and filters irrelevant objects from the FBX
    brainparts = [n for n in bpy.data.objects \
        if n.name not in ("Camera", "Light")]

    for part in brainparts:
        mod = part.modifiers.new(type='DECIMATE', name='Decimate')
        mod.decimate_type = 'COLLAPSE'
        # Sets the specifies strength to the decimate operation.
        mod.ratio = ratio
# Calls function with given decimate strength.
decimate(0.08)
```

Listing 5.1: Blender script applying a decimate modifier to all relevant objects in a scene.

The resulting decimated model, even at the ratio of 0.08, was visibly nearly indistinguishable from the original model when looking at them through the HoloLens 2 display which, as described in section 2.4, is somewhat blurry. Figure 5.2 shows the difference as seen in the Unity editor. Ultimately, a decimation ratio of 0.08 was chosen as a compromise between detail and performance being about 300,000 polygons, this compromise will be discussed further in section 8.1. At this stage requirement 1 and 2 in the initial requirements from section 5.1 could conclusively be answer as possible and completed.

1st iteration: MVP

Having a surface model of the brain running reasonably well on the HoloLens 2, the next step in developing the application was to implement basic AR-based interact features. The

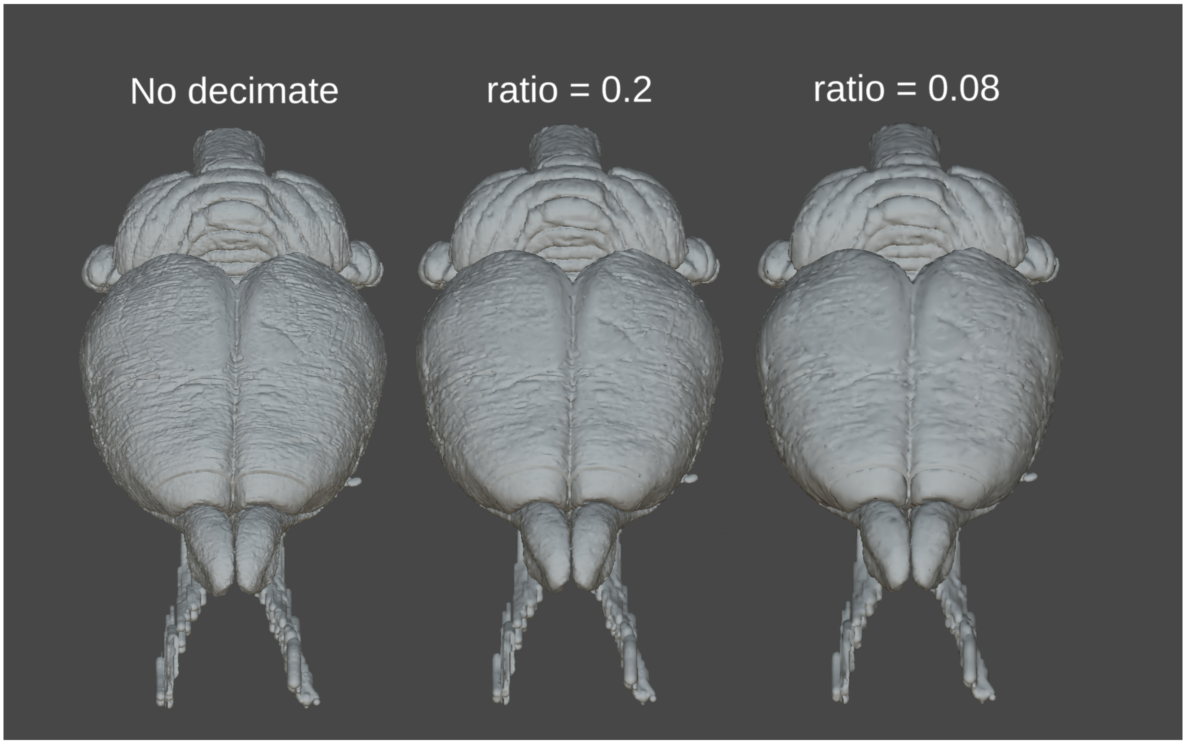


Figure 5.2: WHS rat brain models with decreasing polygon count.

brain model consist of an empty parent object with 29 children each containing the mesh of a delineated brain structure, see Figure 5.3. Adding the `Object Manipulator` component from MRTK and a standard Unity `Mesh Collider` component to each child in the brain model allows for picking apart the brain. This is done by grabbing and moving each separate structure with a MRTK defined *pointer*, this is the logical abstraction for the simplest interact handling with HoloLens 2 giving the user a virtual laser pointer from their finger. The resulting action can be seen in Figure 5.4.

An apparent problem at this stage was that thought the brain structures are separate objects, they were difficult to visually distinguish from each other. A script which took all child objects and applied a random color to each was written and placed on the parent object, thus quickly giving some visual separation of the structures. While implementing this feature, the *material* of each child was changed from Unity's default material to a *MRTK Standard* material. Materials are the way Unity handles rendering details for each object, this is where shader, texture and general rendering options are configured. The MRTK Standard materials is a set of materials using the the `MixedRealityStandard.shader` shader, this shader is optimized for MR use, and superficially for HoloLens, and is meant for fulfill all shader-needs when developing for these platforms.

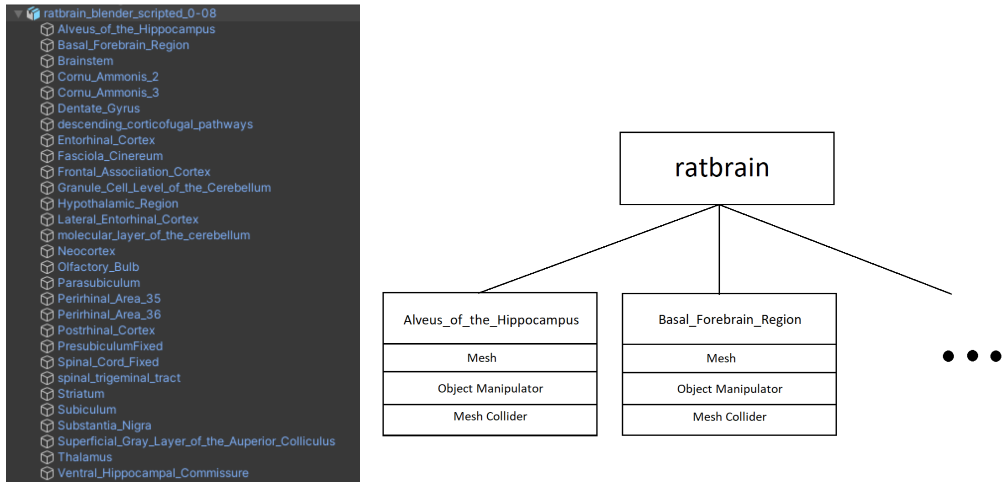


Figure 5.3: The tree structure of the Unity `GameObject` of the brain model.

With a some basic visibility and manipulation features for the brain model, the next natural step was tackling the system requirements, specifically the first functional requirement, implementing brain dissection. A *clipping* shader was written and implement to work with the brain, giving more control over the feature than using MRTKs prebuild clipping feature, but seeing as it was not possible to combine a custom shader with MRTK optimizes feature set for AR rendering, the custom clipping implementation was abandoned in favor of MRTK, using the aforementioned MixedRealityStandard shader. Clipping has the effect of removing vertices by some defined function, and by using a prebuilt clipping plane prefab and declare on which meshes is should act, a dissection affect was created. A handle for manipulating the plane was added for ease of use, by dragged a ball the plane would move such that it was a fixed distance from the ball and perpendicular to the line between the ball and the center of the brain.

Further, a hovering menu displaying the name of the last grabbed brain structure and buttons for the actions moving, transparency and dissection was implemented. This was created by modifying a MRTK prefab and updating its name based on the name of the `GameObject` the `pointer` targeted while dragging, at the same time a selection lighting effect as applied by simply enabling `Border Lighting` in the MixedRealityStandard shader. Unity's layer functionality was used to ensure that it was a brain structurer being dragged. One last feature implemented at this phase was a tap-to-spawn feature, this entailed using the `pointer` to tap on the physical space, and using spatial awareness to place the brain at the locations the the user tapped. In MRTK spatial awareness is enabled by default and its mesh can be identified by a predefined Unity layer, thus Listing 5.2 shows a simplified implementation of the `EventHandler` method, `OnPointerDown` which spawns the brain

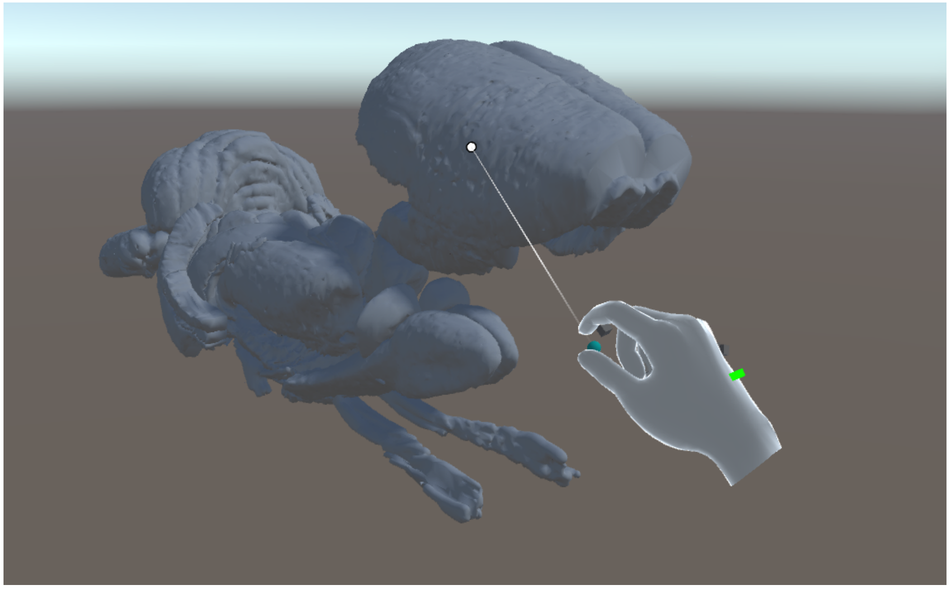


Figure 5.4: Grabbing the neocortex brain structure with a MRTK pointer in the Unity editor.

if the pointer is hitting the spatial awareness mesh and enables border lighting and menu text if it hits a brain structure.

```
void OnPointerDown(MixedRealityPointerEventData eventData)
{
    if (!HasTarget(eventData.Pointer))
        return;
    Vector3 hitPoint = GetHitPoint(eventData.Pointer);
    GameObject target = GetCurrentTarget(eventData.Pointer);

    switch (target.layer)
    {
        case SpatialAwarenessLayer:
        {
            if (BrainHasNotBeenSpawned())
                SpawnBrainAt(hitPoint);
        }
        case BrainStructureLayer:
        {
            if (selectedStructure != null)
                DisableBorderLighting(selectedStructure);
            EnableBorderLighting(target);
            SetMenuText(target.name);
            selectedStructure = target;
```

```
            }
        }
    }
```

Listing 5.2: A simplified version of the event function called when a `Ponter` is clicked.

The application was deployed for HoloLens 2, and was a first MVP demo of the research project. Figure 5.5 shows spawning of the brain model from image 1 to image 2 in the top row, notice the pointer on the table in image 1. Image 3 illustrates the clipping feature, while image 4 has a user taking out the *cornu ammonis 3* brain structure.

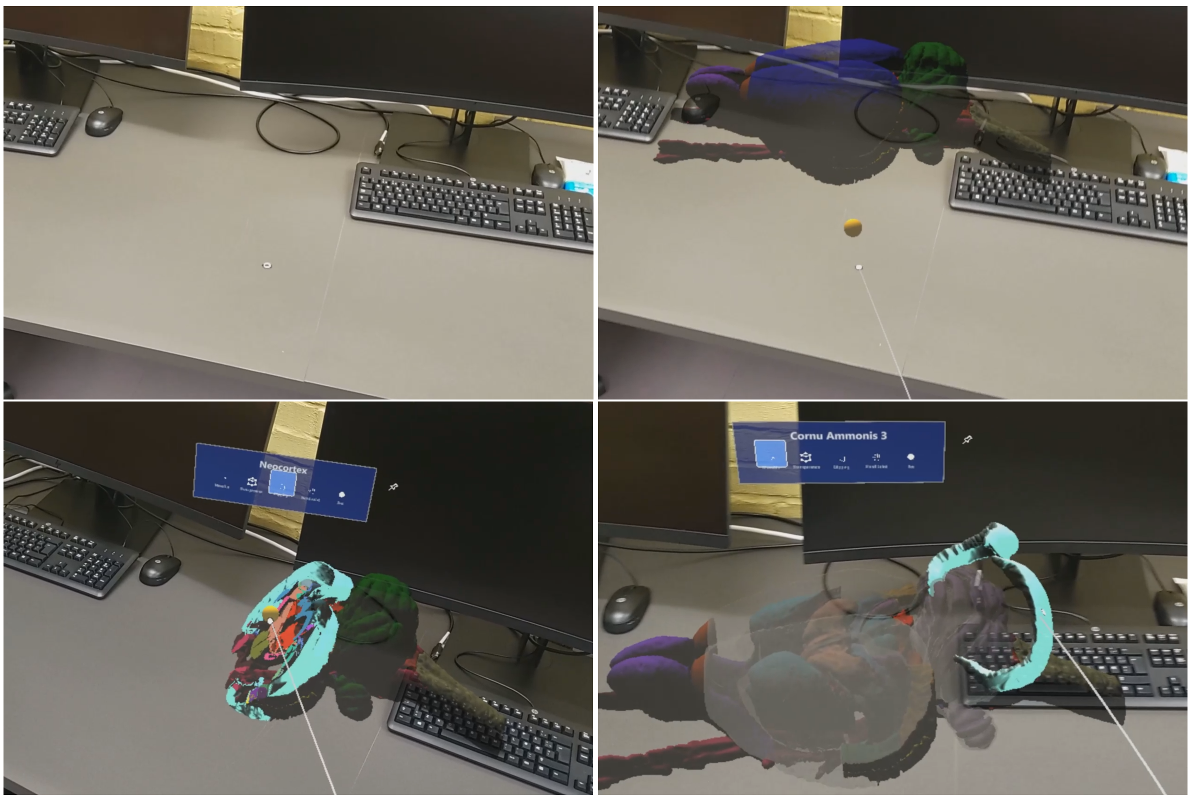


Figure 5.5: The first demo of the application.

Next iterations: Continuing development

This section will give a overview of the continued progress in developing the application and highlighting some specific features.
The first demonstration of the application was done over video conference, with a pre-recorded YouTube video, demonstrating the features of the application.

UI, menus and stuff

Feedback from demo

Coloring brain

Snapping

Porting to Android

The application was originally developed for HoloLens 2, but other platforms were always in mind and because of the use of Mixed Reality Toolkit, deploying to other platform was relatively easy, within the documentation for MRTK, there was guides for how to build for Android. Interaction on Android is more complicated though. Because all interaction happens on screen additional effort has to be laid down to implement a good user experience on smartphone.

Volumetric dissection plane

3rd iteration: Implementing network

4th iteration: Final product

Chapter 6

Deployment

Installation of Nevrolens

This section will be a guide for installing Nevrolens on HoloLens 2 and Android from packages hosted on GitLab. By following the instructions for HoloLens 2, this should also work for HoloLens 1, but has limited testing on that platform as it was not a focus of this research.

Deploy to HoloLens 2

1. **Go to the release page**
   Found here: `https://gitlab.stud.idi.ntnu.no/olevra/nevrolens/-/releases`

2. **Choose a release**
   Preferably the topmost and latest. This research ends on is version 0.3.3.

3. **Download the HoloLens zip package**
   Under *Packages* click on the package for HoloLens (1 and 2) to download it.

4. **Unzip file**
   Open the downloaded ZIP-file and extract it.

5. **Open the Windows Device Portal for HoloLens**
   Guide by Microsoft: Using the Windows Device Portal

6. **Install appxbundle**
   Under *Views* / *Apps* click `Choose File` and locate the APPXBUNDLE-file inside the folder extracted from the ZIP-file. Then click `Install`.

Deploy to Android

1. **Go to the release page on your Android device**
   Found here: `https://gitlab.stud.idi.ntnu.no/olevra/nevrolens/-/releases`

2. **Choose a release**
   Preferably the topmost and latest. This research ends on is version 0.3.3.

3. **Download the Android APK-file**
   Under *Packages* click on the package for Android to download it.

4. **Open the downloaded file.**
   Your device will ask for your permission to install an application from a unknown source. Accepting this, the device will start installing the application.

Chapter 7

Results

7.1 Nevrolens

Nevrolens is the name I've given the application which is the research product of this project. It's an artistic (read; incompetent) combination of the Norwegian word Nevroanatomi and HoloLens. It is a AR application running on HoloLens 1, HoloLens 2 and Android.
The application is focused on delivering a single user experience, with features as cutting planes, scaling, moving brain parts and transparent brain parts. Figure **??** show these features running on HoloLens 2, while Figure **??** show them on Android. It is packages and released on GitLab at `https://gitlab.stud.idi.ntnu.no/imtel/nevrolens`.

7.2 Results

Because of the COVID-19 pandemic no user testing has been done this semester, in fact no medical students or professionals have tried the application in-person. Thus, it has been difficult to do formal interviews or gather much feedback, especially regarding interaction. This project is the preliminary work for my master thesis next semester and result gathering will naturally be a much more in focus then. And though no user testing has found place, we have arranged live demoes with Windows Device Portal over Zoom, which have generated useful feedback. In one such demonstration, I wore a HoloLens 2 and use Nevrolens with guidance from a neuroscientist to extract related regions of the brain and was lectured on their role in behavior. The feedback on its use for a single user, was that there should be a global list menu to toggle different features on each brain part, that there should be ability to increase resolution of a single brain part and some way to visualize microscopical data.

Figure 7.1: Nevrolens v0.1.3 on HoloLens 2

Figure 7.2: Nevrolens v0.1.3 on Android

Chapter 8

Discussion

8.1 Performance

Chapter 9

Conclusion

9.1 Limitations

This project rapport is based on work from a 15 credits course, thus there has been time constraints managing the project with other courses and exams. Combined with the busy schedule of the neuroscientists at the Kavli Institute, it has been challenging to gather feedback in the relatively short time frame of this project. Of course, the COVID-19 pandemic has made physical sharing of the headset problematic, and thus user testing has been difficult. In total, there is a very thin basis for results in this project, but this will hopefully be turned around for the master project.

9.2 Discussion

This project has mainly focused on implementing the application answering Sub-RQ1. As such this can be seen as a minimal viable product for exploring Sub-RQ1, and in that capacity it has shown great promise. The basic tools for dissection is implemented and the medical academics at Kavli Institute and UiO sees great potential in its use. As for the Sub-RQ2 and 3, there is less work done. However, by building the application for Android, I have shown that the possibility for cross-platform collaboration is open. And it will be worked on for the master thesis, by implementing networking with **??**. Sub-RQ3, has not seen any concrete development, I do have some ideas of how it could be implemented with textures generated from a volumetric brain model.

9.3 Conclusion

The state of the project now is a small scale proof of concept for a AR application supporting teaching of neuroanatomy and dissection for medical students. There is still no collaboration or microscopic data visualization in the app. However, the biggest problem with the state of the project is the lack of concrete feedback from medical user groups. It has been received very positively by the neuroscientists who has seen the application in use, and they are awaiting more news on the project. The need for exact user testing and user feedback is still large. The project is however progressing nicely, and with further development I believe this project could create impressing results.

9.4 Further Work

There is still a lot of work left on this project, and it will be continues by me. While there is numerous features and fixes which are waiting in the backlog I would like to focus on the overall picture and what has to be in focus to create an collaborative educational experience in AR.

- **Focus on usability**
  Implement better guidance and affordance such that anyone can use the application. To manage this it will be essential to have user testers and testing with medical user groups. This will give feedback on what makes sense and what doesn't.

- **Collaboration; implement cross-platform networking**
  Networking tools will be required to create a collaborative experience, this will be done using **??** to synchronize the HoloLens 2 application with the Android application.

- **Visualize volumetric data**
  While the HoloLens 2 does support volumetric rendering, I am pretty sure it is not in an adequate resolution for this project. It will however be explored. Other than that volumetric data could be used as 2D textures mapped on the clipping plane used to cut the brain.

- **Explore ways of using other brains**
  Appendix B explains the steps taken by Elden (2017) to use medical imagery in VRVisualizer. An exploration into whether there is a more elegant way of doing this, should be done.

A general focus when continuing this project should be gathering more user data in the form of user tests and interviews, there should be a focus on establishing whether this project can improve learning outcomes for medical students.

9.5 Future Work

This section will lay down suggestions for how to further the project. These are based on the authors experience with the research project and on discussions with the neuro specialists and medical students testing the application.

Import new data sets

As described in section 2.3 the WHS rat brain is under continuous development and the the near future there will be released a forth version of the brain model, with improved delineation. It is a high priority wish from the Kavli Institute to use this brain model in the application.
To import a WHS brain model as a geometric model is a complicated process, which has been explained by Elden in Appendix B. The process of exchanging the geometric model used now with a new one however is trivial, but time consuming. A considerable improvement to the application would be the ability to drop in a new model, preferable at run time such that the application wouldn't need to be build and deploy for each model change. This would enable future WHS versions to easily be added and even other models like human brains could be switched between.
Another essential feature to reduce the need for new builds is to enable configuration in-app. The application uses three different text files which saves the configuration of clustering, infoboard and labels, all of these should either be possible to upload at run time or configuration with settings UI in-app. A logical step could be to use the JSON format for these files instead of the custom parsing done now.

Improve networking

A critical improvement to the application would be to have the initial scene of the application give the user an option of creating a room, joining an existing room or playing offline. The application as of now will automatically behind the scene, join the existing room or create one if there is none. This gives users less control but more frictionless when testing collaborative features. In a full scale application the user control would probably be preferable.

The networking solution in the application has a considerable amount of bugs and unexpected behavior, this is probably something that is difficult to completely circumvent, but an effort to restructure or fix this should be done.

Voice chat

When collaborating in-app remotely the ability to talk to each other would be a great feature, which for a full scale application would be a high priority. This can be done with by using Photon Voice for PUN 2.

Platform specific input and UI

There are few limitations in which platforms the application can run on. However, each platform comes with it's own needs for specific input types and UI. The application has been always been developed with HoloLens 2 as the highest priority and that can be seen in most choices taken conserving UI and input handling. Increasing the user experience on Android and even Windows or WebGL (the application is buildable for both, but needs input handling to be usable) should be a priority. Feedback from user testers gave indication that the augmented reality in the android version did not provide an improved user experience and thus disabling camera and spatial features in Android could be seen as an improvement, which could be extended to a desktop application. A future version of the application running on the web, with WebGL, would probably be possible and further the goal of accessibility.

Appendix A

Acronyms

NTNU Norwegian University of Science and Technology

AR Augmented Reality

MR Mixed Reality

XR Extended Reality

VR Virtual Reality

FPS Frames per second

HMD Head-mounted display

WHS Waxholm Space

GPU Graphics Processing Unit

SDK Software Development Kit

MRI Magnetic Resonance Imaging

DTI Diffusion Tensor Imaging

Appendix B

A geometric model of the rat brain

This is a section from Elden (2017), the master thesis about VRVisualizer the VR application this project is loosely inspired by. The section explains how Elden extracted a geometric model of the rat brain from the medical models which is high fidelity volumetric data. I have included it in this report because it gives insight into a specific solution to a problem I face and it explains how a resource I use in this project was created.

5.2 Exporting segments of a rat brain atlas as geometry for the Rat Brain model

The geometric meshes used for the rat brain model were extracted from a volumetric and segmented atlas. IKT-SNAP was used to export each segment of the brain as an STL file. These geometric meshes were then opened in Blender3 to be converted to OBJ or FBX files. 3DS Max imported the models and performed all modifications made to the geometry and structure. ITK-SNAP requires three files to segment and label the models; the atlas and a segmentation file, both stored as NII files, and a LABEL file for the labels. When all files are loaded the program lets the user select a segment to export and generate a geometric hull along the boundary of the segment. Due to instability experienced with 3DS Max using all 16 GB of RAM available on the computer used for development, Blender was used to first convert the files to FBX files. These FBX files caused no issues when imported into 3DS Max. Since these meshes were too detailed, they needed to be reduced and transformed in 3DS Max. The meshes were reduced such that the entire model consisted of 4.5 million triangles. Most of the meshes had to be transformed such that each segment was where it should be inside the model. For some reason the exported meshes were of several relative scales and heights and a lot of manual work went into moving and scaling the meshes to match the volumetric model seen in ITK-SNAP. Properly processed, the model was exported as an FBX file and sent to UE4.

Bibliography

Elden, M. K. (2017). Implementation and initial assessment of vr for scientific visualisation: Extending unreal engine 4 to visualise scientific data on the htc vive. Master's thesis, University of Oslo.

Hawrylycz, M., Baldock, R. A., Burger, A., Hashikawa, T., Johnson, G. A., Martone, M., Ng, L., Lau, C., Larsen, S. D., Nissanov, J., Puelles, L., Ruffins, S., Verbeek, F., Zaslavsky, I., and Boline, J. (2011). Digital atlasing and standardization in the mouse brain. *PLOS Computational Biology*, 7(2):1–6.

Milgram, P. and Kishino, F. (1994). A taxonomy of mixed reality visual displays. *IEICE Trans. Information Systems*, vol. E77-D, no. 12:1321–1329.

Moro, C., Štromberga, Z., Raikos, A., and Stirling, A. (2017). The effectiveness of virtual and augmented reality in health sciences and medical anatomy. *Anatomical Sciences Education*, 10(6):549–559.

Oates, B. (2006). *Researching Information Systems and Computing*. SAGE Publications.

Papp, E., Kjonigsen, L., Lillehaug, S., Johnson, G., Witter, M., Leergaard, T., and Bjaalie, J. (2013). Volumetric waxholm space atlas of the rat brain for spatial integration of experimental image data. In *Front. Neuroinform. Conference Abstract: Neuroinformatics 2013*.

Papp, E. A., Leergaard, T. B., Calabrese, E., Johnson, G. A., and Bjaalie, J. G. (2014). Waxholm space atlas of the sprague dawley rat brain. *NeuroImage*, 97:374 – 386.

Sielhorst, T., Feuerstein, M., and Navab, N. (2008). Advanced medical displays: A literature review of augmented reality. *Journal of Display Technology*, 4(4):451–467.

Singh, V. and Kharb, P. (2013). A paradigm shift from teaching to learning gross anatomy: meta-analysis of implications for instructional methods. *Journal of the Anatomical Society of India*, 62(1):84 – 89.

Sommerville, I. (2011). *Software Engineering*. Pearson Education, Inc., 9th edition.

Toga, A. W. and Thompson, P. M. (2000). 40 - image registration and the construction of multidimensional brain atlases. In BANKMAN, I. N., editor, *Handbook of Medical Imaging*, Biomedical Engineering, pages 635 – 653. Academic Press, San Diego.

Wish-Baratz, S., Crofton, A. R., Gutierrez, J., Henninger, E., and Griswold, M. A. (2020). Assessment of Mixed-Reality Technology Use in Remote Online Anatomy Education. *JAMA Network Open*, 3(9):e2016271–e2016271.