AR-assisted craniotomy planning for tumor resection

Joost Wooning

AR-assisted craniotomy planning for tumor resection

by

Joost Wooning

to obtain the degree of Master of Science at the Delft University of Technology, to be defended publicly on January 18th 2021 at 13:00.

Student number:	4245318	
Thesis committee:	Prof. dr. A. Vilanova,	TU Delft, chair
	Dr. R. Marroquim,	TU Delft, supervisor
	Dr. T. van Walsum,	Erasmus MC, supervisor
	Dr. F. M. Vos,	TU Delft

Preface

This report was created as graduation thesis for the Master of Science degree in Computer Science at the Delft University of Technology. It was conducted at the Computer Graphics and Visualization research group of the TU Delft and was a collaboration with the Biomedical Imaging Group Rotterdam research group at the Erasmus University Medical Center. Daily supervision was provided by Ricardo Marroquim, from the TU Delft, and Theo van Walsum from the Erasmus MC. Additionally, in the thesis committee, Anna Vilanova took part as chair and Frans Vos as external member. I would like to thank them all for their participation in this thesis, especially my daily supervisors for their continuous support in completing my thesis.

I would like to thank all the participants of the user studies, who were able to take time out of their busy schedule to provide a lot of useful feedback. The enthusiasm from some of the participants really helped in keeping me motivated. Especially the neurosurgery staff I spoke with, who were able to provide a lot of useful feedback during the project and really helped shaping the final visualization.

Completing this thesis was a great challenge, changing goals, underestimations of the amount of work and the coronavirus caused some major delays. Especially the coronavirus, that hit The Netherlands in the middle of my thesis project, made it difficult to communicate and work with everyone and was a continuous stress factor during the project. Currently, the coronavirus has not gone away yet, making this project more difficult until the very end. Although in the end, I am happy with the final result of this thesis. Therefore, I would finally like to thank my friends and family for supporting me throughout the time of this thesis. While my plan had been to finish my study time much earlier, you kept me motivated and caused me to keep moving forward.

Joost Wooning
Delft, January 2021

Abstract

A craniotomy is a procedure were a neurosurgeon has to open the skull to gain direct access to the brain. When a brain tumor has to be removed from a patient, the craniotomy position is of great importance. This mostly defines the access path from the skull surface to the tumor and thus also what healthy brain tissue will be removed to access to the tumor itself. To minimize the amount of important brain structures that are removed, the craniotomy has to be carefully planned. This is a complex procedure, where a neurosurgeon is required to mentally reconstruct spatial relations of important brain structures to avoid these as much as possible.

We propose a visualization using augmented reality which may assist in the planning of a craniotomy. In this visualization the goal is to show important brain structures aligned with the physical position of the patient. This should allow better perception of the spatial relations of these structures and thus assist the neurosurgeon. Additionally to the visualization of the structures, we created a heat map that is projected on top of the skull. This should give a quick overview of in which areas there are many important structures between the tumor and the skull surface, and should therefore be avoided.

User studies were conducted amongst neurosurgeons and surgeons from other fields to evaluate the proposed visualization. We found that many of the participants indeed thought that the visualization can assist in surgery. For the specific case of craniotomy planning, several improvements have to be made on the heat map before it can be useful. Nevertheless, the visualization of the structures in itself can assist neurosurgeons in the planning of a craniotomy. Although more work has be performed at practical aspects of the visualization to make it ready for clinical experiments.

Contents

1 Introduction

In this work we propose a visualization using augmented reality (AR) to assist in the planning of a craniotomy for tumor resection.

1.1. Motivation

When a brain tumor has to be removed through surgery, a surgeon has to have access to the brain through an opening in the skull. The procedure of creating this opening in the skull is called a craniotomy.

Often, most of the healthy tissue between the skull opening and the tumor will be removed to access the tumor. To minimize damaging important brain tissue, this access path needs to be carefully planned. Therefore, to plan the craniotomy, a neurosurgeon looks at important structures in the brain which should be avoided. These are structures such as the vessels and certain brain areas and fibers linked to specific neurological functions, these structures are segmented before the procedure from an MRI scan of the patient.

During the procedure, the surgeon plans this access path from the skull surface to the tumor. Currently, 2D slices of the segmented MRI scan are used. A surgeon will then have to mentally reconstruct these slices to 3D structures to plan the craniotomy. This is not trivial, and less experienced neurosurgeons might benefit from being able to see a 3D model.

When the craniotomy has been planned, the plan has to be transferred to the head of the patient. A neuronavigation system is used, which allows to locate specific positions in the MRI data on the physical position of the patient. The craniotomy location is drawn on the head of the patient, and finally, the actual procedure can be performed.

Augmented reality may assist with this procedure in several ways. Using a stereoscopic display models of the brain structures can be shown in 3D, increasing the perception of spatial relations between the structures. Additionally, AR allows to see the patient as well, giving greater context of the operating site. Since the structures can be shown inside the head, aligned with the patient, no additional transfer step is needed, and the craniotomy plan can directly be drawn on the head of the patient. Furthermore, an AR device might also replace the neuronavigation system. Using the built-in camera of the device the position of the patient can be tracked using fiducial markers.

1.2. Objective

Following the work at the Erasmus MC of Incekara et al. we investigate if AR can assist in neurosurgical procedures [23]. In this study, a proof of concept was shown using a HoloLens to see if it could be compared to traditional neuronavigation methods. A HoloLens was used to transfer a planning of a craniotomy to the head of a patient. In this work we instead created a more interactive visualization using a HoloLens, of which the goal is to assist in the planning of the craniotomy.

The visualization shows surface models of important brain structures that should be avoided. We also created a heat map of these structures which is projected on the skull surface, this should give a neurosurgeon a quick overview of which areas should be avoided in planning a craniotomy.

User studies were performed to evaluate the visualization. Two neurosurgeons have participated along several other surgeons from different fields. Useful insights were gathered, which can be used to further improve the method.

Another research topic in the Erasmus MC is how 3D models can be aligned with a patient. To achieve this, the position of a patient has to be tracked by the system. Some previous studies used a conventional navigation system to provide the tracking [7, 28, 47]. Instead, in this work we investigated if the HoloLens can be used as a navigation system. The built-in camera of the AR device is used to track fiducial markers which can be attached to the patient. This allows to track the patient position relative to the AR device and display aligned 3D models of the patient.

1.3. Contributions

Several contributions were made in this work:

- The main contribution of this work is the proposal of an AR visualization to assist in the planning of a craniotomy for tumor resection. Besides the visualization of important brain structures, a heat map was created which should further assist in the planning. The visualization was evaluated through a user study amongst several domain experts.
- Evaluation of marker tracking with a HoloLens to display models inside a patient. Although the method itself is not novel, the evaluation of the tracking provided useful insights in the accuracy. This might provide useful information to further improve the accuracy of such a tracking method.
- The HoloLens is not able to display dark colors. When a color map is used this can cause color maps to give incorrect insights in the data. We evaluated the performance of some color maps where no dark colors can be shown.

1.4. Outline

This work is organized as follows. Chapter 2 discusses the background and related work. Chapter 3 explains how we used markers to align models to the physical position of a patient. In Chapter 4 we present how we created the visualization. Chapter 5 reports some preliminary experiments that were performed. Chapter 6 reports how we performed the user evaluations and presents the results. Chapter 7 discusses the results that were presented and gives suggestions for future work. Finally, in Chapter 8 we give some conclusions.

2 Background and Related Work

In this chapter we discuss the background and related work. The first section will go into the clinical background necessary for this thesis work. In the next section we explain what augmented reality is and the challenges that come with creating an AR application. Finally, we mention previous work related to visualizations for craniotomy planning.

2.1. Clinical background

A craniotomy is a procedure where an opening in the skull is created for direct access to the brain. To plan this procedure extensive knowledge about the structures in the brain is required. Imaging techniques are used to create 3D images of the head, from these images specific structures can be segmented which are used to determine the access path to the tumor. During the procedure the image data has to be aligned to a physical location of the patient, surgical navigation systems are used for this.

2.1.1. Surgical navigation

With surgical navigation a surgeon has the ability too 'see' inside the patient and to navigate to the area that needs to be treated, see Figure 2.1. This can be used to perform minimally invasive procedures, where only small incisions are used instead of open surgery. With minimally invasive procedures small tools are inserted into the body to perform a procedure, with the benefit of faster recoveries and reduced hospital cost [39].

Navigation systems require a 3D image of the body, which is usually created preoperatively using techniques such as CT or MRI. With pre-operative imaging it is possible that the patient body deforms after the image has been acquired. However, for navigation this is only a problem with non-rigid body parts. Our focus area is the head and since the structures here are relatively rigid, deformation is not a problem for us.

Navigation systems also require that the position of the patient body is known. Often a procedure uses specific tools which must also be tracked and displayed in the same view as that of the model of the patient. Tracking of physical objects can be done in several ways, three often used methods are:

- *Electromagnetic (EM) tracking:* See Figure 2.2.a. Works with a field generator that generates an EM field. The tracking sensors are small coils, a few millimeters in diameter, and can therefore be attached to all kinds of objects but need to be wired. A downside of EM tracking is that interference can occur from metal objects and electrical devices.

- *Optical tracking:* See Figure 2.2.b. Cameras are used to track optical markers, which consist of small spheres that emit or reflect infrared (IR) light. Using stereo cameras or a time of flight sensor the depth of the spheres can be measured. Placing several of these spheres in a known configuration allows to calculate the rotation of the combination of the spheres. A downside of optical tracking compared to EM tracking is a line of sight between the camera and sensor is required.

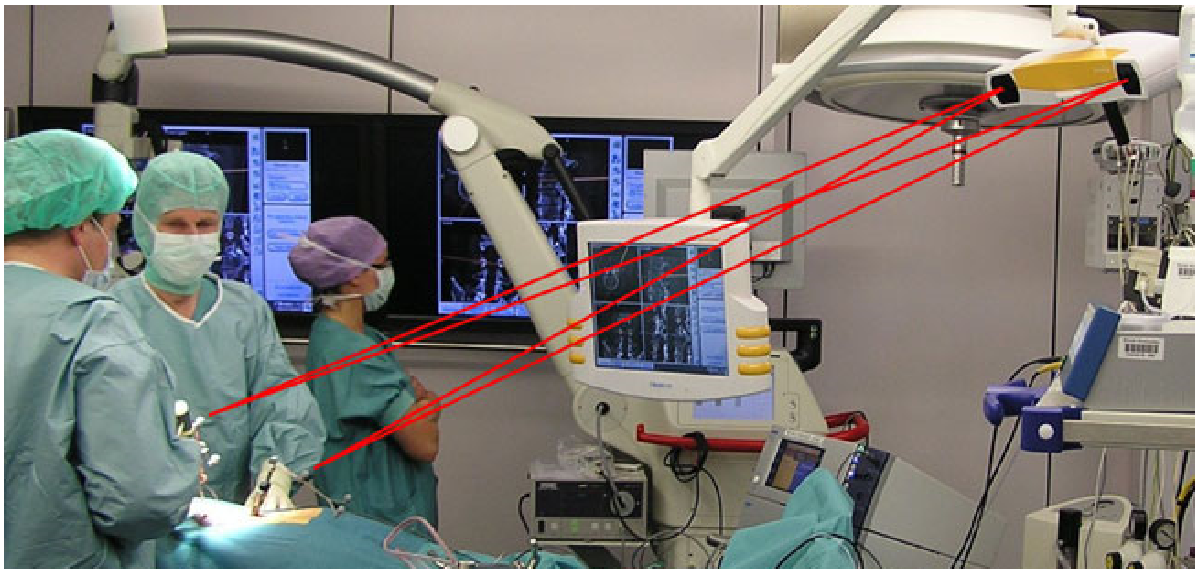

Figure 2.1: Use of a surgical navigation system in practice. The camera system can track IR-reflective spheres attached to the patient and the tool held by the surgeon. On the displays slices of the 3D patient image are shown. Because the relative position of the tool to the body is known, the location of the tool can also be shown in the same view as the patient image. (Mezger et al.)

- *Visual markers:* See Figure 2.2.c. A regular camera is used to detect visual markers in an image frame. These markers can be implemented in several ways, but usually consist of a known black and white pattern. These markers also require line of sight. The benefit of this method is that it is relatively simple to set up and no specialized hardware is required.

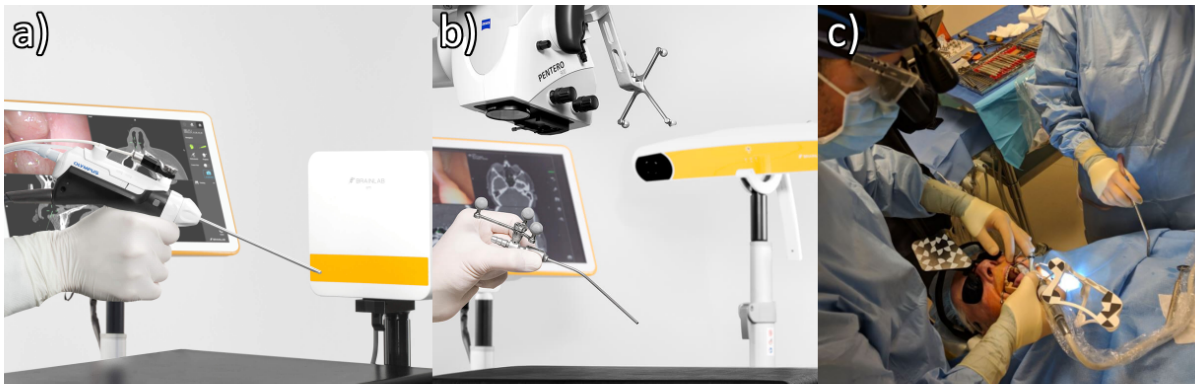

Figure 2.2: *a)* EM tracking system. On the right the field generator can be seen. The hand-held tool contains a small coil which can be detected. (Copyright Brainlab) *b)* Optical tracking system. The camera system on the right can track the IR-reflective spheres. If the spheres are places in a known configuration the system can calculate the pose and distinguish between tools. (Copyright Brainlab) *c)* Visual marker tracking system. The black and white patterns can be detected and located by a regular camera. (Copyright ClaroNav)

To be ale to use the tools the position has to be accurately known relative to the 3D patient image. For most surgical procedures an accuracy of $1 - 2mm$ is enough [36]. A patient-image registration is used to align the coordinate system of the navigation system with the physical position of the patient. Often, a landmark registration method is used. Landmarks can consist of natural landmarks (e.g. the tip of nose or ears), or artificial landmarks attached to the patient prior to creation of the patient image. A tracked pointer can be used to locate the physical landmarks on the patient. Combined with the landmarks in the patient image, the system can align the image and the patient. Another method is to use landmarks which can be detected directly by the navigation system, this way the user does not have to point to the landmarks. Cao et al. used visual markers for this task [9].

A downside of current surgical navigation methods is that during a procedure a surgeon must constantly look at a screen instead of the patient to see the working end of a tool, therefore reducing hand-eye coordination and requiring more experience [21, 59]. Furthermore, most navigation systems use a regular display, which can make it difficult to convey all spatial information of the 3D structures.

2.1.2. Craniotomy planning

In our work we will focus on creating a visualization for craniotomy planning for tumor resection. The end goal of the procedure is to remove a tumor from the brain. To do this the surgeon must first perform a craniotomy, where a certain section of the skull is removed to allow access to the brain. The brain matter between the opening and the lesion is then removed to allow direct access to the lesion. Finally, the lesion itself is removed and the skull is closed.

Because most brain tissue between the skull opening and the tumor is removed, the location of where to perform the craniotomy is important, since this has a large impact on what and how much healthy brain tissue is removed.

Besides the amount of brain tissue there are certain parts of the brain that are more important than other parts. These structures are for example the blood vessels, but also certain parts that are vital for visual or motor functions. This can consist of specific parts in the brain or fibers in the brain. Therefore, these structures should preferably also be avoided.

2.2. Augmented Reality

With Mixed Reality (MR) the idea is to mix the real world with virtual objects. MR is a group of systems that contains two subgroups: Virtual Reality (VR) and Augmented Reality (AR).

With VR a virtual environment is used, where the user is fully immersed in the virtual world. Here the user only sees objects displayed within the virtual world. Head Mounted Displays (HMDs) are commonly used to achieve the immersive aspect. A previous study used VR to plan a craniotomy for minimally invasive procedures [55], the results of this planning then had to be transferred to the patient during the procedure itself.

AR, on the other hand, is less immersive, virtual objects are augmented over the real world. This allows to see the real world and augment it with virtual objects. Since we do want the user to see the patient, we will focus on AR in this work. Augmenting virtual images on top of a view of the real world allows to show objects related to real world objects.

To show the real world there are two solutions, Video See-Through (VST) and Optical See-Through (OST). With VST a camera is attached to the device, the camera stream is then shown on the display and virtual objects are composited over the video frames. OST devices use a semi-transparent display, allowing to see the real world through the display.

With AR there are generally two categories of devices, handheld and HMDs. Handheld devices typically use a VST solution and show data in 2D on this handheld display [54]. HMDs use VST or OST, and have a display directly in front of the eyes of the user. This requires separate displays for both eyes and allows to render 3D images. A surgeon often requires both hands to perform a procedure, therefore a HMD is preferred.

With VST devices achieving good hand-eye coordination during a procedure is non-trivial, it requires an accurate and low latency visualization of the real world [61]. Furthermore, because the real world is not directly visible, VST technology might meet more resistance from users [48]. And with recent developments in OST-HMDs, there are relatively little downsides compared to using VST-HMDs.

2.2.1. Optical See-Through Head Mounted Devices

OST-HMDs are devices that use a semi-transparent display that is directly in front of the eyes of the user. If the display shows nothing, the user can therefore see the world as is. When something is shown, this virtual image is shown in front of what the user is looking at.

There are devices with only a single display, like the Google Glass[1]. These devices can be useful in certain situations where only 2D images are used, for example, to show a stream of an endoscopic camera [35].

However, we will focus on stereoscopic displays which allows to directly show 3D models. While this 3D visualization allows to give some level of depth perception, the stereoscopic depth is not good enough to work on its own. Especially if an object of unknown size is shown to the user, which is often

[1] `https://www.google.com/glass/`

the case for 3D models, it can still be difficult to estimate the depth. Therefore, other depth cues such as realistic shading are also necessary to achieve good depth perception [26].

The first studies using stereoscopic OST-HMDs were conducted in the early 1990s [4]. The devices then were large and required to be wired to an external computer, but generally the idea of the devices is the same as the current ones. These early OST-HMDs used semi-transparent mirrors to combine the real and virtual images. The real images are simply seen by looking through the mirror, the virtual images are shown by reflecting light from the display via the mirror to the eyes. A downside of this approach is that the mirror has a predetermined ratio of what light goes through and what is reflected to the display.

Current OST-HMDs do no longer use these kind of semi-transparent mirrors. Instead they use semi-transparent displays, where the actual pixels are transparent and in front of the user. This allows to have more light from the real world entering the eyes. However, the display is still not completely transparent, which can be a problem for visualization techniques relying on occlusion.

In this project we use a Microsoft HoloLens[2]. While this is one of the better performing OST-HMDs [40, 46], newer devices are currently available in the market, which we expect to perform even better.

To display something on an OST-HMD an accurate mapping from the coordinate system of the HMD to the displays is required. This depends on the position of the eyes relative to the display [49], see Figure 2.3. Most devices therefore estimate the interpupillary distance (IPD) and assume that the position in front of the display is fixed by the fit of the device. This is then used to calculate two camera matrices used when rendering the resulting image for both displays. In the HoloLens the IPD is recovered through a calibration process[3]. Eye tracking using (infrared) cameras can be used as an alternative to a calibration procedure.

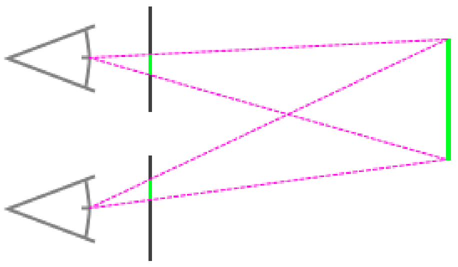
Figure 2.3: Basic idea of how an OST-HMD can display an object. The location of the eyes relative to the display must be known. From this a projection matrix for both eyes can be created, which can be used to transform a 3D location to a 2D location on the display.

For OST-HMDs the display is set so that the eyes accommodate at a fixed distance, for the HoloLens this is set to 2 meters[4]. If an object is shown at a distance which is different than this fixed distance, two problems occur. The first is the vergence-accommodation conflict, which is the conflict of different depth information from the eye accommodation and eye convergence [22]. Another problem occurs if a model is shown aligned to a real object at a depth different from the display accommodation depth, it is not possible to accommodate the eyes on both objects, and one of the two will be unfocused [5].

A downside with using a transparent display which adds light to the view of the user is that it is not possible to show dark colors. Since colors can only be added, there is no way of making a pixel of the display more opaque, any rendering is done on top of what the user can see through the display. As a consequence, it is for example not possible to trivially render realistic shadows [33].

With the HoloLens, when dark colors are rendered they are mapped to transparency in the final image, after rendering all objects. This means that, when a dark opaque model overlaps another model, that model will still be occluded, but the resulting color at that screen position will be transparent, see Figure 2.4.

Another limitation of some OST-HMDs is the small FoV of the display. On the HoloLens the display FoV is $30 \times 17°$, this corresponds to $32 \times 18cm$ at a distance of $60cm$. Although this is small, it has been shown that this is not necessarily an issue in medical settings since the user tends to focus on a small part of their vision [28].

[2]`https://docs.microsoft.com/en-us/hololens/hololens1-hardware`
[3]`https://docs.microsoft.com/en-us/hololens/hololens-calibration#calibrating-your-hololens-1st-gen`
[4]`https://docs.microsoft.com/en-us/windows/mixed-reality/design/comfort#guidance-for-holographic-devices`

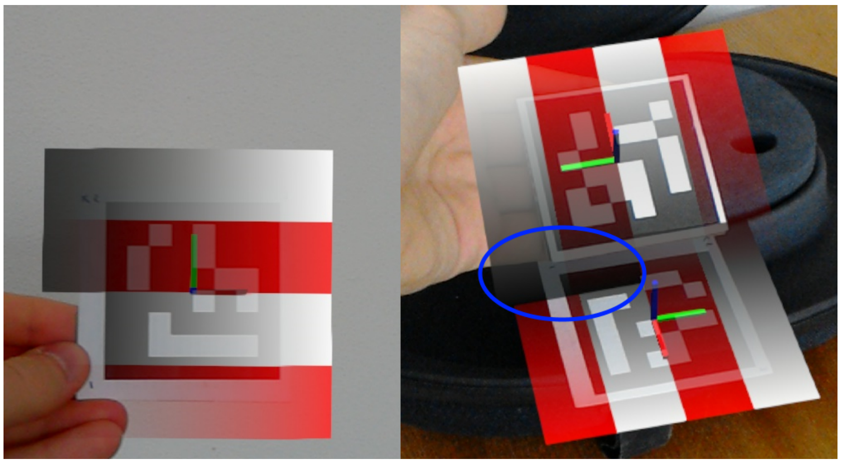

Figure 2.4: Demonstration of how dark colors are mapped to transparent colors in the HoloLens. *Left:* Top two lines show a gradient from black to white and from black to red, with only opaque colors. The bottom two lines show a gradient from transparent to opaque in both white and red. When wearing the HoloLens, the top two lines look exactly like the bottom two, but this color mapping is not applied when recording images. *Right:* Demonstrates that the color mapping is applied at the last rendering step. The left part where the black opaque part overlaps the partly transparent models (in the blue circle), this transparent part is occluded, after the final color transformation, the result in the HoloLens is a fully transparent color.

2.2.2. Object tracking

We need to track the head of the patient to be able to align a model with the head. Even if the patient is in a stationary position this is necessary, since the HMD does move. There are two categories of tracking, outside-in and inside-out.

With outside-in techniques, external tracking systems are used to track the position of the patient and the HMD, then the relative position of the patient to the HMD is calculated to display the model, see Figure 2.5. The benefit is that external tracking systems with a known high accuracy can be used. The tracking data can be sent to the HMD where the relative position of the head to the HMD can be calculated. A visualization error of $1.5mm$ was reported using this method [38]. Another approach is to use markers which can be detected both by the HoloLens and an external tracking system [20].

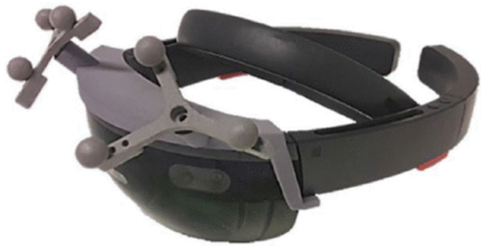

Figure 2.5: Optical tracking spheres can be attached to the HoloLens. The relative location of tracked tools or patients is then known and can be used to display models in the HoloLens. (Meulstee et al.)

Inside-out tracking on the other hand only uses the HMD as hardware, which thus performs the tracking and displaying. The benefit here is that the setup time is reduced, making it potentially useful in more time-critical situations.

For inside-out tracking there are different methods. One technique is using spatial maps to track the position of the HMD in the space, this does not track the patient however, so it can only be used if the patient is stationary. Another technique is using fiducial markers attached to the patient, since the markers are attached to the patient it is possible to move the patient and still keep the model in the correct position. With fiducial markers and a stationary patient it is also possible to place the markers in the background instead.

The HoloLens provides a spatial map of the surrounding which it continuously maintains, see Figure 2.6. Several studies have been performed to find the accuracy of this spatial map. It was found that it is not very stable, and disruptions in the spatial map can move the models by around $5mm$ [58]. Another study found that the HMD localization has a positional error of around $10mm$ [30].

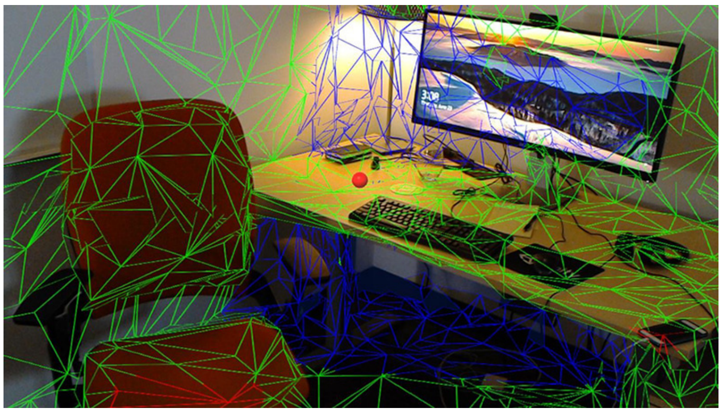
Figure 2.6: Spatial mapping on the HoloLens. (Copyright Unity)

Using fiducial markers instead we can use a camera to track predefined patterns on markers. These can then be used to track objects or the position of the HMD. Several studies in the past looked at creating these markers, Garrido-Jurado et al. lists some of these and reports that ArUco [18, 50] is a highly reliable method with a pixel error of about 0.16 in the corners of the markers [17]. Other proprietary solutions such as Vuforia[5] are also possible, a surface point localization error of $1.92mm$ can be achieved using these markers [15].

Some previous studies also looked at using the HoloLens and markers as a navigation system. They gave accuracy results ranging from $1.5mm$ to $5mm$ using various techniques [2, 9]. However, these studies gave a single accuracy number as their result, giving no insights in how this accuracy is influenced by variables or distributed over the different directions.

2.3. Visualization for craniotomy planning

Previous work has been done in visualizations for craniotomy planning without AR. Many of these use MRI slices in combination with rendering some surface models of structures [57]. Two other studies included a view where the user can look down a proposed craniotomy position to see the structures between the created skull opening and the tumor [12, 42], see Figure 2.7.

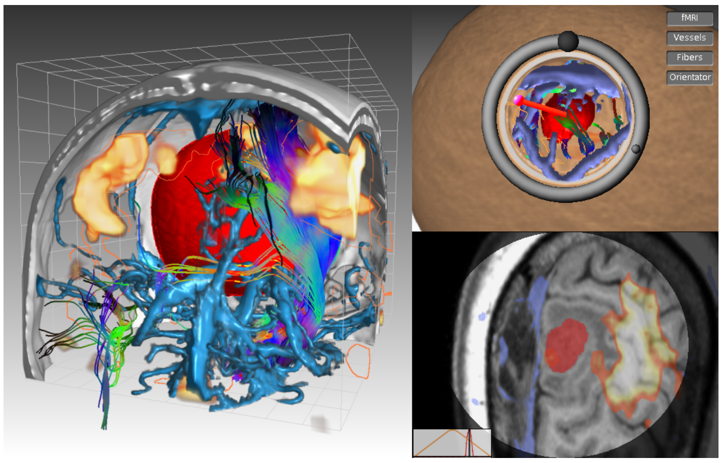

Figure 2.7: The left view shows several brain structures. The view on the top right shows a proposed craniotomy and a view to look down the hole in the skull to the tumor. On the bottom right it shows a slice view with some structures highlighted. (Diepenbrock)

Another study used a technique where the tumor emits rays from its surface. These rays are then attenuated by structures that should be avoided. Thus it is possible to see where there is a higher

[5]https://vuforia.com

number of rays that reach the skull, this should then be the position with the least structures between the tumor and that part of the skull [24], see Figure 2.8.

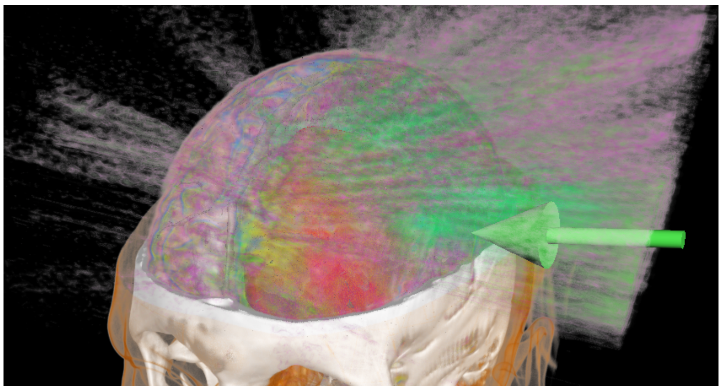

Figure 2.8: In this visualization rays are emitted from the tumor and blocked by important structures. (Kainz)

The opposite is also possible, tracing rays from the brain surface to the tumor. Here a single point in the brain is chosen as the destination, and the amount of important structures between every point on the brain and the destination is calculated. This is then mapped to a color, which results in a heat map on the brain surface [64].

Cutolo et al. used a VST-HMD visualization to perform a craniotomy. They mention that a downside of VST-HMDs is that the camera feed is not perfectly aligned with the eyes. In this paper the craniotomy has been planned beforehand, the visualization is then used to show where to open the skin and skull on the patient.

3 Marker Tracking

In this chapter we will explain how we used ArUco to track markers in the HoloLens and how this can be used to align models created from a 3D patient image to a patient.

The markers that ArUco uses are black and white planar markers. Camera frames are captured in which we find the position of the marker corners. From these corners we can estimate the position and orientation of the marker relative to the camera. Then the transformation from the camera to the display is used to display models relative to the spatial position of the markers.

3.1. Camera system

The HoloLens has several cameras build into it: four grayscale cameras used to capture the environment, a depth camera, and an RGB camera.

In this project we use the single RGB camera for marker detection. Using the multiple grayscale cameras instead might give better results, because the multiple camera streams can be combined to get better accuracy. Using multiple cameras allows to utilize binocular disparity to better estimate the depth of the markers. This can also be achieved by using the depth sensor, although it is not certain the resolution and capture frequency of this camera are high enough to use in practice. But due to time constraints using these different camera streams was not investigated further.

The RGB camera of the HoloLens has several modes that can be used. The first camera mode uses video stabilization, which should make the transition from frame to frame look smoother. This has no use when processing the frames one by one using software, therefore this mode was not used. The second mode uses a large horizontal field of view (FoV) of 67 degrees and a resolution of 1344×756 pixels. The two remaining modes have a horizontal FoV of 48 degrees and a resolution of 896×504 and 1408×792 pixels. Other camera specifications are that it captures 30 frames per second (FPS), has fixed focus, auto white balance, and auto expose time.

The horizontal FoV of the display of the HoloLens is only 30 degrees. Using a camera mode with a much larger FoV makes it possible to detect markers that are outside the range of the display. This would be useful if the markers can be far away from the models that are displayed. But in our target application we want to keep the markers close to the models, since a larger distance will lead to larger error in the location of the models. Therefore, we use the camera modes with a FoV of 48 degrees.

In initial tests we found that the marker detection in a 1408×792 frame takes around $150ms$. Since the detection rate would then be much lower than the FPS of the camera, we wanted to see how the application would perform with lower resolutions. Therefore we added two camera modes where the two original resolutions are downscaled by a factor of two. The original frame is downscaled using bilinear interpolation, which takes about $5ms$. Using nearest neighbour instead would be faster, but this resulted in visibly less accurate tracking.

Every camera has specific intrinsic parameters. These describe the focal length, the camera sensor size, the principal point, and distortion coefficients. The focal length and the sensor size are used to determine the physical scale of the frames and the principal point describes the optical center of the image. The distortion coefficients that the HoloLens provides are for radial and tangential distortion. Radial distortion describes the shape of the lens, and tangential distortion describes the misalignment of the lens. The HoloLens API provides these parameters, which we use to remove distortions from the camera frames and to calculate the physical size the frames represent.

3.2. Marker detection

The HoloLens requires that the main rendering thread runs close to 60 frames per second (FPS). Since the marker detection is slower we run it on a separate thread. Frames are captured at a higher rate than they can be processed, therefore, every new frame is stored in a buffer, overwriting the previous frame, until the marker detection thread can fetch the most recent frame from the buffer. Only the intensity values are saved to the buffer, since color values are not used in the marker detection.

3.2.1. ArUco

To extract the marker poses from the camera frames we use the ArUco library written in C++ [18, 50]. This library used to be open-source but moved to a more restricted licence after version $3.0.13$. Yet, because of a lack of good documentation, being able to read the code can give a much better understanding of how the library works and can be used. Therefore, the last open-source version was used in this work. ArUco has a dependency on the OpenCV library, we used the newest supported version $3.4.9$.

ArUco its detection method is based on contour detection. The idea is to find contours of dark rectangles on a light background. From these contours it then reconstructs the inner binary grid and finds the correct orientation. Then, finally, from the four corners of the marker the pose can be calculated. See Figure 3.1 for an overview of these steps.

The first step of the marker detection process is a thresholding method to binarize the image. The markers black and white, a local adaptive method is used to distinguish between these colors. In this method a window size w is specified and for every pixel of the frame a mean of the neighbourhood of size w is taken. The pixel is then set to 1 if the intensity is lower than the mean of the window, otherwise it is set to 0. This adaptive method usually yields better results than global thresholding, because the amount of light can vary greatly within a single frame. A downside is that this method is relatively slow, especially for larger frame resolutions or larger window sizes.

Therefore, ArUco first scales down the image using bilinear interpolation. We have set ArUco such that it downscales the frame to a size of 704×396, this is larger than ArUco requires for detecting a marker. But we found that with smaller resolution frames ArUco was no longer able to detect the markers consistently, this may be due to poor lighting conditions in our setup. With the 1408×792 camera resolution this downscaling leads to a speedup of the whole detection by around a factor of 5.

The next step is finding contours from the binary image, a border following algorithm is used [56]. The found contours are then filtered based on the size and shape. Contours with a perimeter of less than 80 pixels or that are not a convex quadrilateral are discarded.

Then, the inner regions of the contours are reconstructed to find if the markers are valid. The inner pixels of the quadrilateral contour are mapped to a square. Then a grid is placed over this square and the mean of every cell is used to classify it as black or white. The binary grid has four possible orientations, each of these is tested against the dictionary of possible markers. If a valid configuration has been found, then the identifier and orientation of that marker is known, otherwise the contour is discarded.

To get more accurate results the corners of the contour need to be refined. Therefore instead of doing the corner location estimation on a per pixel basis, lines are reconstructed using linear regression on the edge pixels, and the intersection points of these lines are used as the corners. Note that this step is performed on the full resolution image, and not the downscaled version. The result is a subpixel accuracy of the four corners of the marker.

Finally, the sets of corners are used to find the pose of the marker in camera space. This requires to find the pose of a 3D marker from the 2D points of the projections of these points on a plane, this is called the Perspective-n-Point (PnP) problem. Normally, solving this problem should give one solution. However, when the projection of the 3D to 2D points is close to an affine transformation, which is the case if the marker is small or at a low incident angle, two possible solutions can seem correct. This issue is known as the pose ambiguity problem. The Infinitesimal Plane-Based Pose Estimation (IPPE) algorithm returns both possible solutions with a reprojection error of the solutions [10]. Normally the pose with the lowest error will be chosen, but with prior knowledge about the pose it can sometimes be

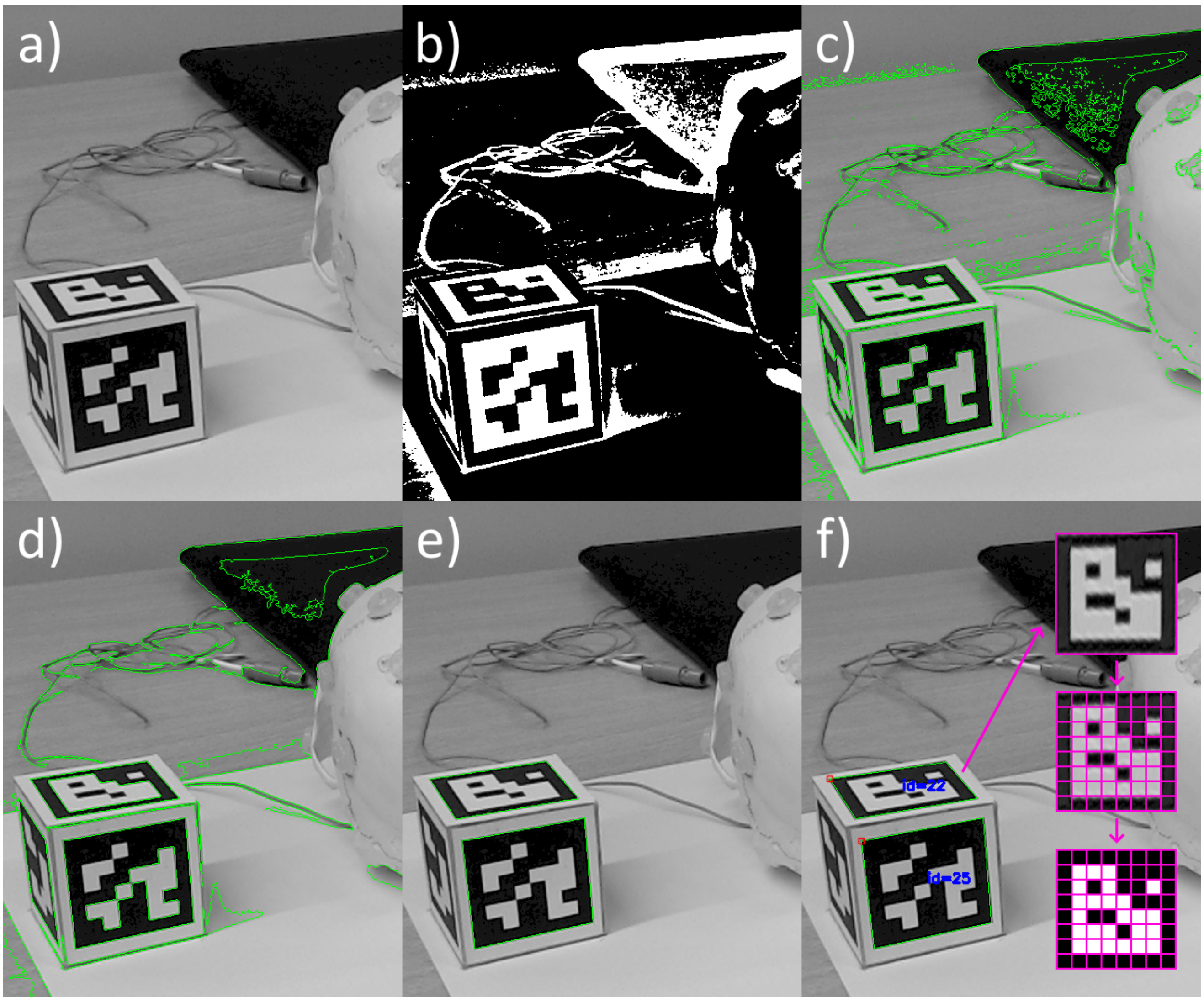

Figure 3.1: The intermediate steps of the ArUco marker detection process. *a)* Original grayscale image. *b)* Adaptive threshold result. *c)* Contours detected in threshold image. *d)* Contours filtered on perimeter length. *e)* Contours filtered on shape. *f)* A grid is placed over the warped image of detected contours, then a binary image is created in the grid which can be used to classify the marker and find the orientation.

better to instead use the pose with the higher reprojection error.

3.2.2. Frame sequences

We can use data from previous frames to improve the marker pose detection. This can be especially useful to discard or correct poses which are incorrectly detected.

ArUco provides a method which tracks the pose of a marker over several frames to improve the detection. If it already knows the pose of a marker from a previous frame, it will solve the PnP problem using an iterative approach based on Levenberg-Marquardt optimization. In this iterative approach the previously detected pose is provided as the starting point for the algorithm. The result of this iterative approach is a single new pose.

A downside of this method is that if the first frame is detected incorrectly, the following frames will continue to be incorrect. To prevent this we have set ArUco such that for an initial pose estimation it is required that the reprojection error of the first result of the IPPE algorithm is at least 10 times smaller than the reprojection error of the second result.

The iterative PnP algorithm can still produce wrong results if the pose is very different from the previously detected pose. This can occur if a marker is occluded for a while. Therefore we reset the previous marker pose if the detected pose is different by at least $5cm$ or $30°$.

Another possible optimization with a sequence of frames is to use a temporal filtering method. With this filtering it should be possible to reduce the amount of noise in the pose of the detected markers. In this project we use a Kalman filter [25]. The filter keeps track of previously detected poses, and is then able to make an estimate of the next pose. When the next pose is retrieved from the frame it can be combined with the estimated pose to get the resulting filtered pose.

A Kalman filter keeps track of its current state and several measurements. A measurement consists of several individually linear variables, these are defined as the translation along the three axes and a quaternion describing the rotation. The state consists of a measurement and the first derivative of a measurement, higher order derivatives can also be included, but were not implemented. Another setting that the Kalman filter requires is the amount of time between the measurements, this is set to the average time that the frame processing takes. An estimate of the noise and expected variability for each of the measurement variables has to also be provided. These settings were estimated based on how the resulting tracking looks in the HoloLens, but for future work it would be better to input actual measurements of these variables.

For every marker that is used, a separate Kalman filter state is created. Then for every frame a new state is estimated, based on the previous state and measurements. The newly detected marker poses are then also provided to the Kalman filters as new measurements. The results from the Kalman filters are then finally used as the pose for the markers.

The Kalman filter also allows to make an estimate of the pose of a marker even though it is not detected. With small markers, dark environments, or fast movements it often happens that a marker is not detected for a single frame. Hiding a model attached to a marker is not really an option since this would result in an annoying flickering effect. The Kalman filter allows to show the model in a predicted position based on previous poses. It is also possible to increase the number of frames that can be filled in by the Kalman filter, but with a relatively low FPS of the marker detection this does not give good results.

In our application the markers are usually at a static position in world space. However, since the user will move around the markers, the pose of the markers is not static in camera space. The HoloLens provides an API to get the position of the HoloLens in the world based on SLAM and the IMU. We could use this API to get an estimate of the markers in the world. If we now assume the velocity of the markers to be zero, this may lead to less noise and more accurate marker detection. However, the detection is then based on the accuracy of the detection of the pose of the HoloLens in the world. Due to time constraints we were not able to look further into this.

3.3. Combinations of markers

From preliminary tests we found that the detection of single markers is quite noisy. Additionally, using only a single marker may lead to problems with occlusion. Therefore, we combine the pose of several markers to reduce noise and deal with occlusion issues.

In our solution we choose one marker to be the main marker in such a combination. All other markers then have a pose defined as the relative transformation to the main marker.

Then, the poses of all markers can be transformed to the space of the main marker. In an ideal situation, all poses would match, but due to measurement errors they will not perfectly align. Thus, we take the average of all transformed poses to retrieve the pose of the main marker. Averaging the transformations is performed in two steps, for the translation we can take the mean position per axis, for the rotations it is required that they are transformed to quaternions before they can be averaged [34]. For future work, weights might be used to improve this averaging, for example, using the reprojection errors from the PnP algorithm as weights.

Using the relative position of the markers can be used to reduce the number of pose ambiguity problem errors. The IPPE algorithm used to solve the PnP problem results in two possible poses for a single marker. But if corners from multiple non-coplanar markers are combined the PnP problem only has a single solution, therefore solving the pose ambiguity errors. Due to time constraints, this was left as future work.

Often the transformations between the markers in a combination are not known beforehand. A calibration process is needed to find these relations. The result of the calibration process should be a set of transformations from every marker to the main marker. The calibration process consists of capturing a 1000 frames of the markers, it should be made sure that all markers are represented in this set and that different orientations are used.

For every individual transformation from any marker to any other marker, we can take the average of all transformations that were found in the frames captured. For every such average transformation we can also make an estimate of how precise it is by taking the root-mean-squared (RMS) error of the translation and rotation. Now we create a graph with all markers as nodes and the transformations between the markers as edges. The edges their lengths depict the RMS error of the transformations. Then, for every marker we find the shortest path to the main marker. The transformations between the markers on the shortest path are concatenated to result in a transformation from the marker to the main marker. This calibration method is similar to the one used in a study by Shaya et al. [53].

During the calibration process some frames can contain outliers of the poses that may lead to a large error during averaging. A simple solution to remove outliers is to manually measure an estimate solution beforehand. When calibrating, every relation that is too far from the manual measurement is discarded. Another possible solution would be to remove outliers based on a moving average strategy during calibration, therefore, not requiring manual measurements beforehand, but it was not implemented.

3.4. Attaching models to markers

To align models to a patient we need the transformation between the markers in camera space and the physical position of a patient. The transformation between world space and the coordinate system of the camera is retrieved from the HoloLens its API.

Markers can be attached to the patient or placed around the patient. In the second case it is not possible to move the patient, because it would change the transformation. If markers are rigidly attached to the patient moving the patient would also move the markers, therefore maintaining the alignment. In this work we will mostly focus on markers placed in the background. Since we did not test with actual patients, this was easier to test with, but the system also works if markers are rigidly attached to the patient.

The registration of the model to the physical patient position is possible in two ways. A manual method, where the user moves the model seen in the HoloLens in order to overlap it with the patient. This method is however quite time intensive and it is difficult to achieve a good precision. While it is relatively easy to see if a model is correctly aligned in the x and y axes, correctly aligning the depth and rotation is not trivial, constantly requiring the user to change viewing angles.

A more automated way is therefore preferred, this can be achieved with a landmark registration method. With this method a set of landmarks is defined on the the patient, for example the tip of the nose and ears. Then, using a tracked pointer, these points can be used in a 3D point cloud registration method. Unfortunately, due to time constraints we were not able to test the accuracy of this registration method, therefore, the manual registration method was used instead.

Because the marker detection usually takes longer than rendering a frame, we do not accurately know the pose of the markers on every rendering frame. Therefore, we need a method to update the marker poses between the marker detection frames. We could use the Kalman filter and make a prediction of the measurements on every rendering frame, but this would still fail with a sudden movement of the HoloLens or a marker.

Fortunately, the HoloLens has some methods to overcome this problem. Using the continuously updated spatial map and a IMU, the HoloLens can estimate its position in a room. We can set the position of the marker relative to a stationary frame of reference, and update the position of the marker in this reference frame when a new camera frame has been processed. The HoloLens updates the camera matrix based on the spatial map and the stationary frame of reference, resulting in updates of the pose of the marker in between the camera frames.

4 Visualization Methods

In this chapter we present our visualization method to assist in craniotomy planning. The visualization consists of two parts, the first part is the rendering of models of brain structures. The second part is the aggregation of these models on the skull, we call this the planning map.

4.1. Models

For our visualization we need 3D models of patient anatomy. In clinical practice this data would come from scans of a patient. For our study we use two datasets provided by the 2010 IEEE Visualization Contest[1]. This dataset consists of two cases, *case 1* contains several types of MRI scans and *case 2* has these same MRI scans and an additional CT scan. All scans contain a voxel to world transformation, allowing to align the individual images. In Table 4.1 we show the details of the used image modalities in the dataset, Figure 4.1 shows a slice of the different image types.

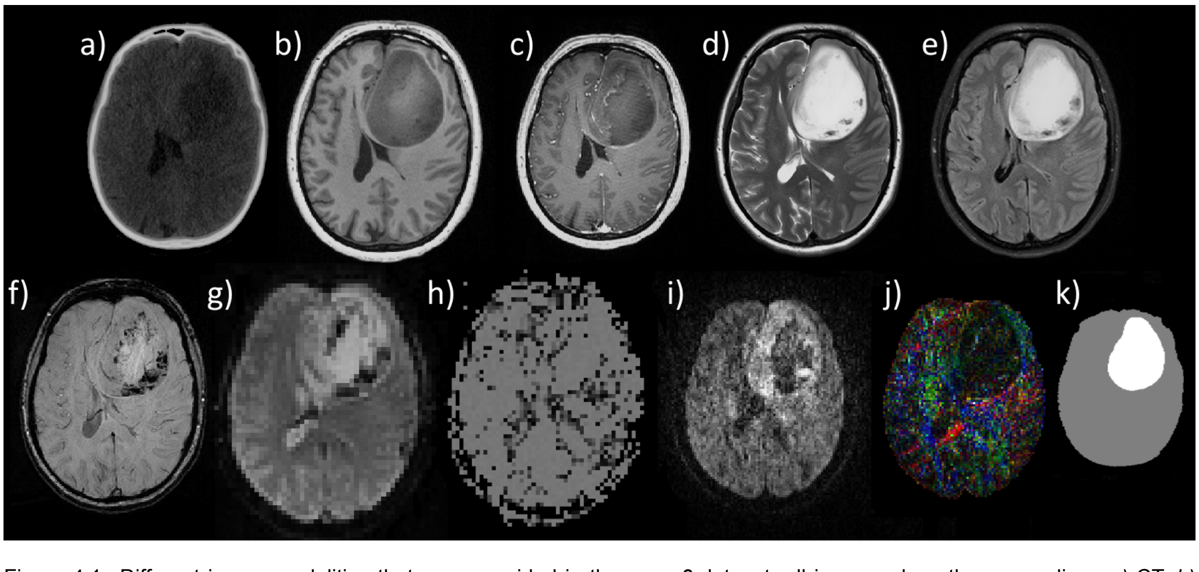

Figure 4.1: Different image modalities that were provided in the *case 2* dataset, all images show the same slice. *a)* CT. *b)* T1-weighted MRI. *c)* T1-weighted MRI with contrast agent. *d)* T2-weighted MRI. *e)* FLAIR. *f)* SWI. *g)* fMRI one time point during finger tapping. *h)* fMRI statistical map. *i)* DTI from a specific gradient. *j)* DTI vector data. *k)* Tumor and brain segmentation.

To visualize these datasets we need to create surface models and other 3D images. Direct volume visualization would require too much computation power, which is not available in the HoloLens. We use MeVisLab[2], using several different modules:

[1] `http://sciviscontest.ieeevis.org/2010/data.html`
[2] `https://www.mevislab.de/`

Fig	Name	Spatial resolution	Usage
a	CT	$0.42mm^2 \times 0.40mm$	High values for bone and low for air. Allows to segment the skull and skin.
b	T1-weighted MRI	$0.49mm^2 \times 1.00mm$	High values for fat, blood, contrast agent, melanin, and proteins. Mostly useful for determining normal anatomical features.
c	T1-weighted MRI with contrast agent	$0.49mm^2 \times 1.00mm$	Equal to T1-weighted MRI, but with contrast agent. Allows to clearly see vessels.
d	T2-weighted MRI	$0.36mm^2 \times 6.00mm$	High values for most fluids, including cerebrospinal fluid. Also highlights tumor because most damaged cells contain a relatively large amount of fluid.
e	Fluid Attenuated Inversion Recovery (FLAIR)	$0.45mm^2 \times 6.00mm$	Similar to T2-weighted imaging, but with a reduced value for cerebrospinal fluid. Can be used as a normal image for the T2-weighted MRI to highlight cerebrospinal fluid even more.
f	Susceptibility Weighted Imaging (SWI)	$0.90mm^2 \times 2.50mm$	Shows some possible artifacts in other imaging techniques, not really useful for our visualization.
g	Functional MRI (fMRI)	$3.00mm^2 \times 3.30mm$	Technique which can assign a probability value to certain neurological activity. During the scan the patient performs a certain activity, which was finger tapping in these datasets. Scan of a 100 time points, containing 5 sessions of 10 time points of activity followed by 10 at rest.
h	fMRI statistical map	$3.00mm^2 \times 3.30mm$	T-values which represent the statistical probability of the voxel being related to the finger tapping activity. 5%, 1% and 0,1% significance thresholds were provided for the image using False Discovery Correction method.
i	Diffusion Tensor Imaging (DTI)	$1.80mm^2 \times 1.98mm$	Measures the diffusion of water in the brain, which is restricted in different directions by regular brain structures. Neurological fibers have a direction that also follow these brain structures. Can be used to create a set of images which each describe the amount of diffusion for a specific gradient direction, these can be combined to get a single diffusion tensor for every voxel. Two separate sets of of 30 gradient directions were provided. The diffusion tensors can be used to find some major fiber-bundles in the brain, making it possible to identify connectivity of some brain areas.
k	Tumor and brain segmentation	$0.49mm^2 \times 1.00mm$	Annotated T1-weighted MRI image containing a tumor and brain mask. Used to create the tumor surface model. Also useful in combination with other images where, for example, only the section inside the brain is relevant.

Table 4.1: Description of the different image modalities provided in the 2010 IEEE Visualization Contest datasets.

- *Threshold*: For every voxel test if it is greater or smaller than a specific value to create a binary image.
- *Reformat*: This creates an image with the exact resolution and spatial properties as a reference image. At every voxel a trilinear interpolation is performed with the data from the original image.
- *Arithmetic*: Perform per voxel arithmetic. Requires both images to have the same resolution.
- *Mask*: For every voxel in an image, if a reference binary mask is 1 then the original value is used, otherwise the result is set to 0. Requires equal resolutions of the input image and the mask.
- *Convolution*: Perform a 3D convolution over the image, any kernel can be specified.
- *FastMorphology*: Opening or closing morphology operation, can be performed on binary or gray images. An opening operation performs an erosion followed by a dilation, can close gaps between areas. A closing operation performs a dilation followed by an erosion, can be used to disconnect areas.

- *RegionGrowing*: Performs region growing operation on binary data from a seed point, keeping only areas that are connected to the seed point.
- *WEMIsoSurface*: Creates a surface mesh using an isosurface. Uses an algorithm similar to the Marching Cubes algorithm [31].
- *WEMReducePolygons*: Reduce the number of polygons by collapsing edges [16]. With large resolution images we get surface meshes with a large number of polygons, which would unnecessarily slow down the rendering. For example the skull mesh created using an isosurface has almost 4 million polygons, we can reduce this to around 7500 without losing too much precision.

We create the following 3D models and 3D images:

- *Skull:* See Figure 4.2.a. We segment this using a threshold on the CT data, after this we can use an isosurface to get the surface mesh for the skull. Although it is possible to segment the skull from MRI data, this is not straightforward [13], therefore we use the CT image.
- *Tumor:* See Figure 4.2.d. Can be directly created from the T1-weighted tumor mask using an isosurface.
- *Vessels:* See Figure 4.2.b. The vessel structures can be segmented from the difference in the aligned T1-weighted MRI images with and without contrast. Here we subtract the image without contrast from the image with contrast, It is possible to segment it with the T1 with contrast only, using a simple threshold, but it results in more disconnected vessels and other artifacts. We are only interested in the vessels inside the brain, the brain mask is used to exclude other areas. In the first case the tumor is also highlighted in the difference image, we use the tumor mask to exclude it. Now for the second case we use an isosurface to create the 3D model of the difference image.

  However, for the first case the contrast image contains some areas with overall higher values. Therefore, we use edge detection to find the vessels. We convolve the image with a Sobel kernel in the x and y direction. Then we take a threshold of the sum of the absolute values of the convolution. After this the vessels are filled with a closing convolution. Finally, we use an isosurface to get the final surface model of the vessel structure.
- *fMRI areas:* See Figure 4.2.e. From the fMRI statistical map we can simply take a threshold for the significance value we choose. We chose a significance threshold of 0.01%, as with a higher value a larger portion of the brain was selected occluding many other structures. The result is a 3D image with binary values defining if the area is significant or not. We save both the surface mesh created using an isosurface and the 3D binary image which is used in the rendering of the planning map.
- *DTI fiber data:* See Figure 4.2.c. The DTI data is a set of images giving an amount of diffusion for a specific gradient direction. These images are combined to create a 3D tensor field [6], with a tensor per voxel describing the diffusion orientation. From this, we can compute the principal direction to extract a 3D vector field (see Figure 4.1.j). The length is defined by the fractional anisotropy, a scalar describing how directional the diffusion process is. The fractional anisotropy is also used as a separate 3D image (texture) to attribute colors to the resulting fibers. We use the MeVisLab *DiffusionTensorAnalysis* module to perform this analysis.

  We define a seed region, then from each voxel in this region we start a fiber tracking algorithm. For this we use the *TensorTractography* module, this uses an algorithm based on following the vector field with constraints on the fractional anisotropy [52]. The result is a set of fibers giving an idea of where the neural connections go from the seed region.

  As a final step we reduce the number of line segments to increase the rendering speed. We do this with the *ReduceFibers* module which removes a node if the $|cos(\theta)| >= 0.99$, with θ the angle between the line segments attached to the node. This method reduces the number of line segments by around a factor of three.

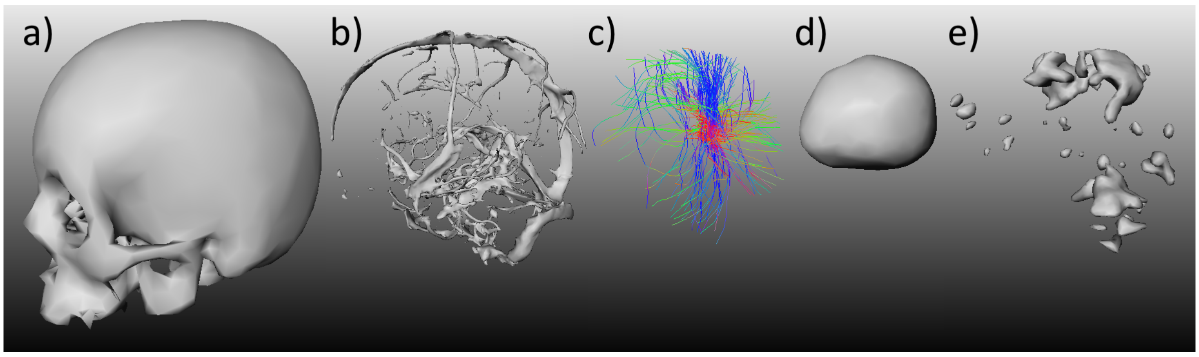

Figure 4.2: Surface models and DTI fibers used in the rendering for the *case 2* dataset. *a)* Skull surface model. *b)* Vessel structure surface model. *c)* DTI fibers, the colors denote the direction of the fibers with (x, y, z) mapped to (r, g, b). *d)* Tumor surface model. *e)* fMRI areas surface model.

See Figure 4.3 for the models created for the *case 1* dataset. For the evaluation and images in this thesis we used a phantom skull. From a CT scan of this phantom a surface model was created using an isosurface, see Figure 4.3.a. Table 4.2 lists the number of vertices and polygons of all the models that were created.

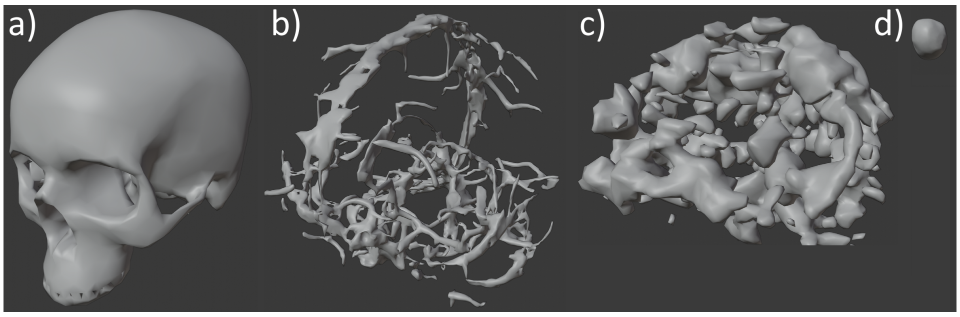

Figure 4.3: Surface models created of the phantom skull and from the *case 1* patient dataset. *a)* Surface model of the skull phantom. *b – d)* Surface models created from the *case 1* dataset, shown are the models of the vessel structure, fMRI areas and tumor.

	Case 1			**Case 2**		
Model	**Vertices**	**Polygons**	**Reduced polygons**	**Vertices**	**Polygons**	**Reduced polygons**
Phantom skull	2650	5472	99.8%	*Equal to case 1*		
Skull	*n.a.*	*n.a.*	*n.a.*	4615	10046	99.8%
Tumor	670	1336	85%	348	692	99.6%
Vessels	11010	22476	97%	17464	34996	95%
fMRI areas	8045	15874	90%	2274	4404	90%
DTI fibers	1969	3560	60% – 65%	4633	8736	60% – 65%

Table 4.2: Number of vertices and polygons of the surface models that were created. The reduced polygons columns lists the percentage of the original isosurface polygons that was removed to get to the resulting number of polygons. For the DTI fibers, the vertices are the nodes between the line segments, and the polygons are the number of polygons that are eventually created on rendering, when planes are used. The polygon reduction for the DTI fibers is an estimate since this is based on the angle between the segments.

To combine the data from the datasets mentioned above with the phantom, the datasets are transformed to fit inside the phantom skull, this transformation was performed manually. This might create some anatomically unrealistic structures, since every skull shape is different and, for example, the vessel structure inside the skull will also be different. Nevertheless, when we consulted a neurosurgeon it was not perceived as a problem to evaluate the visualization.

4.2. Shading

The Blinn-Phong reflection model [8] is used for the shading of the models. This model uses ambient, diffuse, and specular reflections to compute the color of a pixel. The resulting color of a fragment is calculated as:

$$C_{fragment} = k_a * C_{material} + k_d * C_{material} * N \cdot L + k_s * C_{light} * (N \cdot H)^{\alpha}.$$

Where k_a, k_d and k_s are factor for the ambient, diffuse and specular component, respectively. α is a shininess factor of the material, and C describes a color.

4.2.1. Transparency

In the visualization, we want to visualize structures that are behind other structures. To this end, most objects are rendered partially transparent. To render the correct resulting color when multiple partially transparent objects overlap we use alpha compositing.

With alpha compositing a semi-transparent color is combined with a background color. When compositing a over b, where a is a new value and b is the existing background and o is the result:

$$\alpha_o = 1 - (1 - \alpha_a)(1 - \alpha_b) \quad ,$$

$$C_o = \frac{C_a \alpha_a + C_b \alpha_b (1 - \alpha_a)}{\alpha_o} \quad .$$

Is is not possible to directly use this function, since in the rendering API a_o is not available when calculating C_o. Normally, when an opaque background is used, a simplified version of this formula is straightforward to implement. However, AR uses a non-opaque background, therefore this simplification cannot be used.

Premultiplied alpha compositing is used instead [45]. On rendering the color values are pre-multiplied with the alpha component, thus rendering $(r, g, b, 1) * \alpha$ instead of (r, g, b, α). Then, we transform the color compositing formula to

$$C_o \alpha_o = C_a \alpha_a + C_b \alpha_b (1 - \alpha_a) \quad .$$

Here we see that with the premultiplied color values this can be simplified to

$$C_o = C_a + C_b (1 - \alpha_a) \quad .$$

The function to calculate the resulting alpha values is still the same.

With alpha compositing the result is not invariant to the order of the fragments that are processed, having one fragment in front of another is not the same as the inverse. Ideally this is solved by rendering all fragments from the back to the front, but this is a very expensive task. Therefore we use some workarounds, the first is enabling depth testing for all models, this enables the use of a depth buffer where every fragment writes its depth. Then for every other fragment the depth is required to be less than the previously written fragments.

The depth buffer now prevents wrong ordering of fragments, but it also causes underlying structures to no longer be rendered. We again want to render the fragments from the back to the front, but instead of sorting all fragments, we sort the models. Since we are mostly interested in the models in front of the tumor we can use this information to sort them. Furthermore, because this does not change when moving around the object, it can be defined during initialization:

1. *Tumor:* we are interested in what is in front of the tumor, so this is the object that is the most in the back.
2. *DTI fibers:* are usually selected close to the tumor.
3. *fMRI areas:* most interesting areas are also close to the tumor.
4. *Vessels:* most important vessels are near the skull surface.
5. *Skull:* outer layer, everything else is inside the skull.

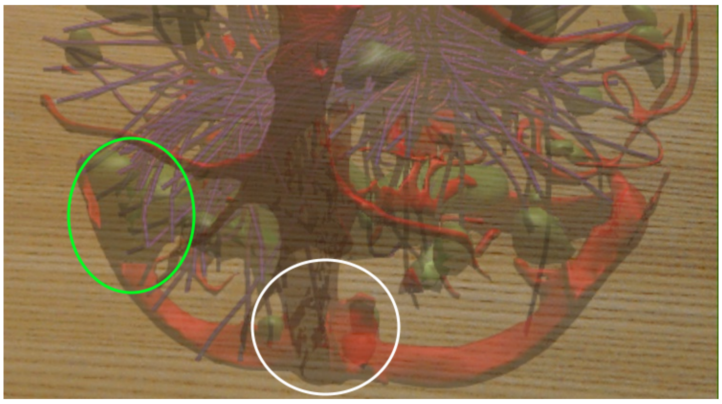
Figure 4.4: Some of the issues with our transparency implementation, for demonstrative purposes the models are more transparent than they are normally. In the green circle we see that the vessels are rendered later than the fMRI areas, because the parts of the vessel behind the fMRI area do not pass the depth test, they are occluded. In the white circle we see self overlap of the vessels, because the order is random we see some polygons that are rendered before the closest layer of the vessel while others are rendered later and fail the depth test.

This method is not perfect, sometimes the order of the processed fragments will be wrong, but this will then only result in a few fragments being occluded, see Figure 4.4.

Another problem with transparency occurs if fragments from the same model overlap. The order of what fragment is processed first is random, causing fragments to randomly pass or fail the depth test. The skull model has multiple layers of surfaces which are directed towards the camera, leading to many visible artifacts, see Figure 4.5. To handle this issue we discard fragments when:

$$n_f \cdot (p_c - p_f) > 1 \quad ,$$

where n_f is the normal of the fragment, and p_f and p_c are the position of the fragment and a point in the center of the skull respectively. This still creates some artifacts near the eye sockets, but this is a cheap way to prevent sorting fragments. This problem also occurs with other models, but this is not significantly noticeable, see Figure 4.4.

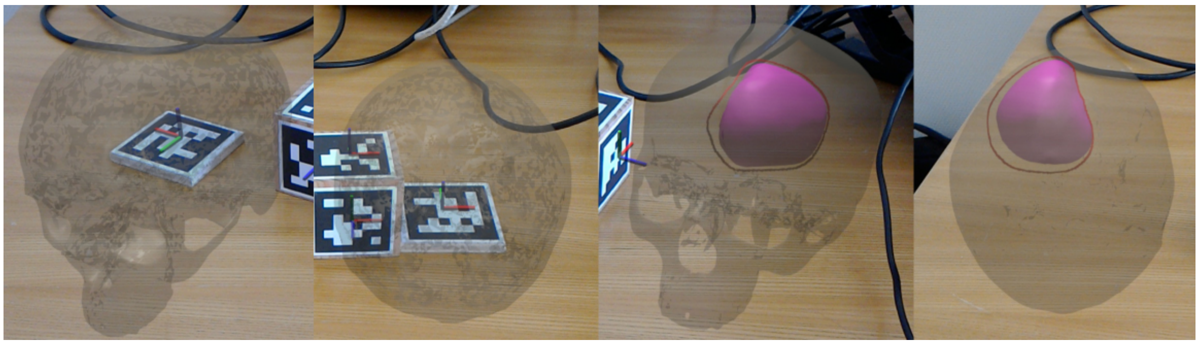
Figure 4.5: Artifacts that are created because of the self-overlap of the skull model. In the left two images you can see that this problem occurs almost everywhere in the skull. The right two images a test has been added to see if the normal of a fragment is directed to the center of the skull, if this is the case, the fragment is discarded. The result is that only near the front of the skull the artifact occur, which is not a region of interest for a craniotomy anyway.

4.3. DTI fibers

The DTI fiber model consists of a set of undirected lines and each line consists of a number of segments. While it is possible to directly render these lines as a line primitive this is not desirable, because the rendered width would not be dependent of the depth, removing an important depth cue. Moreover, since line primitives are only a single pixel wide the possibilities for shading are also limited, see Figure 4.6.a.

A commonly used rendering method is to map the direction of the lines to a color, see Figure 4.6.b. The direction in the x, y and z axes are then mapped to the red, blue and green color components,

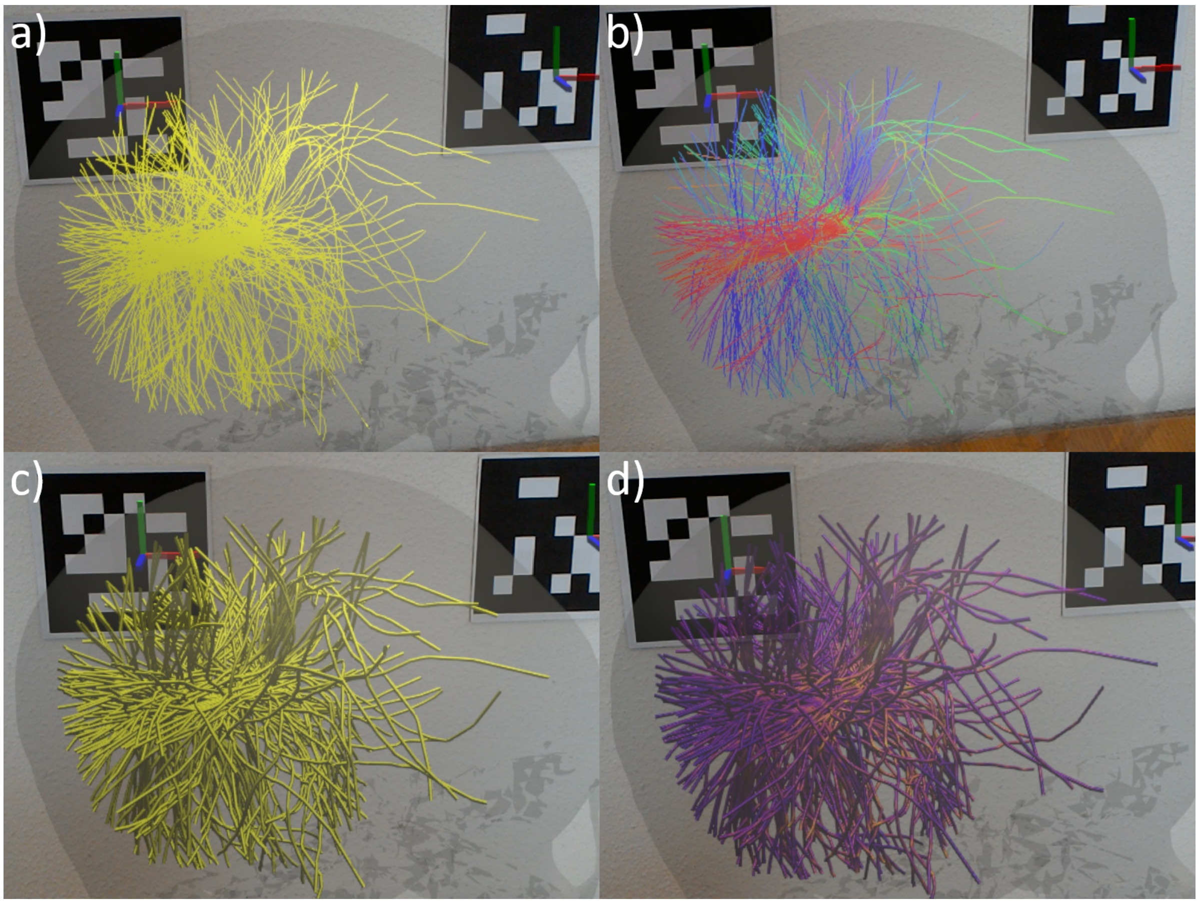

Figure 4.6: DTI fibers rendered in several ways: *a)* rendered as lines with a single color. *b)* rendered as lines with direction mapped to color. *c)* rendered as tubes with single color. *d)* rendered as tubes with additional color information.

respectively. This does allow the user to see the direction of the lines, but it was shown that with this visualization method it is not easy to distinguish the directions [43]. Besides that, this visualization still does not accurately show the depth or which lines are in front of another.

The depth cues that are still in place with line primitives are parallax and binocular disparity in AR. Moving around conveys some depth information of the lines, especially about the relative depth of the lines. Furthermore, in AR you can get some depth information from the difference in the rendering in the left and right display. But even with these two depth cues and the color as direction it is difficult to see how the lines are structured.

Another possibility is to render tubes instead of lines. Every individual line segment is then transformed to be rendered as a small cylinder [37, 51, 62]. Instead of rendering actual tubes, which would require many extra vertices, planes are used. The planes are rotated towards the camera and the fragment normals are manipulated to make the planes look like tubes.

While this implementation is quite straightforward, there are better techniques to visualize the fibers [14, 44, 60]. These methods allow to visualize more individual fibers and fiber bundles, which all should give a better spatial perception of the fibers. Moreover, all the previously mentioned methods look at the DTI data as individual fibers derived from tractography. The actual DTI data is a 3D tensor field with a diffusion tensor at every voxel. And while using fibers allows a simple way to visualize the data, it is actually not really representative. Other studies have focused on visualizing these 3D tensor fields [60]. Nevertheless, we did not look further at these other visualization methods due to time limitations.

For now we will, therefore, focus on visualizing line segments as tubes, see Figure 4.7. The input data consists of both end points of the line segment (v_0, v_2) and the position of the camera pos_{cam}. We calculate in which direction the line should expand to create a plane with the normal to the camera:

$$d = normalize(v_2 - v_0) \quad ,$$

$$n_x = normalize(d \times (pos_{cam} - v_0)) \quad ,$$

$$vertex_{expand} = n_x * r_{tube} \quad ,$$

with r_{tube} as the radius of the tube, which is set to $0.5mm$. Since the line segments are short and the distance to the camera is relatively large, either input vertex can be used for the camera direction with negligible difference. Two new vertices are created v_1 and v_3 at the positions of v_0 and v_2, respectively. We then add $vertex_{expand}$ to the positions v_0 and v_2 and subtract it from v_1 and v_3 to get a plane.

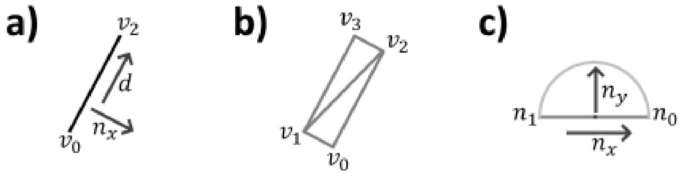

Figure 4.7: *a) Simple line segment, v are the vertices, d the direction of the segment, n_x the direction in which we want to expand the vertices to create a plane. b) Line segment as a plane rotated towards the camera. c) Cross section of plane/tube to show what the normals should be. The actual normals of the points on the surface of the circle are equal to the vector from the center of the circle to the surface point, which is what we calculate.*

The shading of the plane should be such that the normals of the plane follow a cylinder, see Figure 4.7.c. To calculate the normals we define a scalar n_{factor} that has a value of 1 at n_0 and a value of -1 at n_1. This value is then interpolated for every point in between. In the pixel shader we can then calculate a normal following a circle with:

$$n_y = normalize(n_x \times d) \quad ,$$

$$n_{pixel} = normalize(n_{factor} * n_x + (1 - |n_{factor}|) * n_y) \quad .$$

This gives the normal for every pixel, a regular surface shader can then be used to render the plane exactly like a tube.

In this approach, when lines are not perfectly aligned they overlap on one side and leave a small gap on the other side, see Figure 4.8.a. This is especially noticeable with a relatively large tube radius or if the angle between the segments increases.

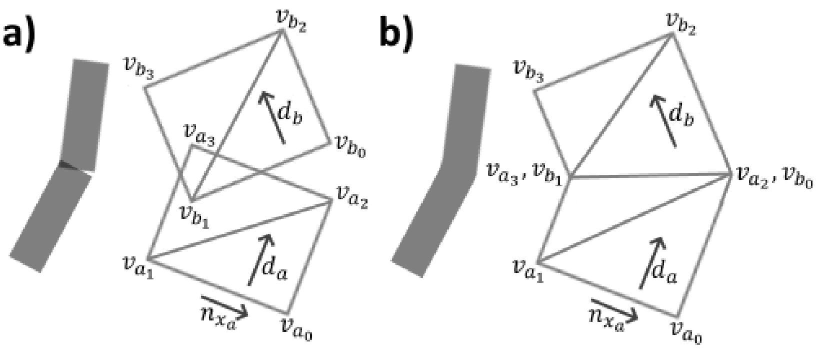

Figure 4.8: *a) When line segments connect on an angle we get an area of overlap and a gap, this is especially visible with a large angle or tube radius. b) To solve this we move v_{a_2} and v_{a_3} along d_a, the amount of movement along d_a depends on the angle between d_a and d_b. The same is then done for v_{b_0} and v_{b_1}.*

We address this by moving vertices along the segment direction to align with the vertices in previous and next segment, see Figure 4.8.b. Thus we calculate the position correction for all vertices:

$$\theta = n_{x_a} \cdot normalize(d_b \times (pos_{cam} - v_{a_0})) \quad ,$$

$$d_{connect} = \begin{cases} d_a, & \text{for } n_{x_a} \cdot d_a < 0 \\ -d_a, & \text{else} \end{cases} \quad ,$$

$$v_{a_{connect}} = tan(\frac{1}{2}cos^{-1}(\theta)) * r_{tube} * d_{connect} \quad .$$

Note that for θ we cannot simply use use the dot product of d_a and d_b, because this would also take the angle along the camera direction in effect, which would be incorrect. $v_{a_{connect}}$ can now be added to the original vertex position to connect the line segment to the previous and next segment.

The shaded tubes themselves convey some direction and depth information, see Figure 4.6.c. Since a color is not used to convey the direction, this can be used to show additional information. In Figure 4.6.d we used the fractional anisotropy for this, but other variables can also be used.

4.4. Planning map

This next section describes the construction of a planning map. This planning map is intended to give an idea of how safe it is to perform a craniotomy for tumor resection from a certain orientation. The planning map is based on structures that are between the skull surface and the target lesion. Figure 4.9 demonstrates the concept in an example of a 2D cross section.

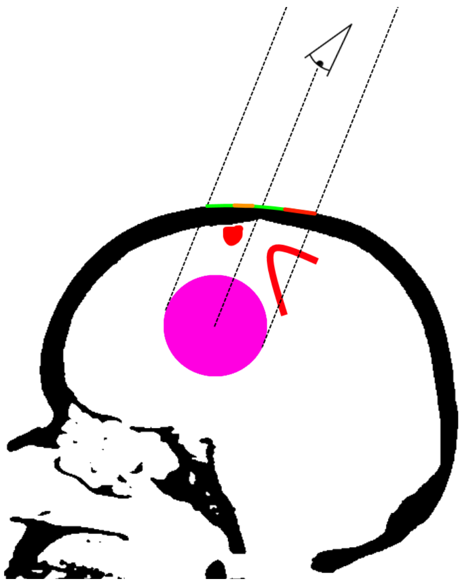

Figure 4.9: 2D cross section of the general idea of the planning map. The black parts are the skull, pink is the tumor, and red are structures to be avoided (for example important vessels). Also shown is the view of the user, where the relative location of the camera to the tumor determines where the planning map is projected on the skull. The result of the planning map is a texture that is projected on the skull surface, showing which parts should be avoided. In this image, green means it is safe and red should be avoided, with orange in between. Weights can be assigned to certain structures to have more or less influence in the resulting texture.

For the planning map we will construct a 2D texture that can be projected on the skull. This texture can be created using a projection of the structures to the skull surface. The result is a scalar value for every pixel, which can be converted to a color using a color map. The basic process only uses standard matrix transformations which are highly parallelizable in the GPU, resulting in little extra rendering time.

4.4.1. Aggregation

To perform the projection we do an additional render pass, where all required structures are rendered an additional time, but in a different way. We perform this projection as an orthographic projection, see Figure 4.10. All structures in this box are then aggregated to get a single resulting value for every pixel.

The direction of this projection is from the position of the camera to the center of the tumor. A projection direction parallel to the viewing direction was also tried, but this proved to be unintuitive. The planning map is always directly in front of the tumor, because the direction follows the position of the camera.

As we explain in Subsection 4.4.3, the resulting texture should have a spatial resolution of $1mm^2$. For different tumor sizes this is achieved using:

$$resolution = nextPowerOfTwo(2 * r_{boundingSphere}) \quad ,$$

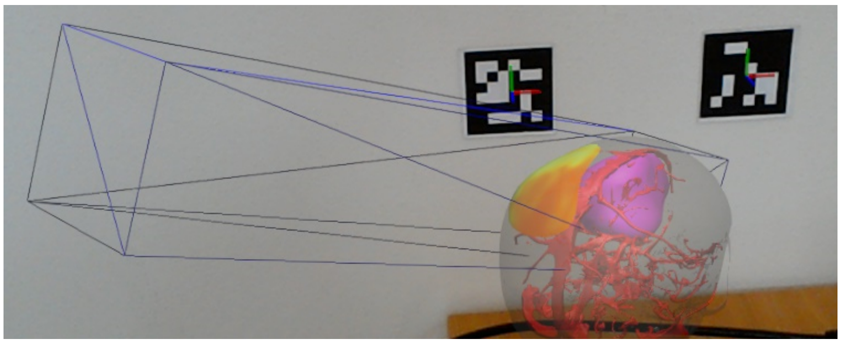
Figure 4.10: The blue mesh shows a box where the orthographic projection takes place. Normally the planning map would be directly between the tumor and the camera, but for demonstrative purposes it is shown at an angle.

where $r_{boundingSphere}$ is the radius of a bounding sphere of the tumor in millimeters. The $nextPowerOfTwo$ function returns the smallest higher integer that is a power of two, a resolution of a power of two is required for efficiency reasons.

The width and height of the orthographic projection is then set to this resolution in millimeters. The near depth of the projection is set to the center of the bounding sphere plus $35cm$ towards the camera, an arbitrary distance which makes sure every structure in front of the tumor is included. The far depth is set to the bounding sphere center plus $r_{boundingSphere}$ millimeter away from the camera.

In the resulting texture, specific color components have a specific meaning. The red color component will be used to show that a pixel contains important structures, the green component shows that that particular pixel is in front of the tumor, the blue component is unused.

Only structures in front of the tumor should be rendered. To achieve this we use depth testing, setting all pixels that are not in front of the tumor to zero depth. Then we draw the structures and aggregate them, for this we use an additive alpha blending method:

$$C_o = C_a \alpha_a + C_b \alpha_b \quad ,$$

$$\alpha_o = 1 - (1 - \alpha_a)(1 - \alpha_b) \quad .$$

See Code listing 1 for a description of how the depth testing and alpha blending was set up. Figure 4.11 shows the rendered results with some intermediate steps.

```
ClearDepthBuffer(0)                // closest possible value
ClearRenderTarget((0, 0, 0, 1))    // set background opaque black
DisableDepthTest()                 // do not test for depth
Draw(tumor, (0, 1, 0, 1))          // draw tumor in green
EnableDepthTest()                  // require closer depth
SetDepthWriteMask(0)               // disable depth buffer writing
SetColorWriteMask((1, 0, 1, 1))    // disable writing green
EnableAlphaBlending()              // allow blending transparent objects
for (s in structures)              // for every structure s
    Draw(s, (1, 0, 0, s.w))        // draw red, structure weight as alpha
```

Code listing 1: Pseudocode of how the planning map texture is created. We start by clearing the render target and the depth buffer. The next step is to render the tumor, this is rendered in green, because the depth buffer is set to `0`, depth testing needs to be temporarily disabled. After drawing the tumor we can enable depth testing, requiring a smaller value, this way only fragments in front of the tumor are rendered. Writing to the depth buffer is disabled for other objects, preventing occlusion of structures. Writing to the green color component is also disabled, to prevent incorrect blending of the green component by other structures. Now we need to render all other structures, we can then aggregate the weights w of the structures using standard alpha blending. Note that the result is order independent, this is usually not the case for alpha blending, but when the color component is always `1`, and only varying the transparency, the order does not matter.

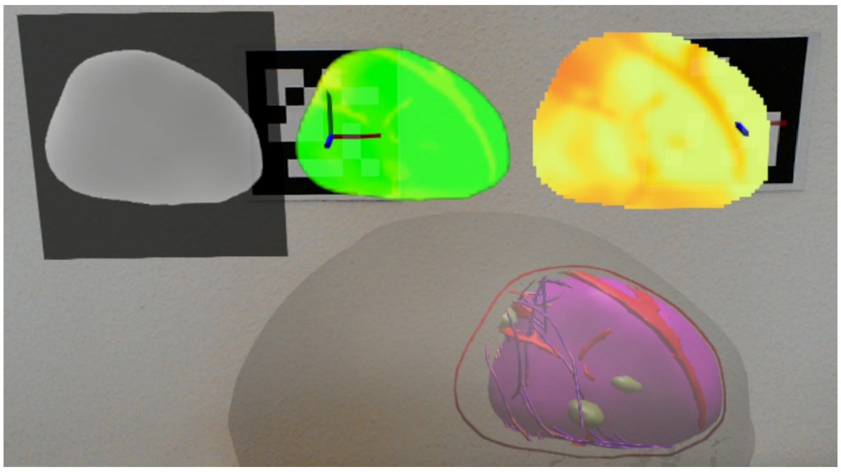

Figure 4.11: Regular rendering with some debug panels shown, the red outline shows were the planning map would be. The left part shows the depth buffer, with black the closest and white the furthest. The image in the middle shows the tumor in green, other structures are rendered in red with green color component writing disabled, resulting in yellow. The right image shows the blurred result with a color map.

4.4.2. Weights

The weights of the vessels and the DTI fibers are set to a constant value, meaning that the value does not depend on the thickness of the structure. We assume that both structures are similarly wide as they are thick. Wide structures do contribute more to the planning map, because they have more visible surface area. The weight of the vessels and the DTI fibers is set to 0.5 and 0.2 respectively.

For the fMRI areas, we cannot use a constant weight, as we would like to take the shape into account. One way to do this would be with an additional render pass, where the depth of the backside of the fMRI area would be written. Then this backside depth with the depth of the fragment itself can be used to calculate the thickness. But because there can be multiple separate surfaces which are all not necessarily convex this is not possible, as it would require multiple depth values to be written in the same pixel.

Instead, we use a simple ray tracing procedure through the 3D binary texture of the fMRI areas. The spatial resolution of the fMRI scan, where this 3D texture is based on, is approximately $3mm$, this is also set as the spatial resolution for this texture, resulting in a texture resolution of $64 \times 64 \times 36$ voxels. This resolution is low enough to not hinder performance significantly. See Code listing 2 for a description of the ray tracing algorithm that is used to render the fMRI areas for the planning map.

```
res = w                                     // initial fragment also counts
in = 1                                      // are we still in the fMRI area?
pos = worldToVoxel * fragmentPos            // set current position
step = worldToVoxel * (mapDir * 0.003)      // set step vector, length is 3 mm
while (in > 0.5)                            // is pos at least half in area?
    pos += step                             // move to next position
    in = texture.sample(pos)                // sample from 3D texture
    res += in * w                           // add step to result
return (1, 0, 0, min(res, 1))               // return color with res weight
```

Code listing 2: Pseudocode of the ray tracing used for determining the weight of a fragment of an fMRI area. The ray tracing itself consists of a loop which every step checks if it is still in that fMRI area. For this we need an initial position pos and a step direction $step$, every step length is set to $3mm$. We know the relation between the voxels and the model, and from the model to the world, thus we can calculate the $worldToVoxel$ transformation. $mapDir$ is set to a vector from the tumor center to the camera. Now in every step we do a texture lookup and see if the interpolated value is higher than 0.5, repeating until the ray tracing algorithm moves out of the fMRI area. The number of voxels passed is then counted and multiplied with a weight w, which is set to 0.08 in our visualization.

4.4.3. Skull rendering

The result of the aggregation is a texture which we render on top of the skull surface. The green component contains a mask image of the tumor and the red component describes the combined weight of the structures between the tumor and the skull surface.

During the skull rendering we now need to map the texture texels to the skull fragments. We use the same orthogonal projection matrix used before. The result is a screen space coordinate of where the skull fragment would end up in the texture. This position can then be used to look up the planning map color from the texture.

To be able to better distinguish the amount of structures, the red color component is mapped to an RGB color. As shown in Section 5.3, in the HoloLens not all color maps work as well as they would with a conventional display. Because of this the Wistia color map is chosen for the planning map.

Specular highlighting has been turned off when rendering the planning map, adding white spots would influence the color map perception. The opacity of the fragments that are part of the planning map is increased to 0.95, otherwise, because of the colors of the structures behind the planning map it would be hard to differentiate the colors on the planning map itself.

Currently, individual structures can clearly be identified on the planning map. This might be desirable, but in our case we want to give a quick overview of where there are more or less structures, instead of an overview of the structures themselves, which can be achieved with a normal rendering. To overcome this we add a Gaussian blur, this makes the result somewhat less accurate, but it is more clear what regions should be avoided at a glance, see Figure 4.12.

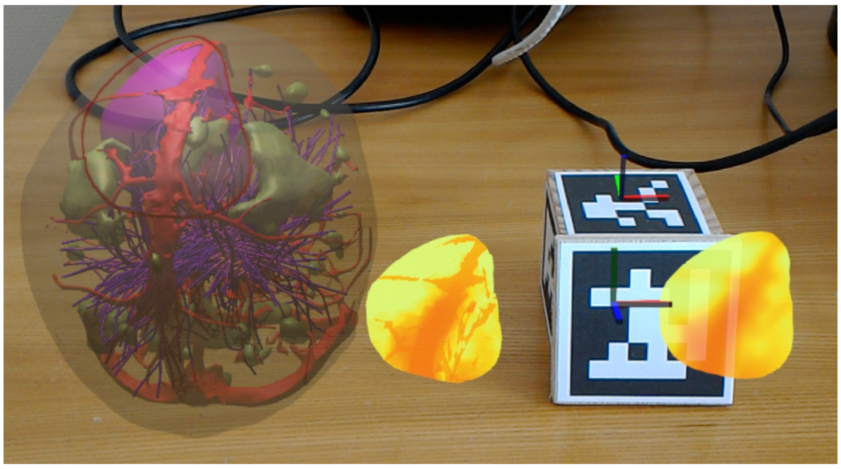

Figure 4.12: Planning map without and with Gaussian blur next to each other. On the skull itself we can see what structures are between the tumor and the skull surface. The left planning map is without the blur filter, where the individual structures are clearly seen. In the right planning map a blur is applied, the structures are faded, which should give a quicker overview of the amount of structures.

We want the blur to be consistent for every scenario. This requires a constant spatial resolution of the planning map texture, which we set to $1mm^2$ per pixel. For the Gaussian blur the original texture is first downscaled by a factor two using a mipmap with two lower resolution levels. Then when rendering skull fragments containing the planning map, a $20 \times 20mm$ square of the texture is retrieved using 25 texture look ups at the lowest resolution MIP level. This is then convoluted with a Gaussian kernel with $\sigma = 1$, resulting in a Gaussian blur over a spatial distance of $20mm$.

4.4.4. Operating the planning map

Normally the position of the planning map is on the skull surface, exactly between the user and the center of the tumor. This is also how the user can move the planning map over the surface of the skull. Thus to see the planning map on the other side of the skull, the user has to physically move to the other side of the skull to see it there. This might not be the most practical way to move the planning map, but no other solutions were implemented.

Because the planning map is always in front of the tumor, it always prevents the user from seeing the tumor itself. This might not always be desirable, therefore it is possible to turn the planning map off. Now the normal skull surface is shown, but with a red outline of the planning map shown.

When a particular orientation of the planning map needs to be further investigated, the user can lock the position of the planning map allowing to better see the structures in front of the tumor. It is also possible to turn off all rendering except what is between the planning map and the tumor, this way it possible to clearly inspect the specific structures, see Figure 4.13.

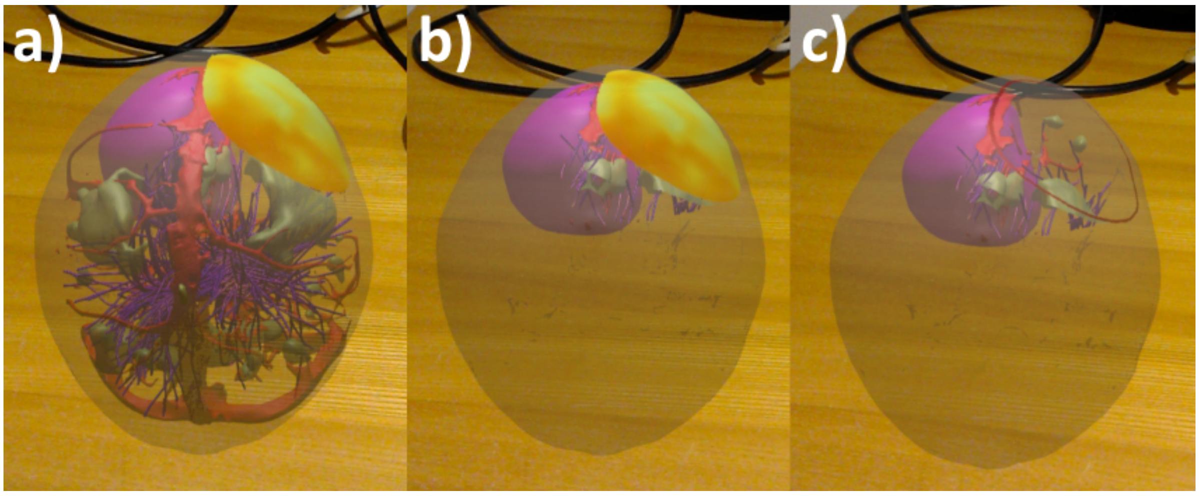

Figure 4.13: *a)* The planning map is locked at a position, allowing the user to better evaluate the position where the planning map is locked. *b)* Only render structures that are between the tumor and the planning map, this way it is clearly visible exactly what structures are in between. It also prevents structures that are not in between from occluding relevant structures. *c)* Hiding the planning map allows to see what is directly underneath it. An outline of the planning map is still shown in red.

4.5. Results

In Figure 4.14 we show the resulting visualization that was created in this work. Figure 4.15 shows the accompanying workflow that can be used with this visualization.

In Table 4.3 we show the frame render time of the visualization. In most cases the rendering time is just higher than the recommended $16.7ms$.

	Frame render time	
Patient dataset	**With planning map**	**Without planning map**
Case 1	$17.5ms$	$16.7ms$ (91.3% GPU utilization)
Case 2	$21.1ms$	$17.5ms$

Table 4.3: Frame render time of the visualization.

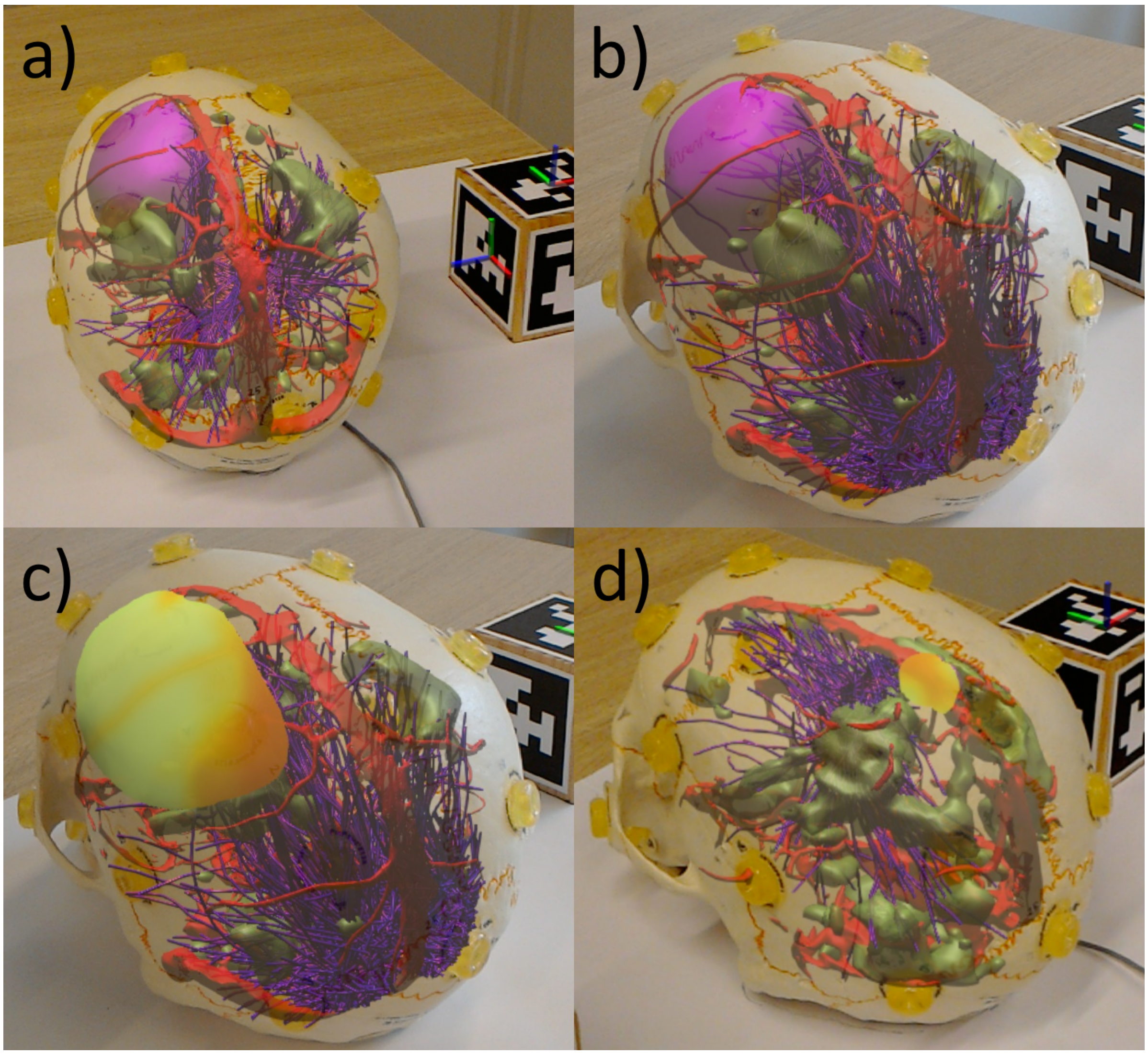

Figure 4.14: Visualization of surface models and the planning map. Due to the inaccurate reprojection the alignment of the models to the phantom is less accurate than when the HoloLens is worn. The image capturing shows the models more opaque than when the HoloLens is worn. *a)* Shows the rendering of the models for the *case 2* dataset. *b) Case 2* dataset with a different segmentation of DTI fibers. *c)* Same visualization as *b* but with the planning map turned on. *d)* Visualization for the *case 1* dataset, including the planning map.

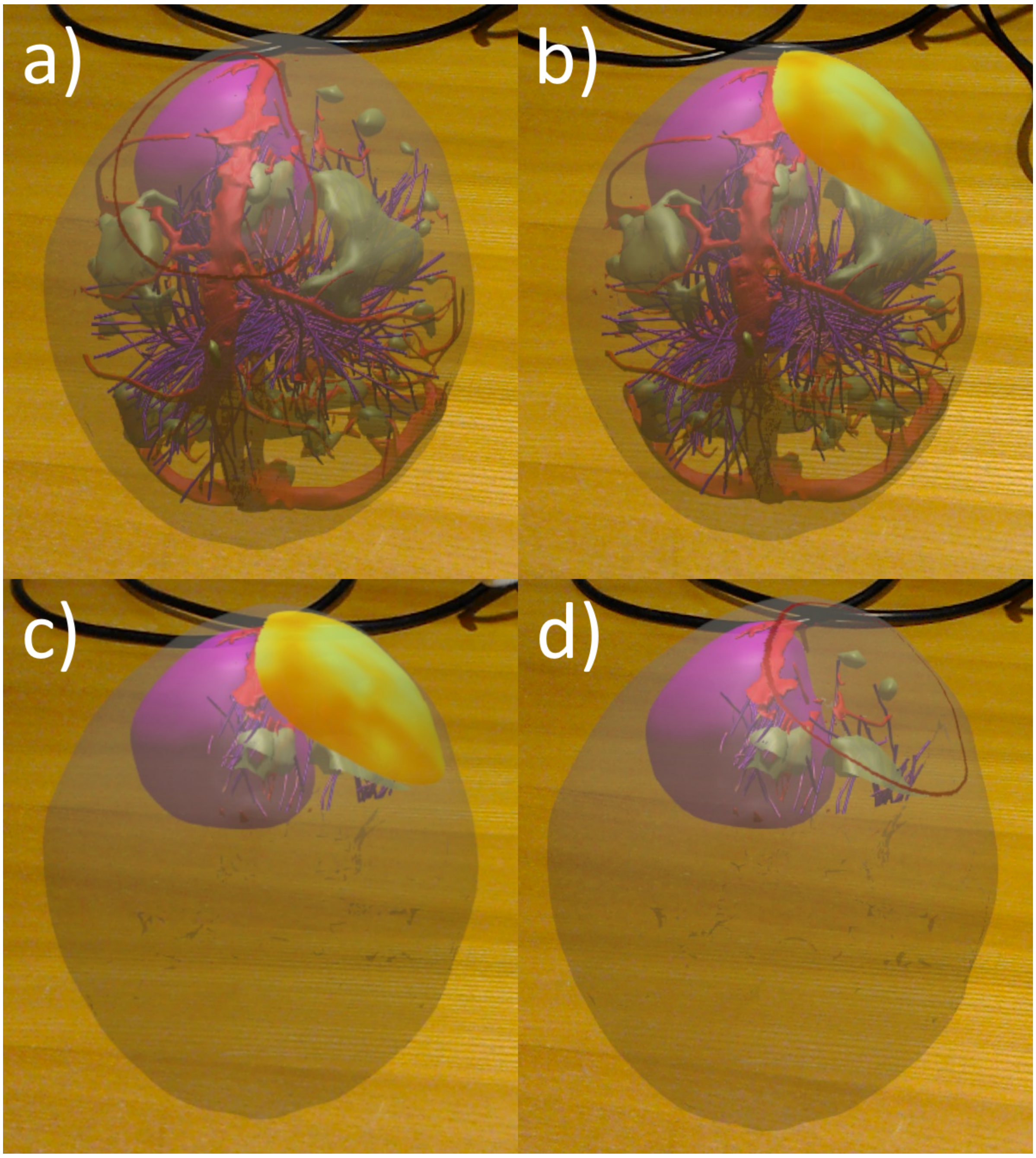

Figure 4.15: Workflow where the visualization may help in finding an optimal craniotomy position. The image capturing shows the models more opaque than when viewed with the HoloLens. *a)* First the structures can be explored to get a general idea of the spatial relations of the structures. *b)* Then using the planning map a location can be found for the craniotomy, this orientation can than be locked. *c)* With the locked orientation, only the structures between the skull surface and the tumor can be rendered. *d)* Then the planning map can be turned off to see that what structures would be removed in the access path to the tumor.

5 Experiments

This chapter describes some experiments that were performed. The first two experiments assessed the accuracy of displaying models relative to tracked markers. In the second experiment we investigated the performance of some color maps with the HoloLens.

5.1. Marker tracking

In this set of experiments we evaluated the performance of tracking markers using ArUco in the Holo-Lens.

5.1.1. Accuracy

The purpose of this experiment is to assess the accuracy of the marker tracking. For this experiment the tracking in the HoloLens will be compared with the tracking from an optical tracking system. See Figure 5.1 for an overview of this experiment.

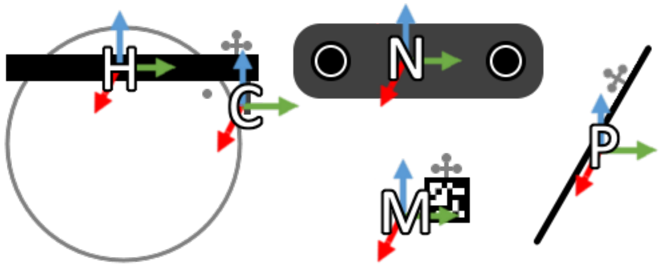

Figure 5.1: Overview of the experiment to find the accuracy of the marker tracking. The markers tracked by the HoloLens are in the coordinate system of the HoloLens camera C. The coordinate system of the tracking system N will be used as the reference space. Spheres are attached to the HoloLens T_N^H, the marker $T_N^{M_{spheres}}$ and the pointer T_N^P.

During the experiment the markers were hand-held and moved around slowly in a random motion. We want to find the affect of several variables on the accuracy:

- *Camera resolution:* The tested resolutions are: 1408×792, 896×504, 704×396, and 448×252, the latter two are linearly interpolated downscaled frames created from the first two resolutions.
- *Marker size:* Four marker sizes were tested: 15, 30, 45 and 60 millimeter. The distance of the marker to the camera was kept relatively constant, at arm's length, about $50cm$.
- *Incident angle:* We tested the accuracy at approximately $0°$, $20°$, $40°$, and $60°$. As the marker were hand-held, the exact marker inclination differed from these angles.
- *Marker motion:* No method was used to measure the accuracy at specific values. Instead, during the experiment the markers were moved randomly, then afterwards the motion was retrieved from the logged poses.

The optical tracking system used is the Polaris Vega VT[1]. This uses IR-reflective spheres as markers, which we call, henceforth, spheres for short. This system can track the spheres with an RMS of $0.12mm$. The coordinate system of the optical tracker N will be used as the reference space. Markers tracked by the HoloLens are in the coordinate system of the HoloLens camera C. Both systems give a pose of the markers or spheres as a translation and rotation, we refer to these as a rigid transformation from the base coordinate system to the marker, for example T_C^M.

Spheres were attached to the HoloLens using some adhesive putty T_N^H, see Figure 5.2.a. Using a clamp, spheres were also attached to a marker $T_N^{M_{spheres}}$, see Figure 5.2.b. A pointer provided with the tracking system was used to retrieve the positions of the marker corners relative to $M_{spheres}$, these corner positions were then used to calculate $T_{M_{spheres}}^M$, see Figure 5.2.c. We then calculate the marker pose in N:

$$T_N^M = T_N^{M_{spheres}} * T_{M_{spheres}}^M \quad .$$

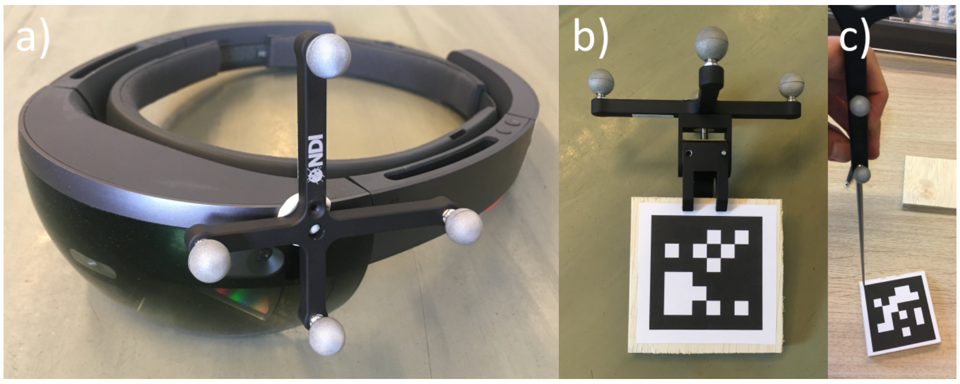

Figure 5.2: Photos of the parts used in the experiment. *a)* HoloLens with optical spheres attached. *b)* Marker with optical spheres attached. *c)* Pointer used to point to the corners of the marker.

With T_N^H and T_N^M we can define the marker relative to the spheres on the HoloLens:

$$T_H^M = (T_N^H)^{-1} * T_N^M \quad .$$

Then we perform a calibration to find the transformation between T_C^M and T_H^M. The optimal transformation is found using a least squares method [1]. Then with this calibration we can find the marker in the HoloLens camera space as tracked by the optical tracker:

$$T_{C_N}^M = T_{calibration} * T_H^M \quad .$$

And finally to find the error, we find the transformation to go from T_C^M to $T_{C_N}^M$:

$$T_{error} = T_C^M * (T_{C_N}^M)^{-1} \quad .$$

We can then split this error into a translation and a rotation as Euler angles.

While we had planned to do the calibration procedure beforehand, an error occurred in the temporal calibration between the HoloLens and the optical tracker. This temporal offset was corrected for by finding a local minimum of the difference of the poses from the HoloLens and the optical tracker. This synchronization error meant that the original calibration was incorrect.

Therefore, we recalculated the calibration afterwards using the poses that were recorded during the experiment itself. Some filtering was performed on the poses, removing outliers and other known inaccurate poses:

- Poses with a distance larger than $70cm$, since the markers were hand-held they could not be further away.
- Poses with a large difference from the previously recorded pose. This can be an incorrect result but often this is just natural movement of the marker, but the accuracy is usually lower so for the calibration we exclude it.

[1]`https://www.ndigital.com/medical/products/polaris-vega/`

- Poses with a large error. We can use the initial incorrect calibration for a rough estimate of the poses and use this to exclude outliers.

From initial results it seemed that there is an error in the camera parameters given by the HoloLens, see Section 7.2. This means that the marker detection error depends on the position of the marker in camera space. Therefore, in the results we only use poses retrieved from a specific area, where the distance from the marker to the center of the frame should be less than $10cm$. We also look at specific depths of the marker, where we chose a section of $4cm$ which contained the most measurements.

In Figure 5.3 we show the poses from the HoloLens and the optical tracker next to each other. If we interpolate the results from the optical tracker to the time points that are recorded by the HoloLens, we can calculate the error of the marker pose detection in the HoloLens.

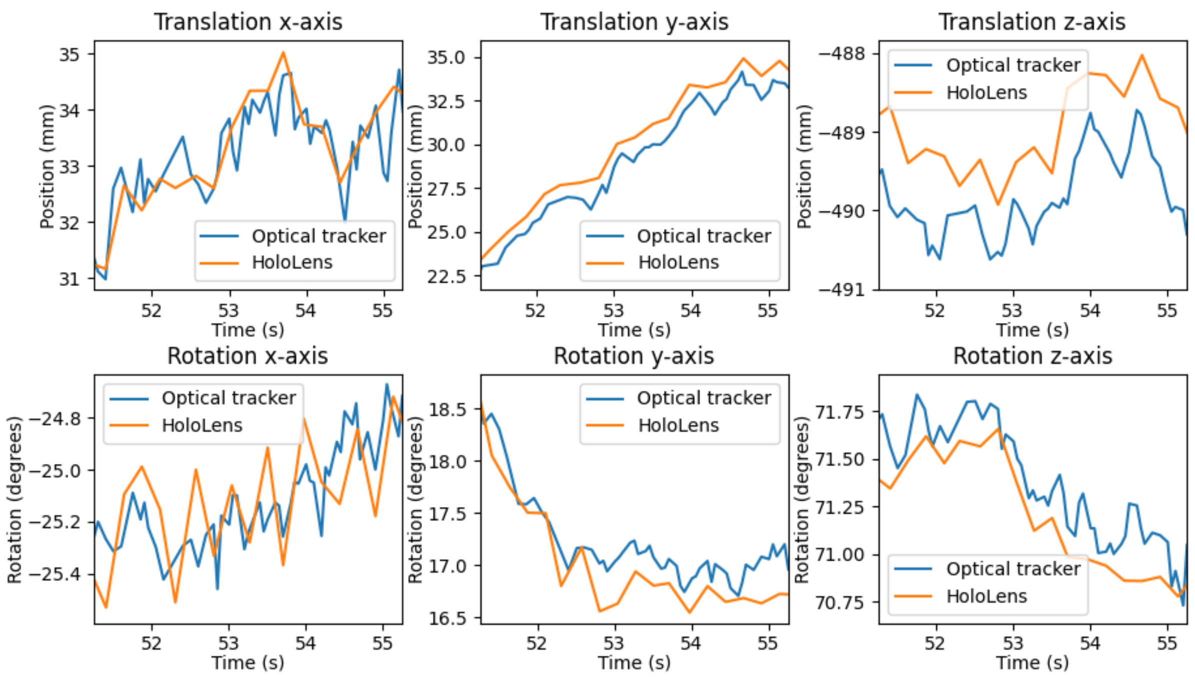

Figure 5.3: Marker pose in the coordinate system of the HoloLens camera. The optical tracker pose is aligned to this coordinate system using the calibration that was performed. This was with camera resolution of 1408×792 and a marker size of $60mm$.

We look at the accuracy for the different resolutions and marker sizes, see Figure 5.4. It is notable that the accuracy increases with larger markers and larger resolutions. We can also see that the results with the camera resolution of 448×252 were poor. For resolution 704×396 the results are similar to those for resolution 1408×792.

Next we look at the incident angle, see Figure 5.5. The translational accuracy simply decreases when the incident angle increases, however, for the rotational accuracy we can see an optimum around $40°$.

Markers can move by rotating or by changing their position. First we will look at the effect of translational motion on the accuracy, see Figure 5.6. The results are as expected, with a lower accuracy when the markers move faster. Note that the speed is calculated using the difference in position and difference in the timestamp. For the optical tracker we interpolate the speed instead of calculating the speed on the interpolated poses.

Next, we look at rotating markers, see Figure 5.7. Here we see that the speed has less influence, for the translation we would expect this, since the position of a marker does not actually change when it is rotated. But for the rotation we also see little correlation between the rotation speed and the accuracy.

5.1.2. Noise

We also performed a short test using stationary markers to be able to report the error introduced by noise in the pose detection. Here a $60mm$ marker and the HoloLens were placed on a table without

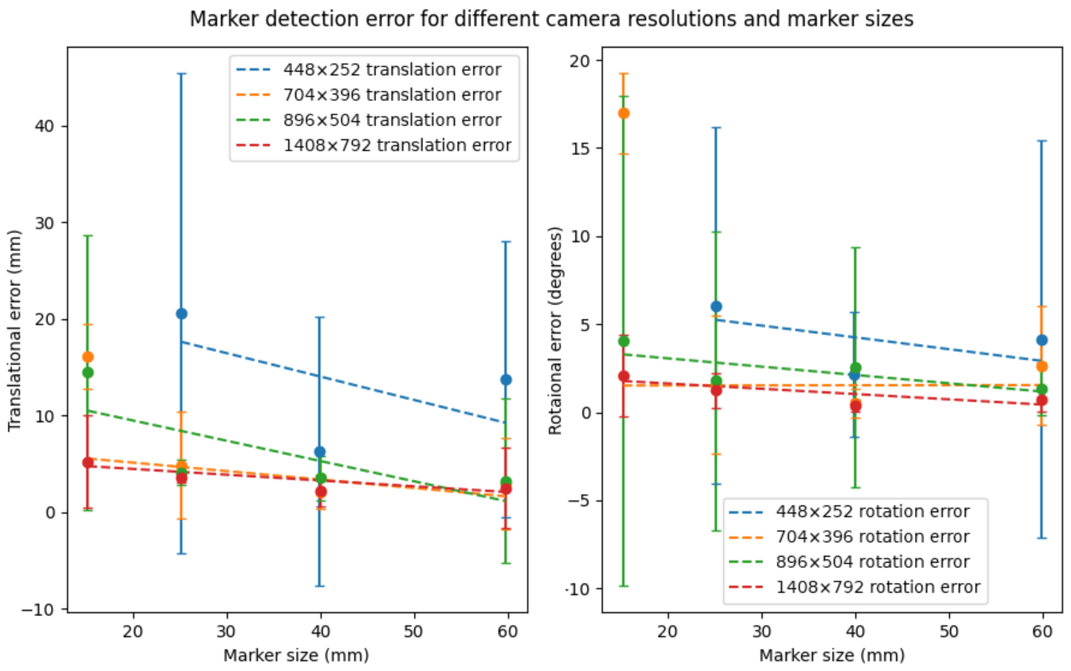

Figure 5.4: Error in marker poses from HoloLens for different marker sizes and resolutions. The dashed line represents the linear regression of the error. For resolution 448×252 with marker size $15mm$ the system was unable to detect the marker.

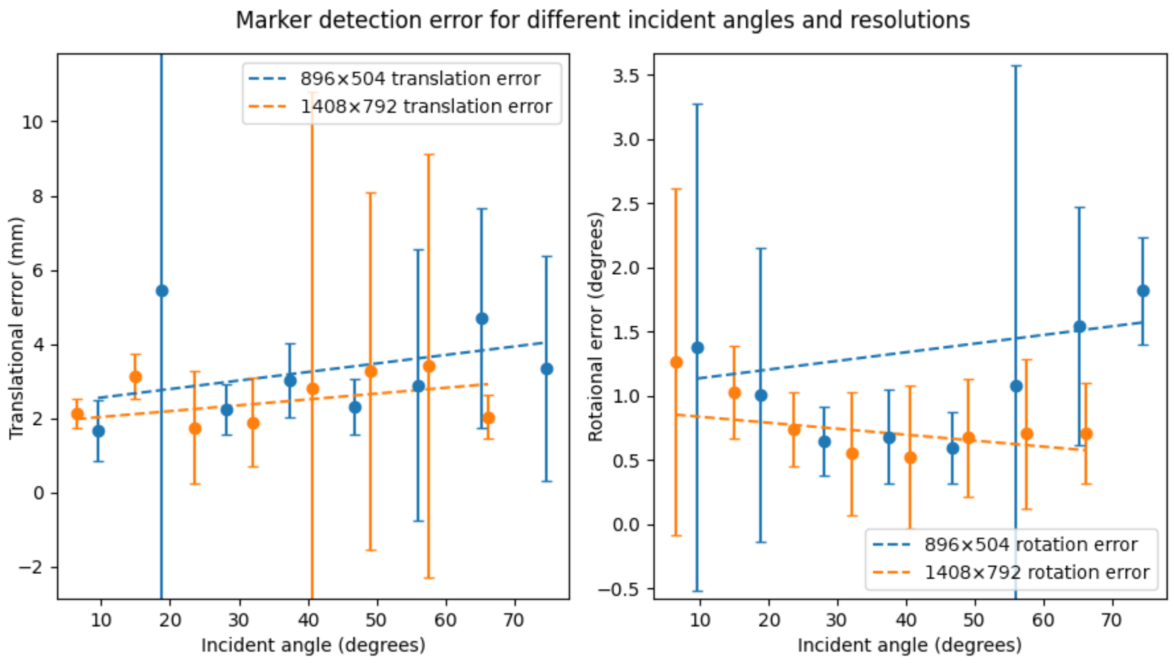

Figure 5.5: Relation between marker detection error and incident angle. The dashed line represents the linear regression of the error. The incident angle is defined as the combined rotation along the x and z axes.

moving them at a distance of $70cm$. We were unable to compare these results to the optical tracker, since the calibration only included a small depth range.

See Figure 5.8 for the results. The noise is roughly 5 times less in the x/y directions than in z direction. Nevertheless, the error introduced by the noise is much smaller then the error found when comparing poses to the optical tracker.

To find factors that influence this we look at the standard deviation of the results. We look at different

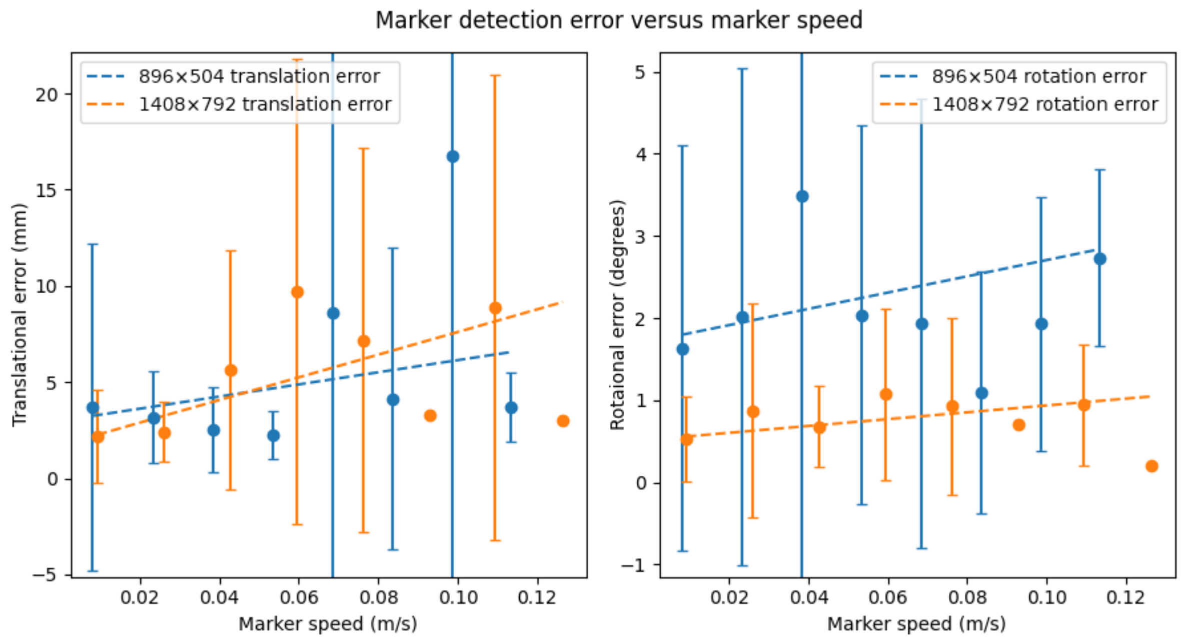

Figure 5.6: Relation between marker detection error and translational changes. The dashed line represents the linear regression of the error.

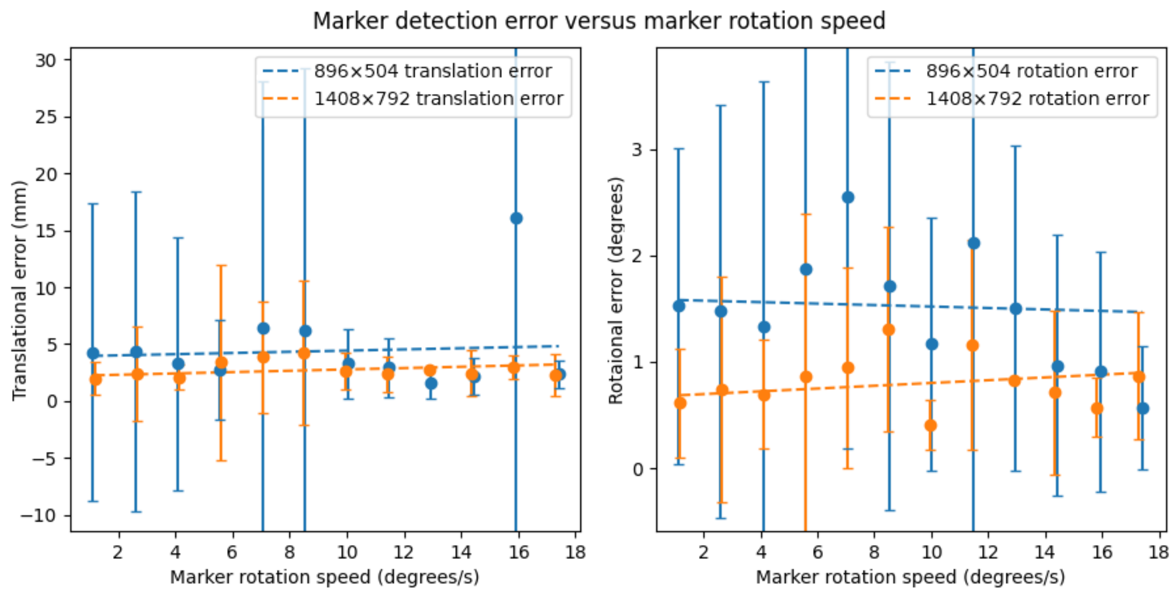

Figure 5.7: Relation between rotation speed and marker detection error. The dashed line represents the linear regression of the error.

resolutions and incident angles, see Figure 5.9. Even for the worst case the noise is quite low, with the maximum amount of noise at less than $1mm$ and $0.3°$.

5.1.3. Kalman filter

To evaluate the Kalman filter we look at the results of the Kalman filter next to the poses given by the tracking without a filter and the optical tracking system. Because the Kalman filter is a temporal filter, a delay is introduced in the pose detection, therefore reporting the error will not give insightful results.

See Figure 5.10 for the results. From the plot we can observe both the strengths and weakness of the Kalman filter. It performs well when filtering out the noise and actual outliers in the detection, peaks

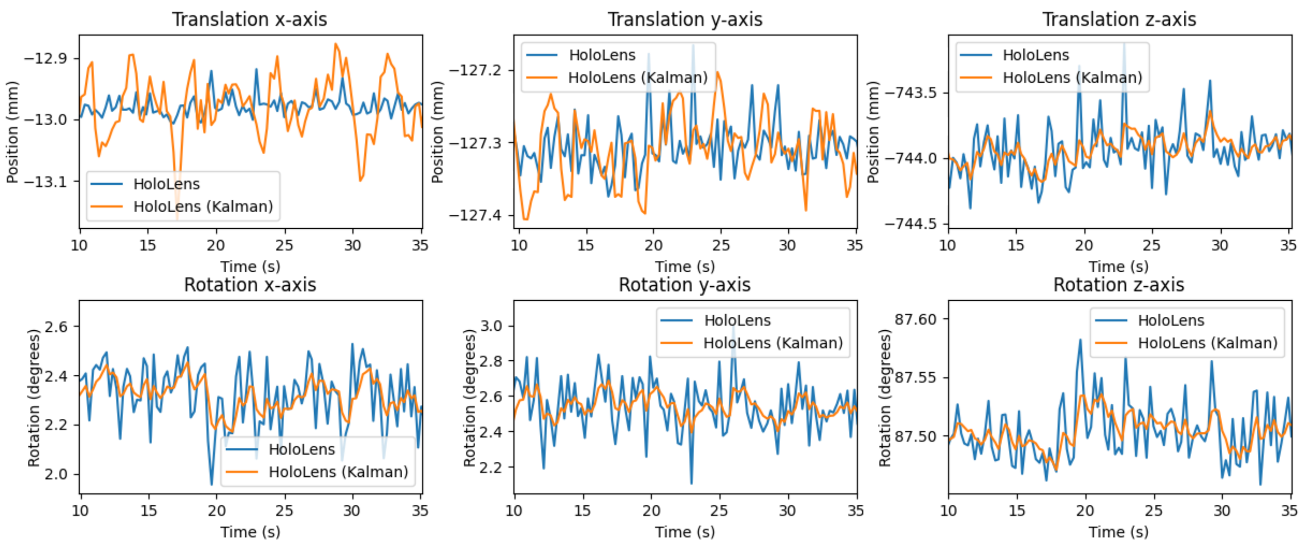

Figure 5.8: Tracking results from a stationary marker, both markers and HoloLens were placed on a table. Camera resolution is 1408×792 and marker size is $60mm$. The distance from the camera to the markers was approximately $70cm$. Note that the Kalman filter does not perform well with very small noise, as it tries to correct too much.

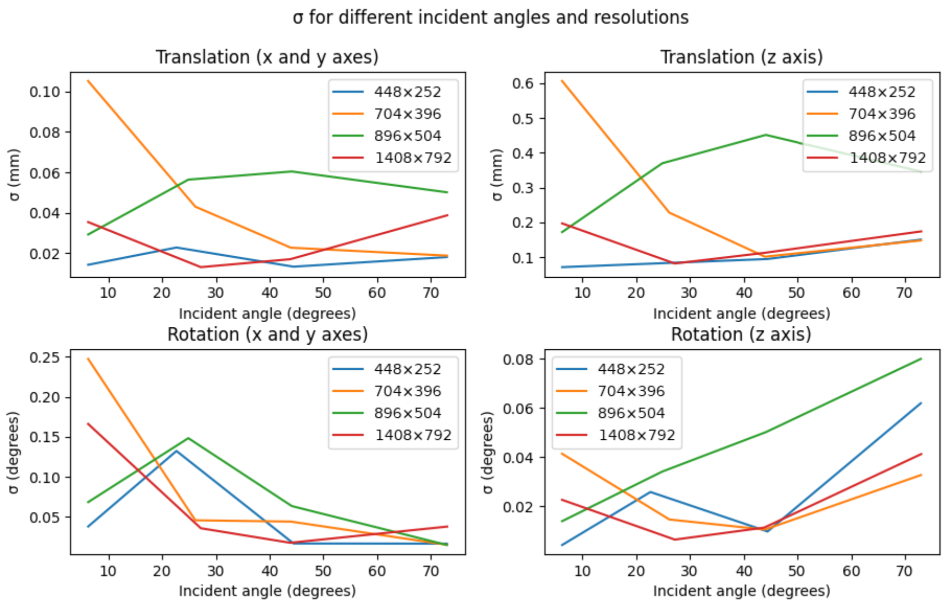

Figure 5.9: Standard deviation of stationary marker detection for different incident angles and resolutions. The marker used in this test were $60mm$ large. The distance from the camera to the markers was approximately $70cm$.

in the pose that are created by noise are filtered out. On the other hand, we also see that it creates a delay in the detection of the markers: if a sudden change in the pose happens, it takes some time before the pose returned by the filter follows that change.

5.1.4. Combinations of markers

To find the error resulting from the noise in a combination of multiple markers, we performed a small experiment using a stationary cube with markers on all sides. In Table 5.1 we show the standard deviation of the detected poses of the individual and combined markers.

These results do not directly give the error of the combination of markers since this does not include the calibration between the individual markers. Unfortunately, we do not have an accurate calibration, therefore, we cannot report the actual accuracy of a combination of markers.

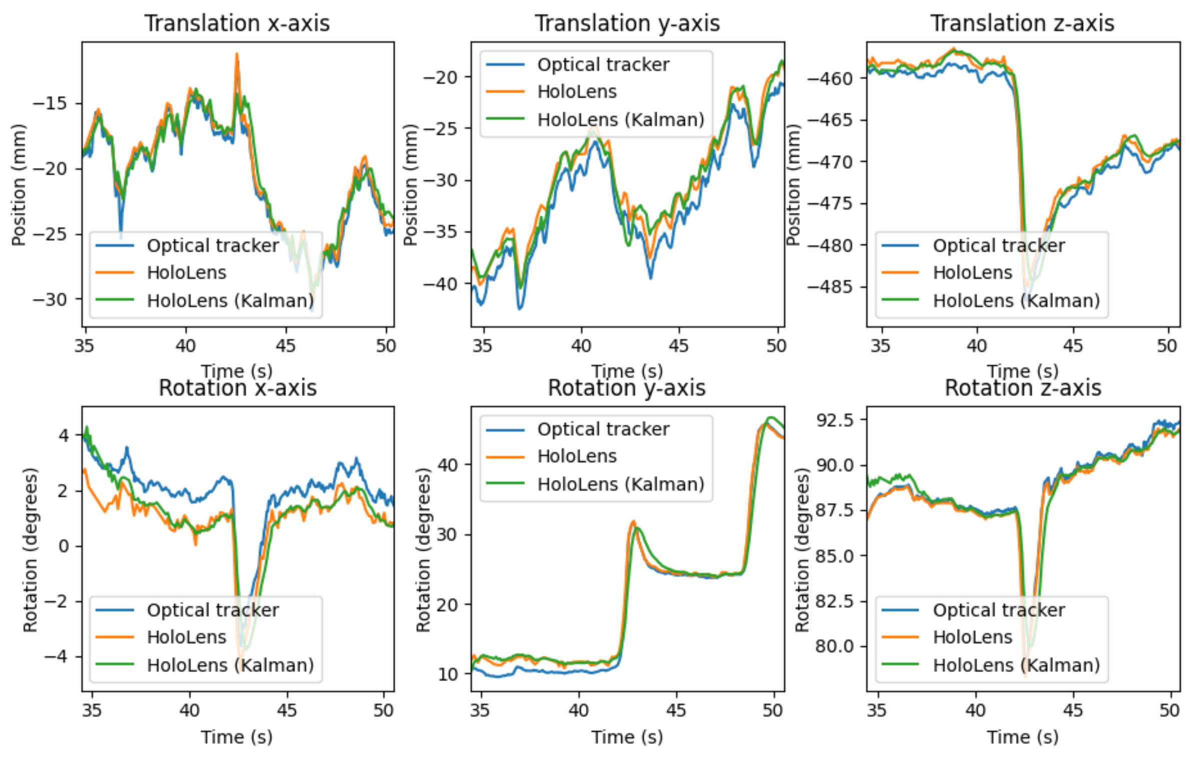

Figure 5.10: Plot of marker poses over time with and without a Kalman filter.

	Translation σ		Rotation σ	
Incident angle	$\lvert x+y\rvert$	z	$\lvert x+y\rvert$	z
$36°$	$0.05mm$	$0.39mm$	$0.09°$	$0.09°$
$65°$	$0.04mm$	$0.55mm$	$0.04°$	$0.05°$
$72°$	$0.09mm$	$1.05mm$	$0.27°$	$0.23°$
Combined	$0.06mm$	$0.47mm$	$0.14°$	$0.06°$

Table 5.1: Standard deviation of tracking a stationary cube with markers on every side. The HoloLens was also placed on a table to make sure there was no movement between the marker and the HoloLens. Tracking was done over 1500 frames with markers of $50mm$ size at $49cm$ distance. The results from the x and y axis are combined, since noise along these axes should be similar.

5.1.5. Frame rate

The frame rate of the marker detection is mostly dependent on the camera resolution that is used. Another important factor is the number of contours that are found during the detection. We tested the frame rate on a white wall with a single $50mm$ marker at around $50cm$ distance. The resulting frame detection times can be found in Table 5.2.

Resolution	Frame detection time
448×252 (downscaled)	$24.6ms$
704×396 (downscaled)	$69.2ms$
896×504	$59.2ms$
1408×792	$101.2ms$

Table 5.2: Average frame detection time over a time span of 10 seconds for different camera resolutions.

5.2. Display and camera registration

We want to find the accuracy of the display and the camera registration. Because an error in either one is indistinguishable from an error in the other, the accuracy of both these transformations is measured together.

On a piece of paper we printed four markers of $60mm$, they were printed $22cm$ apart horizontally and $13cm$ vertically. A dot at a random position between the markers was also printed on the same paper. In the HoloLens a model of this dot is then shown at this same position. During the experiment the printed dot is covered using another piece of paper. The user will then mark the position of the rendered dot which we will later compare to the position of the printed dot, see Figure 5.11. This experiment is a similar setup as in a study by Cao et al. [9].

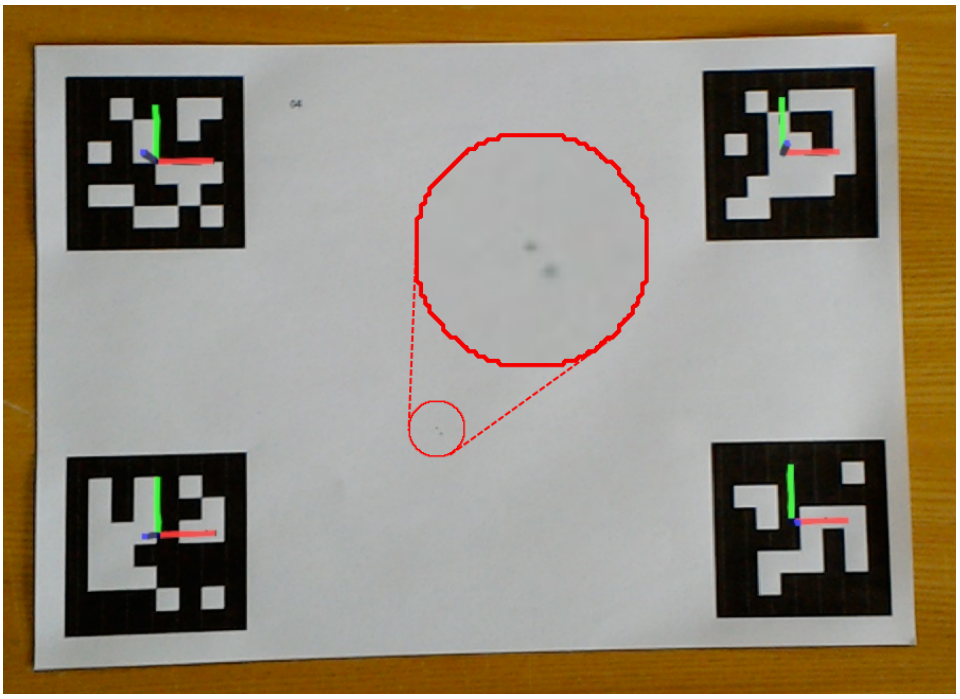

Figure 5.11: Experimental setup to find the accuracy of the display and camera registration. We use four markers of $60mm$ printed in a known pattern. A dot with a width of $0.5mm$ is also printed at a known distance between the markers. In the HoloLens a dot of the same width is shown at the same distance from the markers. The red circle in this figure shows a zoomed-in section of the image containing the two dots. The printed dot is hidden, by placing a piece of paper over the section which can contain the dot. During the experiment the models showing the position of the markers are also hidden, only the dot is shown. The user can mark the location of the model using a needle, also marking it on the original print. Then the distance of marked location with the printed dot can be compared.

Two different incident angles were tested: $0°$ and $50°$. Although these specific incident angles were not enforced during the experiment and are therefore an approximation. Multiple measurements were made, changing the position of the dot and rotating the paper by $45°$ between every measurement.

Because we want to test the error independent of the marker tracking, the settings of the marker tracking were optimized for accuracy.

Before every measurement, a step is taken to make sure the HoloLens is placed on the head correctly. Here a square overlay of a marker is shown and the user views this marker placed flat on a table from above, then the device is placed on the head such that the overlay is aligned with the marker.

During the experiment we found that the relative pose between the markers did not match the horizontal distance of $22cm$. The distance reported by the HoloLens was about 0.5% larger. We then changed the detection parameter that defines the size of the marker. We found that for a marker size of $59.67mm$ the detected distance between the markers was closest to $22cm$. The experiment was therefore carried out twice, once with the settings at $60.00mm$ and once with $59.67mm$.

See Table 5.3 for the results. We see that the results with marker size setting $59.67mm$ are more accurate.

The incident angle is a rotation around the x axis, therefore, it should not affect the results on the

Marker size setting	**Incident angle**	x	y	$\|x + y\|$
$60.00mm$	$0°$	$-0.9 \pm 0.4mm$	$-0.2 \pm 0.4mm$	$1.0 \pm 0.4mm$
$60.00mm$	$50°$	$-0.7 \pm 0.6mm$	$1.6 \pm 1.5mm$	$2.3 \pm 0.7mm$
$59.67mm$	$0°$	$-0.6 \pm 0.4mm$	$-0.1 \pm 0.7mm$	$0.8 \pm 0.6mm$
$59.67mm$	$50°$	$-0.6 \pm 0.7mm$	$1.1 \pm 0.7mm$	$1.5 \pm 0.4mm$

Table 5.3: Visualization error results for different marker size settings and incident angle. For both settings of the marker size, the same printed markers of $60.00mm$ were used.

x axis, which it indeed does not. We do see a large increase in the error in the y axis with a larger incident angle. Before every measurement, the device was made sure to be placed correctly on the head since placing it higher or lower on the nose changes the position of the models on the y axis. Although it should be noted that this also inherently moves the display closer or further from the eyes. We would expect the result for the $0°$ incident angle to be zero on the y axis, because of this alignment. The large error for the results of the $50°$ measurements is therefore probably due to an offset in the z axis.

5.3. Color maps

Since in the HoloLens dark colors are mapped to transparency we cannot assume that color maps perform in the same manner as they would on regular displays, see Figure 5.12. Therefore we want to find out what color maps do work well on the HoloLens. The first step is to find the color mapping used in the HoloLens which calculates the opacity. Then use that to composite the partially transparent color map over a background and see how well it performs.

The following formulas describe the dark color to transparency color mapping:

$$C_{max} = max\{C_r, C_g, C_b\} \quad ,$$

$$R_{r,g,b} = C_{r,g,b}/C_{max} \quad ,$$

$$R_a = C_{max} * C_a \quad ,$$

where $C_{r,g,b,a}$ is the rendered color and $R_{r,g,b,a}$ is the resulting color on the HoloLens.

The color maps evaluated in this experiment are the ones used by *matplotlib*[2]. This collection of color maps also includes color maps from some external sources. The collection consists of 228 color maps, we filtered this to reduce the size of this set. Only perceptually linear color maps are of interest, diverging or categorical color maps were excluded. Because of some overlap in the color maps from different sources, duplicate color maps in the set were discarded. After this filtering we are left with 58 color maps to analyze further.

A metric of how perceptually uniform a color map is, is the lightness $L*$ value in the CIELab color space. Linear change of the other variables in this color space should also provide perceptually linear color maps, but these do not perform well when the size of the subject is small [27]. Therefore we will only use the $L*$ parameter to evaluate the color maps. For easier analysis we use the CAM02-UCS color space, which has the same relevant properties and variables as CIELab [32].

Of a good color map we expect that the $L*$ value to linearly increase or decrease, this means that the color map is perceptually uniform. Furthermore, the linear change should be over a large range of $L*$ values, this makes it easier to distinguish between the different values. These properties should also hold when the color map is used in the HoloLens and composited over a non-uniform background.

Because all dark colors are mapped to transparent, the color map value is composited over whatever the user is looking at. For this experiment we use two backgrounds over which the color maps are composed, one is a white background, the other is a sine wave from black to white with 5 periods over the whole length of the color map. Note that the result for a black background would simply be the original color map. See Figure 5.13 for an example of the *Reds* color map composited over different backgrounds. We can see the performance of the color map using a plot of the lightness values, see Figure 5.14.

[2]`https://matplotlib.org/3.1.1/gallery/color/colormap_reference.html`

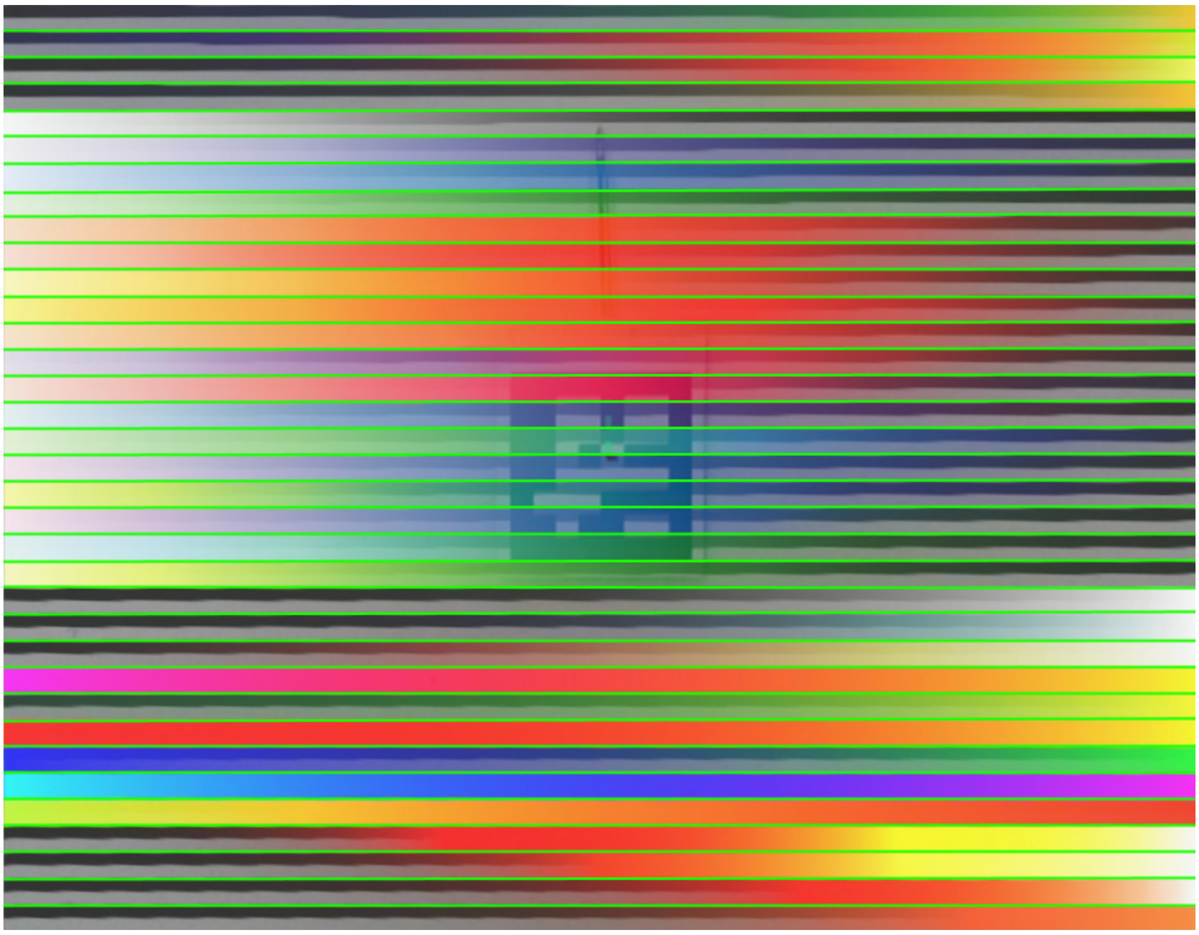
Figure 5.12: Visualization of some randomly selected color maps and how they look when rendered in the HoloLens, with dark colors mapped to transparency. Every color map is divided by a green line, every first line contains the original color map, every second line is the same color map with dark colors mapped to transparency. Many of these color maps are by design perceptually uniform, but are not when this mapping is applied.

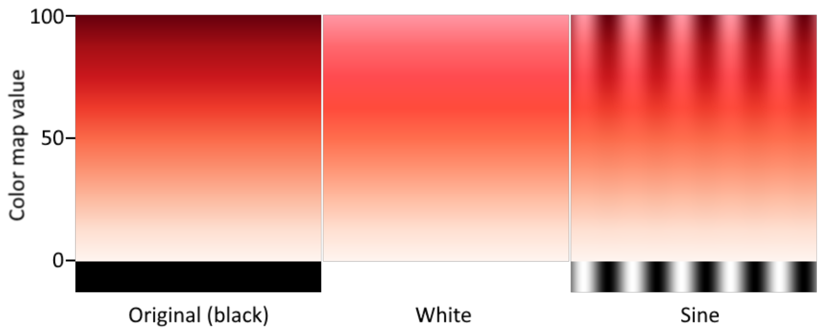

Figure 5.13: *Reds* color map composited over the three different backgrounds used in this experiment. Since this color map is lighter for lower values the background is less visible there. Since dark colors are mapped to transparent the color map composited over a black background is exactly like the original.

We have a set of 58 color maps to evaluate, this is a too large set to display all the results of. Instead we show some examples of bad performing color maps and use these examples to rule out a large portion of similar color maps. Then we will show the lightness plot for the color maps that are not excluded.

In Figure 5.15 we show some common ways in which color maps fail on the HoloLens:

- *Reds*: The color map clearly goes from a very light to a dark color. With a white background this

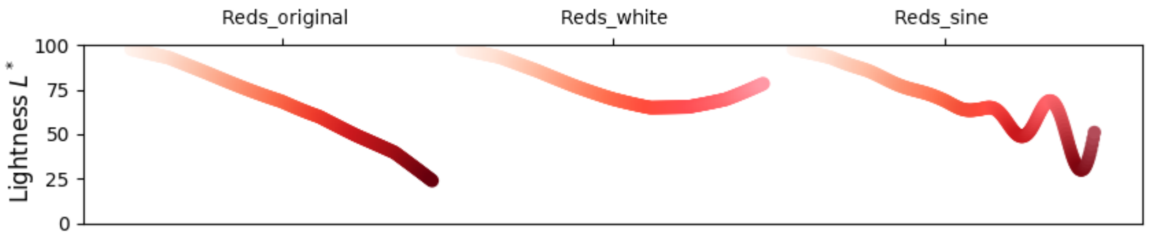

Figure 5.14: *Reds* color map lightness plots, one plot for every background. On the x-axis the color map input value linearly increases, the color is the mapped result for the color map. Of this color the lightness values is taken which is plotted against the y axis.

creates an almost divergent color map. And for the higher values the color map becomes too transparent to be usable. We excluded 12 color maps like this.

- *CET_L3*: Like the *Reds* color map, but from dark to light. In this specific color map it starts with black, which would be completely transparent. We excluded 12 color maps like this.
- *CET_L13*: Like the *CET_L3* color map, but not using the lighter values. This creates an even worse result, where with the white background the color map is inverted. We excluded 3 color maps like this.
- *magma*: Also like the *CET_L3* color map, but with different colors in between. This creates some W-shaped pattern with a white background, clearly not suitable for the HoloLens. We excluded 10 color maps like this.
- *CET_D8*: While the lightness values of this color map are linear, it is meant to be a diverging color map. With the gray portion in the middle, the color map fails when this would be rendered on the HoloLens. We excluded 2 color maps like this.
- *CET_L10*: In this color map we see that the overall values of the color map are somewhat dark. This would mean that the color map is always somewhat transparent, which is then impossible to prevent. We excluded 2 color maps like this.
- *CET_I1*: This is an isoluminant color map. The idea is that the perception of the color does not change depending on the diffuse factor on what it is rendered. While even in the original it is hard to distinguish the different colors, with the relatively dark colors it would be very difficult to use in the HoloLens. We excluded 3 color maps like this.

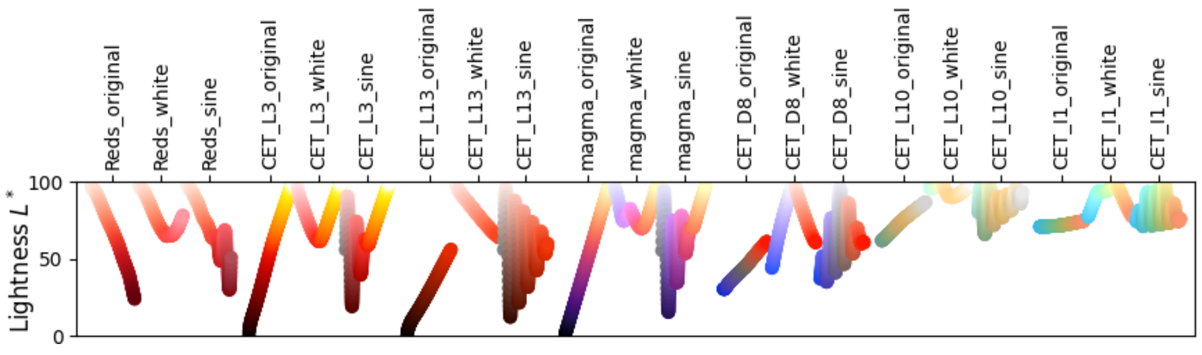

Figure 5.15: Performance of some color maps which we will not use because of transparency issues. For every plot, on the x-axis the color map input value linearly increases. Overlap in the plots is due to limited horizontal space between the plots. The colors in the plots show the mapped color value of the input. For all color maps, there are three plots shown. The first is the original color map with no transparency, the second is where the color map is composited over a white background, the third is where the background is a sine wave from white to black with five periods.

We are now left with 14 color maps that do not fall in a common category to exclude them. In Figure 5.16 we show the lightness plots for these color maps. In this figure we can see a wide variety of color maps and how they look when composited over a white and non-uniform background. There are also some color maps containing little or no dark parts, these will therefore look the same no matter what they are composited over. Note that the color maps with a wide range over the lightness values normally perform best. However, because low lightness values are usually dark colors, these color maps do often not perform well in the HoloLens.

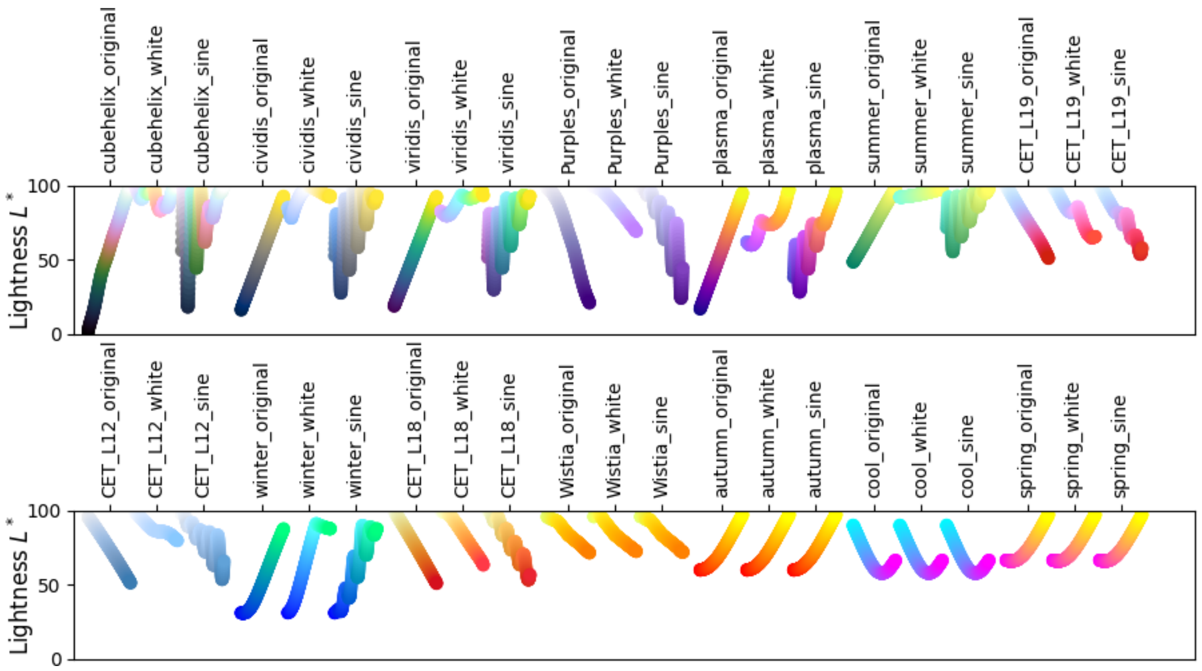

Figure 5.16: Performance of some color maps. For every plot, on the x-axis the color map input value linearly increases. Overlap in the plots is due to limited horizontal space between the plots. The colors in the plots show the mapped color value of the input. For all color maps, there are three plots shown. The first is the original color map with no transparency, the second is where the color map is composited over a white background, the third is where the background is a sine wave from white to black with five periods.

Most color maps start with with a dark color which results in a lot of transparency. Because of this we can rule out the *cubehelix*, *cividis*, *viridis*, and *Purples* color maps. The *plasma* is somewhat better even with dark colors, but it is clearly not linear over a white background. Even though the *summer* color map is not very dark, it is very flat with a white background. We can also rule out the *CET_L19*, *CET_L12*, and *winter* color maps because they are no longer linear over a white background.

We are now left with 5 color maps that should work reasonably well on the HoloLens:

- *CET_L18*: Even though it is somewhat dark, it does not get very transparent, with a minimum alpha of 0.83. Furthermore it has the widest lightness range and is the most linear without transparency. On a white background, it is still linear, although with a smaller lightness range. With a non-uniform background you can clearly see that is is partly transparent, but it is not as bad as many other color maps.

- *Wistia*: This color map has a small lightness range by design. But it also has very little dark sections, with the lowest opacity at 0.95. This means that the color map will be very similar no matter the background.

- *autumn*: Very simple color map, linearly going from red to yellow. Because the red component is always 1, there is no transparency. Is not completely perceptually linear, with in the beginning almost no change in lightness.

- *cool*: Another RGB linear color, from cyan $(0, 1, 1)$ to magenta $(1, 0, 1)$. Has no transparency, since the blue component is always 1. Although it is linear in RGB space, it is clearly not perceptually uniform.

- *spring*: Another RGB linear color, from magenta $(1, 0, 1)$ to yellow $(1, 1, 0)$. Has no transparency, since the red component is always 1. Like the *autumn* color map, not completely linear over the lightness values, with a flat part in the beginning.

In the end we have chosen to use the *Wistia* color map, since this one is the most perceptually linear, even when it is composited over a non-uniform background.

6 Evaluation

In this chapter we describe the user studies that were performed to evaluate the performance of the visualization.

6.1. Experimental setup

User studies were performed to evaluate the performance of the visualization as shown in Section 4.5. The goal is to see if domain experts agree this visualization offers the correct insights in the data.

We are interested in the performance of the two different aspects of the visualization. We want to find out how good the spatial perception of the structures is. And we want to find out if the planning map allows the user to make a more informed decision of where a craniotomy should be performed.

In this experiment with domain experts we asked the participants to perform a task and gave them a questionnaire afterwards. Some time was given to let the participants get comfortable with the program and how to use it. The task that the participants needed to perform is: *Mark the position where you think is the best position and orientation to perform the craniotomy*. Because there are many possible scenarios which can lead to different conclusions we used three different scenarios. Every participant was asked to perform the task for all scenarios in a randomized order. Before the first evaluation we asked a neurosurgeon if these were appropriate scenarios for the evaluation, and if they were realistic.

- Small tumor of $2.4cm$ diameter, at the surface of the skull.
- Large tumor of $8.4cm$ diameter, at the surface of the skull.
- Deep tumor, same scenario as the small tumor, but the tumor was artificially moved deeper, due to lack of a dataset with a deep tumor.

A questionnaire was performed afterwards to evaluate the test, see Appendix A. Besides the questionnaire we also took notes during the sessions to be able to perform some qualitative evaluation.

Some shorter evaluations were also performed, here participants had only around a minute to try the system. They were given the large tumor scenario and no possibility to turn parts of the visualization on or off. Afterwards they were given the same questionnaire, except for the questions about the different scenarios.

A total of ten surgeons from different fields participated in this evaluation, the participants had an average of 11 years of experience. We categorized the participants into three groups: neurosurgeons, long evaluations and all the participants. Two neurosurgeons participated in the evaluation, with an average of 20 years of experience. Six people did the long evaluation, including the neurosurgeons, with an average of 17 years of experience.

Most of the evaluations were somewhat hurried, due to limited time from the clinicians. This left no time to explain the controls of the visualization, instead, we controlled the visualization ourselves and participants had to ask to turn structures on and off.

#	Question	Neuro-surgeons	Long	All
1	It was clear what the different structures are	4.0	4.2	4.1
2	I was able to distinguish structures by their color	4.5	4.3	4.1
3	The shape and size of the structures was clear	4.0	4.2	4.1
4	The spatial position of the structures was clear	4.0	4.0	3.5
5	The distance between the different structures was clear	4.0	4.0	3.4
6	It was clear if structures were behind or in front of other structures	4.0	4.3	4.0
7	The depth of the tumor in the head was clear to see	4.0	4.0	3.9
8	I did not perceive the latency of the system as an issue	3.0	3.2	2.6
9	I did not perceive jittery movement of the models as an issue	3.0	3.2	2.6
10	It felt like the alignment of the models to the head was accuracy enough	3.5	3.7	2.9
11	The visualization did make clear how many structures lay between the craniotomy surface and the tumor	4.0	4.3	3.8
12	The visualization did make clear what specific structures lay between the craniotomy surface and the tumor	3.5	4.1	3.7
13	I was able to easily move the planning map on the skull surface	3.5	3.5	2.8
14	The planning map helped me in finding the right position for the craniotomy	4.5	4.2	3.5
15	Structures hiding behind each other was not a problem	4.0	3.8	3.4
16	The visualization was useful in the small superficial tumor scenario	4.0	3.8	*N.A.*
17	The visualization was useful in the small deep tumor scenario	3.5	4.0	*N.A.*
18	The visualization was useful in the large superficial tumor scenario	4.0	3.8	*N.A.*

Table 6.1: Results from the closed questions of the evaluation.

6.2. Results

In Table 6.1 we list the results of the closed questions from the evaluation.

The biggest problem that was perceived during the evaluation was with the marker tracking. Especially the jitter and the latency of the system was perceived as a large problem.

Participants also mentioned problems with the device, some experienced problems with the small FoV of the display. This was especially a problem in combination with the marker tracking, were the camera was required to focus on a marker, but the user had to focus the display on the phantom to see the models. They also mentioned that the device was too heavy, although this is only a problem when used for long periods of time such as when used intraoperatively.

What people answered to be the most informative part of the visualization was the general 3D visualization of the models. They especially liked that with AR the spatial relations between the tumor and other brain structures was clearly visible.

The benefit of this system compared to other systems according to the participants is that this system shows a 3D visualization which can be viewed from different angles. Another benefit is that the planning created using this method may be faster and more accurate which would lead to less damage to healthy tissue. Another mentioned benefit is that the craniotomy location can be easily drawn on the head using this visualization.

From the two neurosurgeons we got some useful insights in the structures that were used. They mentioned that some structures could be added: the ventricles, cranial nerves, corpus callosom and smaller vessels. However, they mentioned that the neurosurgeons themselves would prefer to do segmentation of the structures, since they know what is important they can then select what to visualize and what not.

During the evaluation almost all clinicians asked to turn the planning map off and only used the regular rendering of the structures to plan the craniotomy. In the last evaluation with a neurosurgeon it was mentioned that the planning map was perceived not useful. The opinion was that it should be more the decision of the surgeon how to operate and what access path to use.

7 Discussion and Future Work

In this chapter we discuss the results and give recommendations for future work. First we discuss the marker tracking and the accuracy of the camera of the HoloLens. Then we discuss the experiment about the color maps. Lastly, we discuss the the visualization and possible improvements that can be made.

7.1. Marker tracking

From the evaluation we find that the marker tracking was the perceived biggest problem. Several improvements can be made to the marker detection which might improve the results. Otherwise, other methods such as described in Subsection 2.2.2 can be used. Another method by Kunz et al. uses the HoloLens to directly track IR-spheres [29], which is also an interesting method to investigate for future work.

7.1.1. Experimental setup

As described in Section 7.2, there is an error in the camera calibration. This means that the accuracy of the marker tracking is dependent on the position of the marker relative to the camera. Therefore, we only use values from a relatively small specific area in the results. While the results now give some insight in the possible accuracy of the marker tracking, it does not accurately describe the actual accuracy of the marker detection used with the HoloLens.

An error in the camera parameters makes it difficult to create an accurate calibration between the HoloLens and the optical tracker. For the calibration only poses from the same specific area are used, which then further increases the uncertainty in the error of the poses outside this area. Future work should make sure that the camera parameters are accurate before performing such an experiment.

We had planned to perform the calibration beforehand, but a synchronization error invalidated our calibration. We therefore used the recorded poses to create the calibration. These same poses were then used to calculate the results, thus introducing a bias in the results.

7.1.2. Noise

We can calculate the theoretical accuracy the marker tracking can achieve. With the resolutions and FoVs of the camera modes we can calculate the spatial resolution at arm's length, $50cm$. ArUco achieves an average corner accuracy of 0.16 pixels with a maximum of around 0.39 pixels [50]. If we take the average corner accuracy and change the position of the individual corners on the x-axis by 0.16 pixels, in the worst possible distribution of the corner error we get a pose estimation error shown in Table 7.1. The error results for the rotation scale exponentially, if we take the maximum pixel error of 0.39 pixels instead, the maximum rotation error for the 1408×792 camera resolution would be $0.60°$.

From the results in Subsection 5.1.2 we see that our setup achieves a standard deviation in the poses which is lower than the theoretical accuracy. But, when comparing the poses of the HoloLens and the optical tracker we see that the noise was not really an issue, since the errors are a lot higher than the noise.

Resolution	Spatial resolution	Maximum rotation error	Maximum translation error
1408×792	$0.32mm$	$0.10°$	$0.43mm$
896×504	$0.50mm$	$0.22°$	$0.64mm$
704×396 (downscaled)	$0.63mm$	$0.40°$	$0.85mm$
448×252 (downscaled)	$0.99mm$	$0.93°$	$1.28mm$

Table 7.1: Spatial resolution of different camera modes at a distance of $50cm$ with FoV of $48 \times 27°$. Maximum error of rotation and translation of pose estimation by changing the position of the corners on the x axis. The corners are moved on the x axis by the average corner error of ArUco, which is 0.16 pixels. For the rotation the worst possible move is when three corners are moved to one side and one corner to the other side. For the translation the worst possible configuration is when the two left corners move to one side, and the two right corners move to the opposite side.

7.1.3. Accuracy

The pose of a marker is defined by the position of the four corners. These corners are found at the intersections found on the edges of a marker, which are found using linear regression of the pixels on the edges. Therefore, we expect that the error mostly depends on the number of available pixels on the edges of the marker. Most of our results seem to confirm this, decreasing the relative size of the marker on the frame or decreasing the resolution decreases the accuracy.

With moving markers there are two additional error sources. The first is motion blur, decreasing the pixel accuracy on the marker edges. The second is that the temporal alignment between the HoloLens and the optical tracker is not perfect, with fast movements this difference causes larger errors.

The effect of motion blur was probably not constant during the experiment. This is due to the motion blur being dependent on the lighting conditions, which was not a controlled factor during the experiment. This makes it impossible to report how much of an influence these factors had on the results.

From the results with varying incident angle, we see that the rotational accuracy is optimal around $40°$. Since with a higher incident angle the number of pixels is again lower we would expect that the accuracy decreases. An explanation is that we combined the rotation on the x and y axes in our results. If the rotation along this axes is the same, the number of pixels on the edges of the marker do not decrease as much as a rotation over a single axis would. It is therefore possible that, coincidentally, this was more often the case around an incident angle of $40°$.

Another possible explanation is that for the lower incident angles the possibility that a wrong solution from the ambiguity problem was chosen is higher. This is because the reflection of the marker pose is similar to the correct solution with a low incident angle. For the lower incident angles it is also harder to detect when such a wrong value has been chosen, since the difference in rotation does not differ too much from the correct solution.

7.1.4. Kalman filter

We found that the Kalman filter decreases the amount of noise in the pose detection. However, it also increases the delay in the detection of the markers. For future work it would be good to perform an analysis of what parameters should be set for the Kalman filter, this would probably make the filter perform better and more accurately represent the tracking while suppressing noise.

Currently the Kalman filter measurements are based on the poses relative to the camera. This means that if the HoloLens moves around the makers, the poses change. Instead, if the spatial map of the HoloLens is utilized to calculate the world-space poses of the markers, the Kalman filter can be applied in this coordinate system, where the markers are often stationary. This way the movement of the markers can be assumed to be zero, which might improve the results.

7.1.5. Combinations of markers

With multiple markers the chance of having an outlier in the marker detection is larger. The effect of this outlier is also noticeable in the resulting combination of markers. Future work should therefore find a way to effectively filter out these outliers to create more consistent tracking of combinations markers.

With the results from the accuracy of individual markers it would be possible to assign a weight to a marker based on the pose. This way when combining the markers, a marker with an expected lower accuracy could have less influence in the combined marker.

It might also be possible to use the relative poses of the markers in the PnP problem. If multiple markers are used the corners are no longer coplanar, resulting in that there is no longer an ambiguity in the solution for this problem. Future work might use such a method, as described by Muñoz-Salinas et al., and see how this improves the detection [41].

7.1.6. Detection rate

Another factor of the marker detection is the detection rate, if a marker can actually be detected. Future work should also look into this, since loss of detection was an issue during the visualization evaluation. Marker identification is based on the inner grid of the marker and is performed on a downscaled version of the frame. Several factors such as motion blur, extensive downscaling and the lighting conditions can influence the detection rate.

7.2. Camera parameters

In Section 5.2 we reported that the visualization error of the display and the camera registration seems smaller with a marker size setting of $59.67mm$. Using a ruler we confirmed that the printed size of the marker was exactly $60mm$, with a measurement error of at most $0.2mm$. We were not able to explain where this error comes from, so we can only assume that this is an error in the camera parameters provided by the HoloLens. The HoloLens documentation mentions that the camera parameters may indeed leave an error[1].

Future work should make sure that the HoloLens camera calibration does not contain errors. This can be done by performing a camera calibration ourselves instead of relying on the provided camera parameters [63].

7.3. Color maps

We have chosen to use the *Wistia* color map for the planning map. Even though this color map did not have a large range of $L*$ values, it was almost always perceptually linear, no matter over what background it was composited.

The HoloLens is not able to show models with full opacity. Future work might also evaluate how this influences the performance of color maps. One would need to figure out what the highest possible opacity of the HoloLens is, then composite the color map with that transparency over a background and evaluate the resulting color maps.

7.4. Visualization

In general, from the results in Section 6.1 it seems that the participants approved the visualization. However, many improvements should still be made be before it can be used in practice.

7.4.1. Experimental setup

While all the scenarios used in the visualization were in fact artificial datasets, this was not perceived as a problem to evaluate it. Both the scenarios with the superficial tumor were aligned to the phantom skull, which did not match the brain structures that were used. The small deep tumor scenario was the same as the small superficial one but with the tumor model moved down, yet, even this seemed not to be a problem.

In our evaluation the skull was used to project the planning map on, however, in a real setting it might be better to project this on the skin, because this is the surface of the patient that the surgeon can actually see during planning. Nevertheless, it might also be useful in this case to show the skull as a model.

For future work effort needs to be made to create in-application controls. This way surgeons can control the system themselves, which would result in a more realistic evaluation of the system.

7.4.2. Structures

From the two neurosurgeons we received some useful insights in the structures that were used. They mentioned that some structures could be added: the ventricles, cranial nerves, corpus callosom and

[1]`https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/locatable-camera#distortion-error`

smaller vessels. For fMRI areas and DTI fibers usually only the structures near the tumor are segmented, other regions are not that relevant. However, they mentioned that the neurosurgeons themselves would prefer to do this segmentation, since they know what is important they can then select what to visualize and what not.

It was also mentioned that it would be useful to categorize the DTI fibers based on their function. This would be an easy addition since the colors of the DTI fibers is currently determined by the fractional anisotropy, however, there was no real basis on why this variable was chosen. Instead, colors can be assigned for different categories of DTI fibers.

For the visualization of the vessels it was found that it was difficult to recognize the deeper vessels. Using MRI slices a surgeon can easily see the general anatomical context of the vessels and recognize them. Without this MRI data this is difficult to do. For future work it might be interesting to look into ways to incorporate the MRI slices in the visualization.

Currently, in our visualization only basic shading methods are used. Future work might also want to look into other shading methods, such as ambient occlusion or showing shadows. This might further improve the depth perception of the structures shown.

7.4.3. Craniotomy planning

From the results we conclude that the visualization helped in planning a craniotomy location in the evaluation. There seems to be no significant difference between the different scenarios used.

From the closed questions it seems that the planning map was perceived useful, however, during the evaluations no participant really used the planning map. Only once it was used to confirm the chosen location that was found using the visualization of the structures.

From discussion with a neurosurgeon, it turned out that there are many other factors that influence the planning of the craniotomy. And in practice this is not only determined by the amount of structures that is between the skull and the tumor. Practical problems such as how the skull can be cut open also play a role. There are also structures such as the cerebral falx which it is not possible to operate through, although for such structures it should be possible to add them to the planning map.

Another problem was with the shape of the planning map, this is not always a simple outline of the tumor. For example, with a small tumor a small craniotomy is not the right solution, since it is not possible to operate through that, instead a more elongated opening is created. And for deep large tumors a smaller craniotomy might still be large enough to operate through [55].

The planning map thus requires ways to change the shape of the craniotomy interactively. This way the planning map might be useful to confirm a chosen craniotomy location. With the current implementation the planning map cannot yet be used to find this optimal craniotomy location, and visualizing the structures instead was preferred to find an access path.

7.4.4. Possible other uses

Both neurosurgeons were also interested in using this technology intraoperatively. Then it would also be possible to see the location and spatial relations of the structures after the craniotomy has been performed. Although it is possible that the brain shifts after the skull has been opened, this shift is usually in the order of a few millimeters [19]. When operating on the tumor it is also possible to see how much of the lesion still has to be removed, although this will probably require continuous updates of the tumor model.

Both neurosurgeons also mentioned that this technology can be used for ventriculostomy procedures. Here internal pressure on the brain is caused by an excess of cerebral fluid, this is released by catheterising the ventricles. Azimi et al. performed a study for this application using a HoloLens [3].

Most participants were not neurosurgeons and many were interested in applying this technology to their field. This should not be difficult to apply with other rigid body parts, but for non-rigid body parts it is difficult to align the model with the patient. Since it would require that the model is continuously updated with the changing shape of the body part.

It was mentioned that this system could also be used for planning before entering the operating room, where a surgeon can use this in an office to get a better 3D perception of the brain structures. However, since it would only be necessary to show the patient model, a see-through display has no apparent benefit. Moreover, with the downsides of AR mentioned in Section 2.2, VR would be better suited here.

8 Conclusions

Outside of the optimal cases, where we can achieve an accuracy of $2mm$, it seems that our marker tracking solution is not yet accurate enough to use in practice. This accuracy will probably improve in the near future with newer devices and higher camera resolutions. Although this does not solve the jitter in the tracking, which future work should focus on improving.

Future work could also look into other ways to track patients and tools. This can be done by using outside-in tracking, using hybrid markers or track other types of markers. Previous studies looked into the accuracy of these different tracking methods, but these tracking methods should be combined with using insightful visualizations.

AR visualization has great potential in neurosurgery and other types of surgery. Many of the participants in our evaluation thought that AR can indeed assist in surgical procedures. Mostly because it better allows to see spatial relations of the anatomical structures and provides more context of the operating site.

Neurosurgery specifically can also benefit from this visualizations, although improvements are still required. More collaboration with neurosurgeons is required, they will eventually use the system and it requires their insights to make the visualization better. The visualization of brain structures can be improved using methods from literature and feedback provided in the evaluation. Controls of the application have to be added to make it possible to use the application in practice.

To make the planning map possibly useful, it should be possible to change the craniotomy shape interactively. Even then it seems that neurosurgeons prefer to look at the brain structures instead to plan a craniotomy. Although with improvements and more adaptation of AR in the field, this might change in the future.

Bibliography

[1] K S Arun, T S Huang, and S D Blostein. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, 1987. ISSN 1939-3539 VO - PAMI-9. doi: 10.1109/TPAMI.1987.4767965.

[2] Ehsan Azimi, Long Qian, Nassir Navab, and Peter Kazanzides. Alignment of the Virtual Scene to the Tracking Space of a Mixed Reality Head-Mounted Display. *arXiv e-prints*, page arXiv:1703.05834, 3 2017. URL http://arxiv.org/abs/1703.05834.

[3] Ehsan Azimi, Zhiyuan Niu, Maia Stiber, Nicholas Greene, Ruby Liu, Camilo Molina, Judy Huang, Chien-Ming Huang, and Peter Kazanzides. An Interactive Mixed Reality Platform for Bedside Surgical Procedures. In Anne L Martel, Purang Abolmaesumi, Danail Stoyanov, Diana Mateus, Maria A Zuluaga, S Kevin Zhou, Daniel Racoceanu, and Leo Joskowicz, editors, *Medical Image Computing and Computer Assisted Intervention*, pages 65–75, Cham, 2020. Springer International Publishing. ISBN 978-3-030-59716-0.

[4] Ronald T. Azuma. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, 8 1997. ISSN 1054-7460. doi: 10.1162/pres.1997.6.4.355. URL http://www.mitpressjournals.org/doi/10.1162/pres.1997.6.4.355.

[5] Takehiko Bando, Atsuhiko Iijima, and Sumio Yano. Visual fatigue caused by stereoscopic images and the search for the requirement to prevent them: A review. *Displays*, 33(2):76–83, 2012. ISSN 0141-9382. doi: https://doi.org/10.1016/j.displa.2011.09.001. URL http://www.sciencedirect.com/science/article/pii/S0141938211000825.

[6] Peter J Basser, James Mattiello, and Denis LeBihan. MR diffusion tensor spectroscopy and imaging. *Biophysical journal*, 66(1):259–267, 1994.

[7] Sylvain Bernhardt, Stéphane A. Nicolau, Luc Soler, and Christophe Doignon. The status of augmented reality in laparoscopic surgery as of 2016. *Medical Image Analysis*, 37:66–90, 4 2017. ISSN 1361-8415. doi: 10.1016/J.MEDIA.2017.01.007. URL https://www.sciencedirect.com/science/article/pii/S1361841517300178.

[8] James F Blinn. Models of Light Reflection for Computer Synthesized Pictures. In *Proceedings of the 4th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '77, page 192–198, New York, NY, USA, 1977. Association for Computing Machinery. ISBN 9781450373555. doi: 10.1145/563858.563893. URL https://doi.org/10.1145/563858.563893.

[9] Andong Cao, Ali Dhanaliwala, Jianbo Shi, Terence Gade, and Brian Park. Image-based marker tracking and registration for intraoperative 3D image-guided interventions using augmented reality. *arXiv e-prints*, page arXiv:1908.03237, 8 2019. URL http://arxiv.org/abs/1908.03237.

[10] Toby Collins and Adrien Bartoli. Infinitesimal plane-based pose estimation. *International journal of computer vision*, 109(3):252–286, 2014.

[11] Fabrizio Cutolo, Antonio Meola, Marina Carbone, Sara Sinceri, Federico Cagnazzo, Ennio Denaro, Nicola Esposito, Mauro Ferrari, and Vincenzo Ferrari. A new head-mounted display-based augmented reality system in neurosurgical oncology: a study on phantom. *Computer Assisted Surgery*, 22(1):39–53, 2017. ISSN 24699322. doi: 10.1080/24699322.2017.1358400. URL https://doi.org/10.1080/24699322.2017.1358400.

[12] Stefan Diepenbrock. Pre-Operative Planning of Brain Tumor Resections. *IEEE Visualization Contest*, 2010.

[13] Belma Dogdas, David W Shattuck, and Richard M Leahy. Segmentation of skull and scalp in 3-D human MRI using mathematical morphology. *Human Brain Mapping*, 26(4):273–285, 12 2005. ISSN 1065-9471. doi: 10.1002/hbm.20159. URL https://doi.org/10.1002/hbm.20159.

[14] S Eichelbaum, M Hlawitschka, and G Scheuermann. LineAO—Improved Three-Dimensional Line Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):433–445, 3 2013. ISSN 1941-0506. doi: 10.1109/TVCG.2012.142.

[15] Taylor Frantz, Bart Jansen, Johnny Duerinck, and Jef Vandemeulebroucke. Augmenting Microsoft's HoloLens with vuforia tracking for neuronavigation. *Healthcare Technology Letters*, 5(5):221–225, 10 2018. ISSN 2053-3713. doi: 10.1049/htl.2018.5079. URL https://digital-library.theiet.org/content/journals/10.1049/htl.2018.5079.

[16] Michael Garland and Paul S Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, 1997.

[17] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 6 2014. ISSN 0031-3203. doi: 10.1016/J.PATCOG.2014.01.005. URL https://www.sciencedirect.com/science/article/pii/S0031320314000235.

[18] S Garrido-Jurado, R Muñoz-Salinas, F J Madrid-Cuevas, and R Medina-Carnicer. Generation of fiducial marker dictionaries using Mixed Integer Linear Programming. *Pattern Recognition*, 51: 481–491, 2016. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2015.09.023. URL http://www.sciencedirect.com/science/article/pii/S0031320315003544.

[19] Ian J Gerard, Marta Kersten-Oertel, Kevin Petrecca, Denis Sirhan, Jeffery A Hall, and D Louis Collins. Brain shift in neuronavigation of brain tumors: A review. *Medical Image Analysis*, 35: 403–420, 2017. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2016.08.007. URL http://www.sciencedirect.com/science/article/pii/S1361841516301566.

[20] Florian Heinrich, Gerd Schmidt, and Christian Hansen. A novel Registration Method for Optical See-Through Augmented Realty Devices and Optical Tracking Data.

[21] Marc Herrlich, Parnian Tavakol, David Black, Dirk Wenig, Christian Rieder, Rainer Malaka, and Ron Kikinis. Instrument-mounted displays for reducing cognitive load during surgical navigation. *International Journal of Computer Assisted Radiology and Surgery*, 12(9):1599–1605, 2017. ISSN 18616429. doi: 10.1007/s11548-017-1540-6.

[22] David M. Hoffman, Ahna R. Girshick, Kurt Akeley, and Martin S. Banks. Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of Vision*, 8(3):33, 3 2008. ISSN 1534-7362. doi: 10.1167/8.3.33. URL http://jov.arvojournals.org/article.aspx?doi=10.1167/8.3.33.

[23] Fatih Incekara, Marion Smits, Clemens Dirven, and Arnaud Vincent. Clinical Feasibility of a Wearable Mixed-Reality Device in Neurosurgery. *World Neurosurgery*, 118:e422–e427, 2018. ISSN 18788769. doi: 10.1016/j.wneu.2018.06.208. URL https://doi.org/10.1016/j.wneu.2018.06.208.

[24] Bernhard Kainz. Multivariate Beam Ray Obstacle Visualization for Brain Tumor Resection. *IEEE Visualization Contest*, 2010.

[25] H C Kam, Y K Yu, and K H Wong. An Improvement on ArUco Marker for Pose Tracking Using Kalman Filter. In *2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 65–69, 2018. ISBN null VO -. doi: 10.1109/SNPD.2018.8441049.

[26] Marta Kersten-Oertel, Sean Jy Shyang Chen, and D. Louis Collins. An evaluation of depth enhancing perceptual cues for vascular volume visualization in neurosurgery. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):391–403, 2014. ISSN 10772626. doi: 10.1109/TVCG.2013.240.

[27] Peter Kovesi. Good Colour Maps: How to Design Them. pages 1–42, 2015. URL http://arxiv.org/abs/1509.03700.

[28] Ivo Kuhlemann, Markus Kleemann, Philipp Jauer, Achim Schweikard, and Floris Ernst. Towards X-ray free endovascular interventions – using HoloLens for on-line holographic visualisation. *Healthcare Technology Letters*, 4(5):184–187, 10 2017. ISSN 2053-3713. doi: 10.1049/htl.2017.0061. URL https://digital-library.theiet.org/content/journals/10.1049/htl.2017.0061.

[29] Christian Kunz, Paulina Maurer, Fabian Kees, Pit Henrich, Christian Marzi, Michal Hlaváč, Max Schneider, and Franziska Mathis-Ullrich. Infrared marker tracking with the HoloLens for neurosurgical interventions. *Current Directions in Biomedical Engineering*, 6(1):20200027, 2020. doi: https://doi.org/10.1515/cdbme-2020-0027. URL https://www.degruyter.com/view/journals/cdbme/6/1/article-20200027.xml.

[30] Yang Liu, Haiwei Dong, Longyu Zhang, and Abdulmotaleb El Saddik. Technical evaluation of HoloLens for multimedia: A first look. *IEEE Multimedia*, 25(4):8–18, 10 2018. ISSN 19410166. doi: 10.1109/MMUL.2018.2873473. URL https://ieeexplore.ieee.org/document/8493267/.

[31] William E Lorensen and Harvey E Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, page 163–169, New York, NY, USA, 1987. Association for Computing Machinery. ISBN 0897912276. doi: 10.1145/37401.37422. URL https://doi.org/10.1145/37401.37422.

[32] M Ronnier Luo, Guihua Cui, and Changjun Li. Uniform colour spaces based on CIECAM02 colour appearance model. *Color Research & Application*, 31(4):320–330, 8 2006. ISSN 0361-2317. doi: 10.1002/col.20227. URL https://doi.org/10.1002/col.20227.

[33] S Manabe, S Ikeda, A Kimura, and F Shibata. Shadow Inducers: Inconspicuous Highlights for Casting Virtual Shadows on OST-HMDs. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1331–1332, 2019. ISBN 2642-5254 VO -. doi: 10.1109/VR.2019.8798049.

[34] F Landis Markley, Yang Cheng, John L Crassidis, and Yaakov Oshman. Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, 2007.

[35] Antonio Martinez-Millana, Jose-Luis Bayo-Monton, Aroa Lizondo, Carlos Fernandez-Llatas, and Vicente Traver. Evaluation of Google Glass Technical Limitations on Their Integration in Medical Systems. *Sensors*, 16(12), 2016. ISSN 1424-8220. doi: 10.3390/s16122142. URL https://www.mdpi.com/1424-8220/16/12/2142.

[36] Majid Maybody, Carsten Stevenson, and Stephen B. Solomon. Overview of Navigation Systems in Image-Guided Interventions. *Techniques in Vascular and Interventional Radiology*, 16(3):136–143, 9 2013. ISSN 1089-2516. doi: 10.1053/J.TVIR.2013.02.008. URL https://www.sciencedirect.com/science/article/pii/S1089251613000292.

[37] D Merhof, M Sonntag, F Enders, C Nimsky, P Hastreiter, and G Greiner. Hybrid Visualization for White Matter Tracts using Triangle Strips and Point Sprites. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1181–1188, 2006. ISSN 1941-0506. doi: 10.1109/TVCG.2006.151.

[38] Jene W. Meulstee, Johan Nijsink, Ruud Schreurs, Luc M. Verhamme, Tong Xi, Hans H. K. Delye, Wilfred A. Borstlap, and Thomas J. J. Maal. Toward Holographic-Guided Surgery. *Surgical Innovation*, 26(1):86–94, 2 2019. ISSN 1553-3506. doi: 10.1177/1553350618799552. URL http://journals.sagepub.com/doi/10.1177/1553350618799552.

[39] Uli Mezger, Claudia Jendrewski, and Michael Bartels. Navigation in surgery. *Langenbeck's Archives of Surgery*, 398(4):501–514, 2013. ISSN 14352443. doi: 10.1007/s00423-013-1059-4.

[40] Simon Moosburner, Christopher Remde, Peter Tang, Moritz Queisner, Nils Haep, Johann Pratschke, and Igor M. Sauer. Real world usability analysis of two augmented reality headsets in visceral surgery. *Artificial Organs*, 43(7):694–698, 7 2019. ISSN 0160-564X. doi: 10.1111/aor.13396. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/aor.13396.

[41] Rafael Muñoz-Salinas, Manuel J Marín-Jimenez, Enrique Yeguas-Bolivar, and R Medina-Carnicer. Mapping and localization from planar markers. *Pattern Recognition*, 73:158–171, 2018. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2017.08.010. URL http://www.sciencedirect.com/science/article/pii/S0031320317303151.

[42] Dominik Ospelt, Annette Hausd, and Stefan M. An Exploration and Planning Tool for Neurosurgical Interventions. *IEEE Visualization Contest*, 2010.

[43] Sinisa Pajevic and Carlo Pierpaoli. Color schemes to represent the orientation of anisotropic tissues from diffusion tensor data: Application to white matter fiber tract mapping in the human brain. *Magnetic Resonance in Medicine*, 42(3):526–540, 9 1999. ISSN 0740-3194. doi: 10.1002/(SICI)1522-2594(199909)42:3<526::AID-MRM15>3.0.CO;2-J. URL https://doi.org/10.1002/(SICI)1522-2594(199909)42:3%3C526::AID-MRM15%3E3.0.COhttp://2-j.

[44] THJM Peeters, A Vilanova, and R ter Haar Romeny. Visualization of DTI fibers using hair-rendering techniques. In *Proc ASCI*, pages 66–73, 2006.

[45] Thomas Porter and Tom Duff. Compositing Digital Images. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, page 253–259, New York, NY, USA, 1984. Association for Computing Machinery. ISBN 0897911385. doi: 10.1145/800031.808606. URL https://doi.org/10.1145/800031.808606.

[46] Long Qian, Alexander Barthel, Alex Johnson, Greg Osgood, Peter Kazanzides, Nassir Navab, and Bernhard Fuerst. Comparison of optical see-through head-mounted displays for surgical interventions with object-anchored 2D-display. *International Journal of Computer Assisted Radiology and Surgery*, 12(6):901–910, 6 2017. ISSN 1861-6410. doi: 10.1007/s11548-017-1564-y. URL http://link.springer.com/10.1007/s11548-017-1564-y.

[47] Emily Rae, Andras Lasso, Matthew S. Holden, Evelyn Morin, Ron Levy, and Gabor Fichtinger. Neurosurgical burr hole placement using the Microsoft HoloLens. In Robert J. Webster and Baowei Fei, editors, *Medical Imaging 2018: Image-Guided Procedures, Robotic Interventions, and Modeling*, volume 10576, page 20. SPIE, 3 2018. ISBN 9781510616417. doi: 10.1117/12.2293680. URL https://www.spiedigitallibrary.org/conference-proceedings-of-spie/10576/2293680/Neurosurgical-burr-hole-placement-using-the-Microsoft-HoloLens/10.1117/12.2293680.full.

[48] Jannick P. Rolland and Henry Fuchs. Optical Versus Video See-Through Head-Mounted Displays in Medical Visualization. *Presence: Teleoperators and Virtual Environments*, 9(3):287–309, 6 2000. ISSN 1054-7460. doi: 10.1162/105474600566808. URL http://www.mitpressjournals.org/doi/10.1162/105474600566808.

[49] Jannick P Rolland, William Gibson, and Dan Ariely. Towards Quantifying Depth and Size Perception in Virtual Environments. *Presence: Teleoperators and Virtual Environments*, 4(1):24–49, 1 1995. doi: 10.1162/pres.1995.4.1.24. URL https://doi.org/10.1162/pres.1995.4.1.24.

[50] Francisco J Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. Speeded up detection of squared fiducial markers. *Image and Vision Computing*, 76:38–47, 2018. ISSN 0262-8856. doi: https://doi.org/10.1016/j.imavis.2018.05.004. URL http://www.sciencedirect.com/science/article/pii/S0262885618300799.

[51] Marc Schirski, Torsten Kuhlen, Martin Hopp, Philipp Adomeit, Stefan Pischinger, and Christian Bischof. Efficient Visualization of Large Amounts of Particle Trajectories in Virtual Environments Using Virtual Tubelets. In *Proceedings of the 2004 ACM SIGGRAPH International Conference on Virtual Reality Continuum and Its Applications in Industry*, VRCAI '04, page 141–147, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138849. doi: 10.1145/1044588.1044615. URL https://doi.org/10.1145/1044588.1044615.

[52] Mathias Schluter, Olaf Konrad-Verse, Horst Karl Hahn, Bram Stieltjes, Jan Rexilius, and Heinz-Otto Peitgen. White matter lesion phantom for diffusion tensor data and its application to the assessment of fiber tracking. In Amir A Amini and Armando Manduca, editors, *Medical Imaging 2005: Physiology, Function, and Structure from Medical Images*, volume 5746, pages 835–844. International Society for Optics and Photonics, SPIE, 2005. doi: 10.1117/12.596197. URL https://doi.org/10.1117/12.596197.

[53] Karam Shaya, Aaron Mavrinac, Jose Luis Alarcon Herrera, and Xiang Chen. A Self-localization System with Global Error Reduction and Online Map-Building Capabilities BT - Intelligent Robotics and Applications. pages 13–22, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-33503-7.

[54] Marco Solbiati, Katia M Passera, Alessandro Rotilio, Francesco Oliva, Ilaria Marre, S Nahum Goldberg, Tiziana Ierace, and Luigi Solbiati. Augmented reality for interventional oncology: proof-of-concept study of a novel high-end guidance system platform. *European radiology experimental*, 2:18, 12 2018. ISSN 2509-9280. doi: 10.1186/s41747-018-0054-5. URL http://www.ncbi.nlm.nih.gov/pubmed/30148251http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC6092730.

[55] Axel T Stadie, Ralf A Kockro, Luis Serra, Gerrit Fischer, Eike Schwandt, Peter Grunert, and Robert Reisch. Neurosurgical craniotomy localization using a virtual reality planning system versus intraoperative image–guided navigation. *International Journal of Computer Assisted Radiology and Surgery*, 6(5):565–572, 2011. ISSN 1861-6429. doi: 10.1007/s11548-010-0529-1. URL https://doi.org/10.1007/s11548-010-0529-1.

[56] Satoshi Suzuki and KeiichiA Be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985. ISSN 0734-189X. doi: https://doi.org/10.1016/0734-189X(85)90016-7. URL http://www.sciencedirect.com/science/article/pii/0734189X85900167.

[57] Olivier Vaillancourt, Gabriel Girard, Arnaud Boré, and Maxime Descoteaux. A Fiber Navigator for Neurosurgical Planning (NeuroPlanningNavigator). *IEEE Visualization Contest*, 2010.

[58] Reid Vassallo, Adam Rankin, Elvis C. S. Chen, and Terry M. Peters. Hologram stability evaluation for Microsoft HoloLens. In Matthew A. Kupinski and Robert M. Nishikawa, editors, *Medical Imaging 2017: Image Perception, Observer Performance, and Technology Assessment*, volume 10136, page 1013614. International Society for Optics and Photonics, 3 2017. doi: 10.1117/12.2255831. URL http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2255831.

[59] Balázs Vigh, Steffen Müller, Oliver Ristow, Herbert Deppe, Stuart Holdstock, Jürgen Den Hollander, Nassir Navab, Timm Steiner, and Bettina Hohlweg-Majert. The use of a head-mounted display in oral implantology: A feasibility study. *International Journal of Computer Assisted Radiology and Surgery*, 9(1):71–78, 1 2014. ISSN 18616429. doi: 10.1007/s11548-013-0912-9. URL http://link.springer.com/10.1007/s11548-013-0912-9.

[60] A Vilanova, S Zhang, G Kindlmann, and D Laidlaw. An Introduction to Visualization of Diffusion Tensor Imaging and Its Applications. In Joachim Weickert and Hans Hagen, editors, *Visualization and Processing of Tensor Fields*, pages 121–153. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-31272-7. doi: 10.1007/3-540-31272-2{_}7. URL https://doi.org/10.1007/3-540-31272-2_7.

[61] Colin Ware and Ravin Balakrishnan. Reaching for Objects in VR Displays: Lag and Frame Rate. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 1(4):331–356, 1994. ISSN 15577325. doi: 10.1145/198425.198426.

[62] Song Zhang, Cagatay Demiralp, and David H Laidlaw. Visualizing diffusion tensor MR images using streamtubes and streamsurfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):454–462, 2003.

[63] Z Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000. ISSN 1939-3539 VO - 22. doi: 10.1109/34.888718.

[64] Lin Zheng, Yubo Zhang, and Kwan-liu Ma. Quantitative Visualization of Access Paths for Preoperative Planning in Neurosurgery. *IEEE Visualization Contest*, 2010.

A Questionnaire

A.1. Introduction

This experiment is to evaluate a visualization using an augmented reality headset for assisting in a craniotomy planning. The visualization consists of two parts: The first part is the visualization of some brain structures, the second part is a planning map which aggregates these same structures into a color projected on top of the skull. The brain structures used in this visualization are:

- Skull surface
- Tumor surface
- Some DTI fibers are selected and shown
- Arteries and veins segmented from MRI with contrast agent
- Some brain areas based on a finger tapping procedure in an fMRI scan

The idea of the planning map is to show how safe it is to perform a craniotomy from a particular orientation. This is shown using a color map, where yellow means few structures between the planning map and the tumor and orange means a lot of structures in between. The orientation of the planning map can be changed by moving around the patient, the planning map follows the user and is therefore always directly in front of the tumor, the location can however be locked to be able to look from another angle. The values of the planning map are based on the DTI fibers, vessels and shown brain areas.

The visualization will be shown and we ask to perform the following task:
Lock the planning map at the position where you think would be best to perform the craniotomy for tumor resection.

We ask to perform this task for three scenarios:

- A small tumor at the surface of the skull.
- A large tumor at the skull surface.
- A small tumor deeper in the brain.

Before the sessions time will be given to get comfortable with the system.

In this experiment we will record the following data, all data will be anonymized:

- The resulting locations of the task
- The time to perform the task
- The answers to the questionnaire
- The process of completing the task is recorded on the HoloLens

A.2. Background

1. **What is your profession?**

2. **How many years of experience do you have?**

A.3. Closed questions

1. **It was clear what the different structures are**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

2. **I was able to distinguish structures by their color**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

3. **The shape and size of the structures was clear**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

4. **The spatial position of the structures was clear**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

5. **The distance between the different structures was clear**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

6. **It was clear if structures were behind or in front of other structures**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

7. **The depth of the tumor in the head was clear to see**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

8. **I did not perceive the latency of the system as an issue**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

9. **I did not perceive jittery movement of the models as an issue**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

10. **It felt like the alignment of the models to the head was accuracy enough**

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

11. The visualization did make clear how many structures lay between the craniotomy surface and the tumor

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

12. The visualization did make clear what specific structures lay between the craniotomy surface and the tumor

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

13. I was able to easily move the planning map on the skull surface

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

14. The planning map helped me in finding the right position for the craniotomy

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

15. Structures hiding behind each other was not a problem

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

16. The visualization was useful in the small superficial tumor scenario, please also note why

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

17. The visualization was useful in the small deep tumor scenario, please also note why

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

18. The visualization was useful in the large superficial tumor scenario, please also note why

☐ Strongly disagree ☐ Disagree ☐ Neutral ☐ Agree ☐ Strongly agree

A.4. Open questions

1. What part of the visualization was the most informative?

2. Are the structures used in the planning map (DTI fibers, vessel structures and certain brain areas) sufficient to determine the position of the craniotomy? If not, which structures should be added?

3. What could be improved about the visualization?

4. **What could be the benefit of this method compared to other methods?**

5. **Do you see other applications for this system?**

6. **Do you have any other remarks?**