ON THE INFLUENCE OF STEREOGRAPHIC 3D ON USER DEPENDENT DIRECT PROJECTED AUGMENTED REALITY IN THE OR

UNIVERSITY OF TWENTE

MASTER THESIS

Report Number:
EWI/SAS 2012-002
Author:
S.G. Epskamp (0088617)
Thesis Committee:
Dr. Ir. F. van der Heijden
Prof. Dr. T. Ruers
Dr. M. Poel
Prof. Dr. Ir. C. H. Slump

January 20, 2012

ABSTRACT

This report is the result of a master thesis assignment at the Signals and Signals group, University of Twente. The project is focused on easing the task surgeons face in the path planning phase of an operation, by directly projecting internal structures on a patient's body, using direct projected augmented reality.

Augmented reality is the technique of adding information or virtual objects to real-life environments. This technique can be applied to ease the task surgeons face when planning operations. The goal in path planning is to minimize tissue damage done by incisions, by determining the shortest safe path to the target tissue. In the current workflow this task is frustrated by the fact that the surgeon has to integrate the information he obtains from images shown on separate monitors into his view of the patient. Direct projected augmented reality could ease this task by projecting the structures directly onto the skin of the patient. This should give the surgeon a sort of 'x-ray' vision in which he can directly see the position and orientation of the target structure in the patient. This could greatly increase his accuracy and speed in this planning task. This scenario is the subject of this thesis. A hardware rig was built to augment a Styrofoam mannequin with images of a 3D model of a tumor. This rig was used to test the influence of stereoscopic 3D and the display of a guidance grid emphasizing the patient's body in a simple path planning task.

The hardware rig consists of a projector suspended above a table on which the mannequin lies. The user position is measured using an electromagnetic tracker. The setup is driven by a standard PC. The real-time images showing a geometrically undistorted image of the tumor to the user are generated with the use of OpenGL API.

Through user testing, we find out that both the use of stereoscopic 3D and the display of a guidance grid increases the accuracy of users in the path planning task. Stereoscopy improves user speed as well, while the grid doesn't.

I would like to thank:

My supervisor, Ferdi van der Heijden, coming up with the crucial ideas for this project, and his guidance in the completion of this Thesis.

Geert Jan Laanstra, for his initiative and enthusiasm in building the perfect hardware rig.

My parents, for their continued support during my whole life.

My girlfriend, for her support during the process of creating this thesis.

TABLE OF CONTENTS

1 INTRODUCTION

For women, breast cancer is the most common cancer. About 11% of the Dutch female population will suffer from this disease during their life time. Once the disease has reached a certain stage, it is only treatable with an operation that removes the entire tumor at once. Removal of the tumor must be done with great care and accuracy, since leaving even a tiny bit will eventually lead to a reoccurrence. The task is a difficult one because distinguishing the tumor from surrounding tissues is problematic since both have a spongy texture and similar color.

In the current workflow, the knowledge available on the shape and location of the tumor is not used optimally. Usually a small marker attached to a wire is inserted into the tumor in a pre-operation procedure, CT or MRI scans are used to guide this insertion. It is essential that the shortest possible path to the tumor is chosen, to ensure minimal tissue damage. During the operation, the surgeon follows the wire to find the tumor, and relies on his experience and the medical images shown on separate monitors to remove the complete tumor. His focus is on removing all of it, while minimizing damage to surrounding tissues. It is essential that the surgeon can correctly judge the volume, shape and location of the tumor to perform this task optimally.

The tools available during path planning and operation mainly consist of the display of medical images on separate monitors; the surgeon will have to mentally combine the displayed images with the reality of the patient lying on the table. Physically integrating these modalities could offer advantages. This thesis explores a solution to this problem by applying augmented reality (AR) techniques, in which the patient is augmented with the medical images acquired in earlier stages to ease the mental task the surgeon faces.

Augmented reality is a broad term that describes many techniques in which computers are used to add information to our everyday world. Examples are:

- The projection of information on height, speed and fuel level into a fighter jet pilot's view.
- Mobile phones that can augment live video from their camera with housing prices in the neighborhood
- Projecting a clock onto the wall.

In this thesis, augmented reality is defined as follows:

"Augmented Reality is the name for any technique in which (images of) the direct surroundings are augmented with localized information on these surroundings."

Augmented reality is applied to the case described above by projecting a rendering of a 3D model of a tumor directly onto the patient's body. By measuring and correcting for the position of the surgeon's eyes a sort of "x-ray vision" effect is obtained, in which the surgeon can look straight into the patient's body, and see the embedded tumor. Stereoscopic 3D, in which each eye is supplied with a different image, will further improve the surgeon's ability to judge the volume, shape and location of the tumor.

The first goal of this project is to create a demonstrator that shows the principle of augmented virtual reality in the OR. This involves creating a working, real-time system that is able to generate and project images of a 3D model of a tumor on the mannequin representing the patient in such a way that the tumor's 3D geometry appears undistorted to the user. The user should retain this correct image,

regardless of his/her position in relation to the patient. Furthermore, the system must support the display of the 3D model in stereoscopic 3D. Finally the system should be able to project a grid that emphasizes the surface of the patient.

Compared to a final, real situation in the operating room many requirements and conditions are relaxed. Firstly, only a single projector is used, even though this causes some areas of the patient to be occluded, and is it possible for the user to block the projection. Furthermore, the patient is represented by a styrofoam mannequin. This is a long shot from a real human in terms of deformability. The mannequin is completely static, while a real patient's body will be deformed in comparison to the medical images, under the influence of pressure, the movement of limbs, breathing, etc. Incisions made in the body during operation will cause further deviation from the 3D data recorded pre-operatively. Finally, not much attention was paid to the minimization of errors. This project must be seen as a proof of principle, not the final solution to building a system that is ready for deployment in real operating rooms.

The second goal is in testing the effect that several parameters have in the accuracy of performing a path planning task with the demonstrator. Users are tested on their ability to discover the closest point on the mannequin to a virtual tumor below the mannequin's surface. The parameters tested involve the tumor's rendering, to be specific:

- The effect of stereoscopic 3D
- The display a grid that emphasizes the mannequin's surface.
- The effect of the tumors shape, either realistic or designed to maximize the stereoscopic effect.

To realize the goals mentioned above, a hardware platform is realized that allow users to walk around a mannequin, viewing it from multiple angles. A projector suspended above the mannequin allows projecting images on it, an electromagnetic tracker is used to track the users' head and allow him/her to point out positions on the mannequin's surface. The setup is driven by a regular PC. This hardware platform is described in chapter 2: "The hardware rig". To control the setup a software program was written in Java. To generate the desired images in real-time, the OpenGL API was utilized. The mathematical model of the software program and its implementation are detailed in chapter 4: "Model" and chapter 5: "Software implementation".

A user study was conducted with this demonstrator. 19 Users were shown a virtual tumor in several places throughout the body, and were asked to indicate the position on the mannequin's surface closest to the tumor. The parameters mentioned above were varied throughout the tests, as to gain insight into the role they play in the users' performance. Users were also questioned on their preferred rendering method, to gain some qualitative insights. The exact setup of the user study, its statistical analysis with mathematical model, and a discussion of the results is presented in chapter 6: "Usability testing".

2 PRIOR RESEARCH

The field of augmented reality is an active field of research in user interaction groups around the world, and has been so for over 40 years [1]. Augmented reality has found applications in entertainment, manufacturing, visualization, path planning, and in medical and military applications. A wide variety of experimental setups were created. The systems described below are mostly picked for their focus on medical situations.

1990 - AACHEN UNIVERSITY OF TECHNOLOGY

At Aachen University, a module was developed that assists in head and neck surgical procedures. CT scans of the patient's skull are registered to radiopaque markers. The module is calibrated using a mechanical digitizer. The system generates images of the current position of the digitizer in relation to the skull in three perpendicular views which are shown on a nearby monitor. The system has to be recalibrated every time the patient moves. [2]

1993 - GENERAL ELECTRIC.

At General Electric in 1993, Lorensen et. al built and tested a system that mixed computer generated images with a live video feed of an operation. The 3D model used was built by applying a marching cubes algorithm to segmented CT data. The resulting images were aligned manually to the image of the patient, and mixed with the video feed by a technician. The result was shown on a monitor in the OR. [3]

1994 - MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Grimson et. al describe a method for registering these 3D datasets to data obtained from the patient in the OR. [4] In this way, knowledge is gained on the exact position of the patient in the OR. Laser striping is used to acquire a depth map of the patient's skin. This data is registered to segmented mri scans using 3D/3D surface matching. The calibration must be reperformed if the patient or camera move and the overlay can only be generated for a single viewpoint.

1995 - MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Mellor expanded on [4] by creating a system that can register 3d models to camera footage in real time, using markers. The location and orientation of a skull is inferred as a test. An initial calibration must be performed using a laser scanner. After the calibration is performed, the method is fully automatic, runs in nearly real-time and is accurate to a fraction of a pixel. [5]

1999 - UNIVERSITY OF NORTH CAROLINA

In a technique called dubbed *Shader Lamps*, Raskar et. al created a system that uses projectors to give real-world objects a different 'skin', or make it seem like they are in motion. The system uses manually aligned physical models. The projections intensity is corrected based on surfaces slope and corrected for real-world secondary scattering. [6]

2003 - UNIVERSITY OF MONTRÉAL

At Montréal University Tardif et. al developed a system that uses a direct pixel mapping from a camera to a projector to display undistorted image data on a non-planar surface. The method uses structured light to create this mapping. [7]

2006 - HANYANG UNIVERSITY

A group at Hanyang University has developed a system that can generate geometrically undistorted projections on a non-planar surface. The image is shown using multiple projectors, which are blended

together and whose intensity is corrected based on the slope of the display surface. It isn't mentioned if the system is real-time or was tested with users. [8]

2010 - UNIVERSITY OF HAGEN

The University of Hagen built a system that tracks the user using the Kinect depth sensor. A flat table is used as the display surface for a projector. The projection gives the illusion that an object is standing on a recessed plane in the table. The setup suffers from display lag, caused by processing the depth image. [9]

Concluding, we see that the developed technologies for use in the OR become increasingly adaptive and automatic. Manual calibration becomes increasingly unnecessary with the advent of more advanced sensors and registration techniques. Steps are being taken to support registration even without markers. While early systems just show the position of tracked tools in relation to certain markers, the patient is increasingly visualized as well. On separate monitors at first, but later on in integrated forms. The display technologies in the medical settings consist of monitors and heads-up-displays, while experimentations with projection are ongoing mostly outside of the medical field. The system described in this report is fairly original. Not a single paper could be found that described real-time projected augmented reality on non-planar surfaces for use in a medical setting.

3 THE HARDWARE RIG

A hardware rig was built as the platform for the demonstrator to run on. The platform consists of the following parts: (see figure 3-1)

- A mannequin to represent the patient. The mannequin has to be as close to a real human body as possible, in an effort to mimic the situation in during operation more closely. On the other hand the mannequin has to be completely static so that a 3D model of its surface only has to be obtained once. Furthermore the mannequin's surface has to be suitable for projecting images on.
- A projector to project the tumor images. The projector has to be able to focus on a surface only ~1.5 m away which is rare in consumer models. Furthermore its image mustn't shift while the bulb is warming up, to prevent recalibration becoming necessary over time. Furthermore the projector must cause a minimum of interference to the tracking solution discussed next.
- A tracking solution to track the user's head, and allow him/her to point out positions on the mannequin's surface. The tracker has to offer maximum precision and minimal latency. High latency will degrade the user experience, as the projected images will visibly lag behind while moving around the mannequin. Low precision will degrade both the acquired data and user experience. If will cause saved positions to differ from actuality, and cause the projected image to be incorrect.
- A standard desktop computer containing a 3D accelerator card was used to drive the rig.

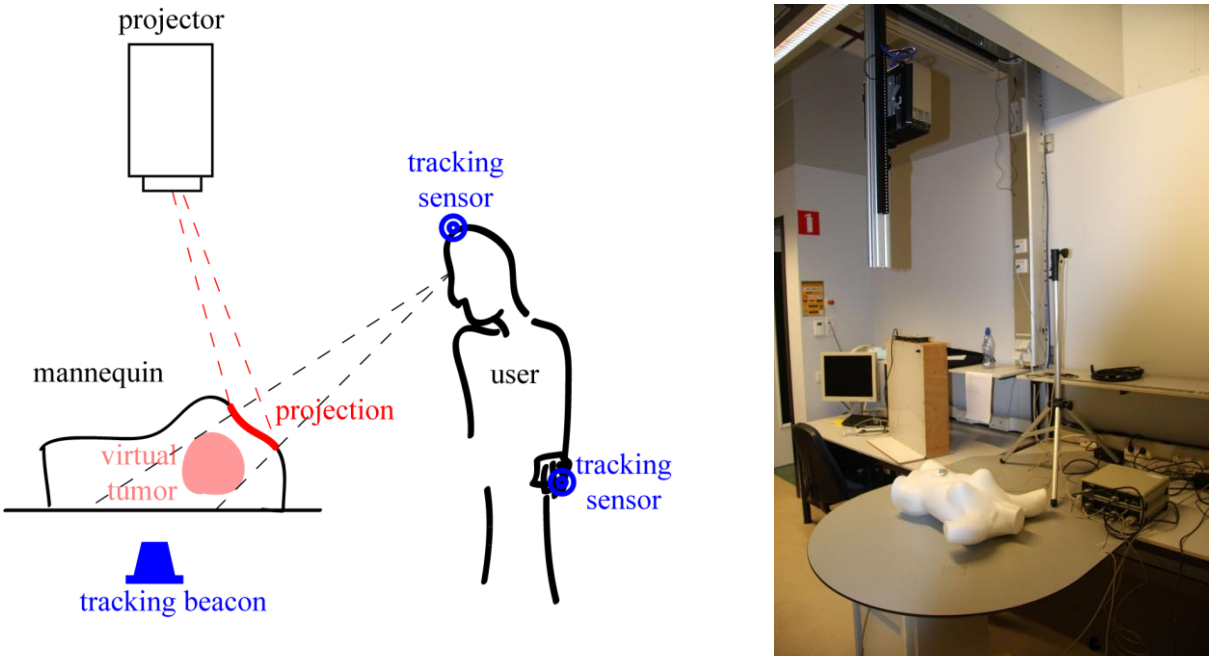


figure 3-1 - Hardware rig, schematic and in practice.

As a mannequin a styrofoam model is used. (See figure 3-2) The model is chosen to ensure a projection surface that approximates a human body in shape and form. It's even white surface is suitable for projection, unlike other mannequins that feature colored cloth. The shape is completely static and the surface doesn't dent easily, as opposed to several foam models. In this way possible deformations don't have to be accounted for in the model and software.

figure 3-2
The mannequin

figure 3-3
The several parts of the electromagnetic tracker. From top to bottom: Processing unit, beacon and tracking sensor.

To track the user's eyes position and to allow the user to indicate positions on the mannequin's surface, a Flock of Birds electromagnetic tracker by Ascension is used (see figure 3-3). It operates by generating strong magnetic fields in a beacon, that are picked up by one or more sensors. Both the beacon and the sensors contain three perpendicular wire coils. A current is passed in order through the coils in the beacon, to generate magnetic fields of varying orientation. These fields in turn generate a current in the coils in the sensors. By measuring these currents, an indication for the position and orientation of the sensor with regards to the beacon can be found. [10] Because of the use of magnetic fields, the proximity of large metal objects or electronics can interfere with these measurements. The hardware rig tries to minimize this interference by using a metal less table, with the beacon directly underneath. Al the needed electronics are placed as far as possible from the beacon and sensors.

This has a precision of 1.4 mm. Latency can be as low as 6 milliseconds, but increased with the amount of filtering that is applied. An advantage of electromagnetic tracking is that it supplies relatively good accuracy, especially when the tracking sensors are close to the beacon. The beacon was placed directly under the mannequin for this reason. The accuracy is a distinct advantage over the optical trackers that were considered as an alternative. These generally have to be placed far away to completely cover the scene, but accuracy decreases at these distances. The only available depth sensor, the Microsoft Kinect, also offers starkly lower accuracy. Furthermore it does not suffer from the line-of-sight problems of optical trackers. The latency is potentially higher than optical trackers, but in testing it turned out to be low as to not seriously degrade the user experience. This tracker supports multiple wired sensors, making it very easy to track both the users head and his/her hand. Finally, the flock of birds system has an additional sensor that can measure and compensate for a source of electromagnetic interference. We use it to compensate for the projector, since it is so close to the tracking sensor on the user's head.

The projector used is an Optoma EP719 (see figure 3-4). It is based on DMD (Digital Micro mirror Device) technology. DMD devices project different intensities by changing the number of times the source light is reflected into the lens instead of into a heat sink. The higher the number of times the light is reflected into the lens, the higher the resulting intensity of that particular pixel. The absence of a cathode ray tube is an advantage as it lessens the electromagnetic interference for the tracker. The projector supports focusing on a surface as close as 1 meter. Testing showed no significant amount of image shift during warm-up, making this particular projector suitable for our intended usage.

figure 3-4 The chosen projector.

The suspension height is a compromise between several conflicting requirements. On the one hand, the closer the projector is to the display surface, the smaller the area that the projector covers, hence the higher the resolution in this area. On the other hand, a maximum amount of surface area on the mannequin should be covered.

To drive the setup a, by current standards, high-end PC was used. It contains a NVidia GTX 295 3D accelerator card. To generate real-time 3D images, a 3D accelerator is almost a necessity. The central processing unit (CPU) is an Intel i7-920, which contains four physical cores. Since the software is mostly single threaded, and most of the work is done on the 3D accelerator anyway, no strong requirements are placed on the CPU. The other components are not described here, since they are mostly irrelevant to the performance of the system.

4 MODEL

The software in the demonstrator faces the task of generating images of a 3D model of a tumor that make the tumor's 3D geometry appear undistorted to the user when projected on the mannequin. In this chapter, this problem is described in mathematical terms. We start of out with an introduction to our model of 3d space, and how 3d surfaces are defined in it. This is followed by a description of the projector's model. Next the problem of generating the images is defined in these terms. Finally the radiometric model used in rendering the tumor is described. The implementation of this model is described in chapter 5.

4.1 GEOMETRIC MODEL

To define the relationship between 3D models in our system it is important to choose a coordinate system and the origin. In our system, one of the corners of the projector suspension rig is chosen as the origin. The chosen coordinate system is equal to the OpenGL coordinate system, shown in figure 4-1. In this system the axis labeled z points towards the user, out of the screen, y points up and x to the right.

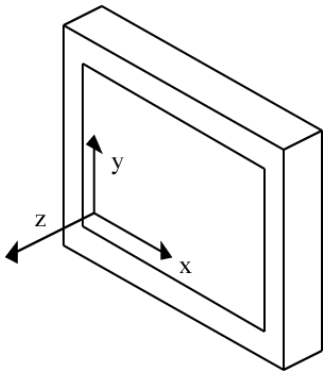


figure 4-1 – The OpenGL coordinate system visualized.

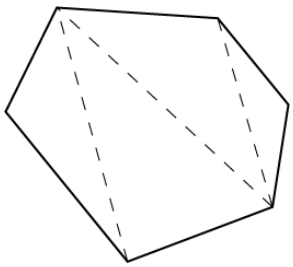


figure 4-2 – A polygon, with triangulation.

A surface is not modeled by a continuous function in our system, but by polygons. A polygon is a flat surface surrounded by line segments, called edges. The corners of these line segments are called vertices. In our system, only polygons with exactly three vertices, i.e. triangles, are allowed. For every vertex in the system the normal is defined. The normal is the vector that is perpendicular to the *represented* surface at that vertex location. Since several triangles meet at most vertices, there are several different vectors to perpendicular their respective surfaces. There is however only 1 vector perpendicular to the continuous surface we are modeling with this representation. Every normal in the system is *normalized*, that is, it has unit length. By giving every vertex 2D coordinates a mapping from

3D to 2D space is created. Such a mapping is called a UV mapping, and its coordinates are called UV coordinates. UV coordinates can be used to map a 2D image onto a polygon.

4.1.1 PROJECTOR

The beamer in the system is modeled using the pinhole camera model (see figure 4-3). This model can be seen as a very basic type of camera that passes light through a small hole instead of a lens. Beams of light are directly projected onto the camera's image plane. The relation between the location of a point in the world and its projection on the image plane is therefore linear if expressed in homogeneous coordinates. It may seem strange that a camera model is used to model a projector, but their optical principles are the same. Compared with a camera, light in the projector just travels in the opposite direction and is generated there instead of captured.

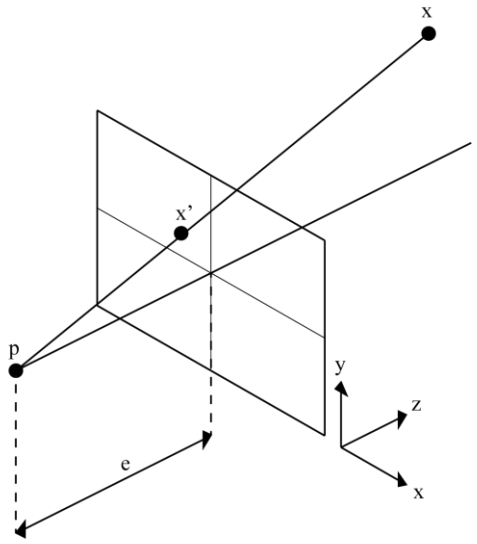


figure 4-3 – Pinhole camera model used to model the projector.

We define the following variables

$\mathbf{x}^W$ $-$ *the point* **x** *in world coordinates*
$\mathbf{x}^P$ $-$ *the point* **x** *in projector coordinates*
$\mathbf{x}'$ $-$ *a projection of the point* **x**
$\mathbf{p}^W$ $-$ *the projector position in world coordinates*
$\mathbf{P}$ $-$ *the projection matrix*
$\boldsymbol{\theta}_{\alpha,\beta,\gamma}$ $-$ *the projector's rotation in world coordinates, expressed in Euler angles*

All coordinates are homogenous.

To calculate the location where a 3D coordinate $\mathbf{x}^W$ is projected on the image plane *pp* we multiply with the projection matrix**P** and end up with intermediate results **b** (see equation 4-1).

$$\mathbf{b} = \mathbf{P}\mathbf{x}^W \qquad \textbf{equation 4-1}$$

b Is a homogenous 3D coordinate, $\mathbf{x}' = \frac{b_{xy}}{b_w}$ represents the 2D coordinates on the image plane. The projection matrix **P** is defined in equation 4-2.

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/e & 0 \end{bmatrix}$$ **equation 4-2**

In this equation, e represents the focal distance, i.e. the distance between the projector position **p** and the image plane. If the area of the image plane is limited, there is an inverse linear relation between e and the field of view that is modeled by the projection matrix **P**. A smaller value of e leads to a larger field of view.

The calculations above assume all points are defined relative to **p**. In other words, they assume the projector is located at the origin, with no applied rotation **θ**. If this isn't the case, we have to translate a point **x** to this projector coordinate system. To achieve this, the location of the projector is first subtracted from the point, after which it is rotated by $-\boldsymbol{\theta}$.

Using the above, it is possible to define a single matrix, that, when multiplied with a point, translates it to projector coordinates, and projects it onto the image plane. We call this matrix **C**. (see equation 4-3)

$$\mathbf{x}' = \mathbf{C}\mathbf{x}^W$$ **equation 4-3**

C is defined in equation 4-4.

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\boldsymbol{\theta}_x) & -\sin(\boldsymbol{\theta}_x) & 0 \\ 0 & \sin(\boldsymbol{\theta}_x) & \cos(\boldsymbol{\theta}_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\boldsymbol{\theta}_y) & 0 & \sin(\boldsymbol{\theta}_y) & \\ 0 & 1 & 0 & \\ -\sin(\boldsymbol{\theta}_y) & 0 & \cos(\boldsymbol{\theta}_y) & \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot$$
$$\begin{bmatrix} \cos(\boldsymbol{\theta}_z) & -\sin(\boldsymbol{\theta}_z) & 0 & 0 \\ \sin(\boldsymbol{\theta}_z) & \cos(\boldsymbol{\theta}_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & -\mathbf{p}_X \\ 0 & 1 & 0 & -\mathbf{p}_Y \\ 0 & 0 & 1 & -\mathbf{p}_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ **equation 4-4**

4.1.2 THE PROJECTION PROBLEM

The essence of the projection problem is finding images of the virtual tumor that, when projected on the mannequin's surface, are observed by the user as if they stem from real 3D tumors fixed inside the body. A 2D schematic of this situation is shown in figure 4-4.

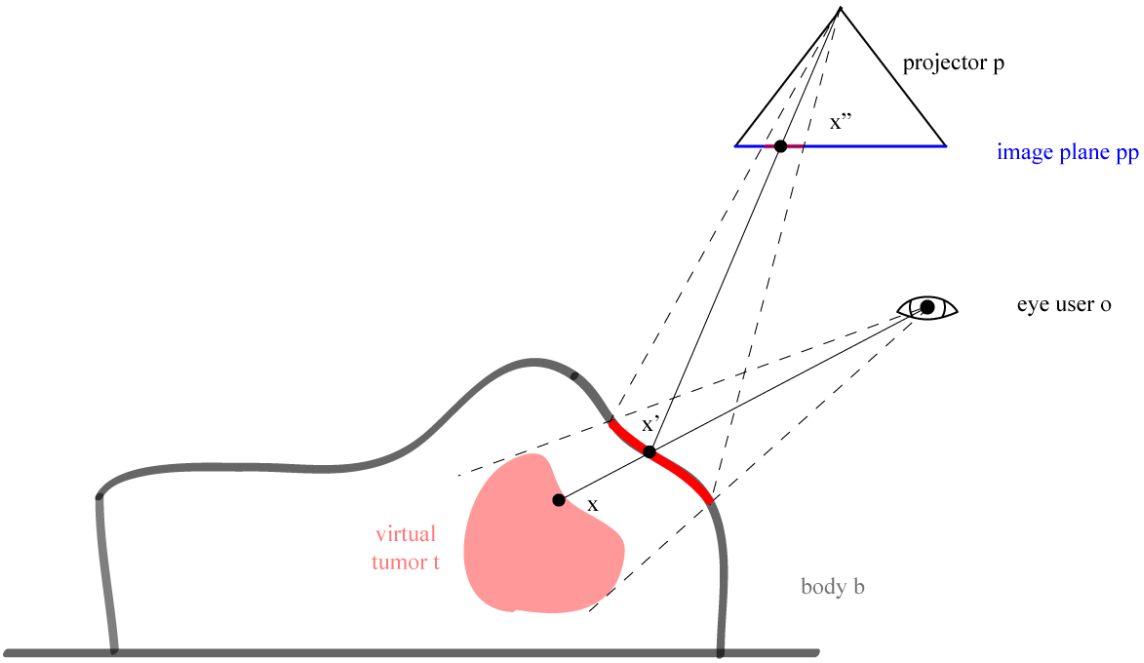


figure 4-4 – The projection problem

We define the following additional variables:

b	$-$ *The surface of the body substitute; in 2D a closed contour.*
t	$-$ *The tumor model, a closed contour; must lie completely within b..*
T	$-$ *The collection of all points on t that are visible from the user's point of view.*
$\mathbf{o}_{x,y,z,w}$	$-$ *The homogenous position of the user's eye.*
$\mathrm{k}(\mathrm{A}, \mathbf{a}, l)_{xyzw}$	$-$ *The projection of a set of points* A *onto a contour l in the direction of* **a**.
I	$-$ *A set of points that together forms the desired image.*

The solution to the problem can be described as first finding the set of points T′ that consists of the projection of T on the body surface according to the user's point of view. The desired image of the projector is then formed by the projection of T′ onon the projector. This is expressed in equation 4-5.

$$\mathrm{I} = \mathrm{k}(\,\mathrm{k}(T, \mathbf{o}, b), \mathbf{p}, pp) \qquad \textbf{equation 4-5}$$

In equation 4-5, pp is the projector plane of the projector. The proposed solution is modeled in 2D, the generalization to 3D is straightforward, however. To do this, every closed contour simply becomes a closed surface, and pp becomes a plane instead of a line. To find a stereoscopic image, we simply utilize the above equations twice, with different positions for **o**.

4.2 RADIOMETRIC MODEL

To display the model, not just the position of every point in the image plane must be found, the surface colors of the tumor and the guidance grid have to be calculated as well. A realistic radiometric model consists of: A realistic lightning model for determining the brightness of the surface based on the light sources and several factors like (self-) shadowing. A realistic model for the reflectance of the tumor surface. And finally a way of compensating for the color of the patient's skin, our projection surface. However, due to time constraints, the radiometric system was kept simple.

Our system uses an additive RGB color space. We use a single primary color to render the tumor. To give the model depth and definition a simple lightning model is applied. Since light in outside daylight falls from above, points on a surface are brighter when they are "facing up". Our lightning model therefore makes points brighter based on the angle between the surface normal at that point and a vector that is pointing up. This model is based on the assumption that all light comes from above, without taking shadows into account. It allows for a simple, clear reading of the shape of the model, without adding visual noise that might influence the user. An image showcasing a rendering with this lightning model, as well as two others, can be seen in figure 4-5. The model is defined mathematically in equation 4-6 and equation 4-7.

$\mathbf{r}^{\mathbf{a}}_{r,g,b} - resulting\ color\ at\ point\ \mathbf{a}$
$h^{\mathbf{a}} - resulting\ brightness\ at\ point\ a, ranging\ from\ 0\ (dark)\ to\ 1\ (bright)$
$\mathbf{n}^{\mathbf{a}}_{x,y,z} - normal\ at\ point\ a$
$\mathbf{u}_{x,y,z} - [0,1,0] - normalized\ vector\ pointing\ upwards$
$\mathbf{g}_{r,g,b} - primary\ \ color\ used\ for\ model$

$$\mathbf{r}^{\mathbf{x}} = h^{\mathbf{x}}\mathbf{g} \qquad \textbf{equation 4-6}$$

$$h^{\mathbf{x}} = \frac{1 + (\mathbf{u} \cdot \mathbf{n}^{\mathbf{x}})}{2} \qquad \textbf{equation 4-7}$$

Since both n and o are of length 1, their dot product $o \cdot n$ equals the cosine of the angle between these two vectors. This ensures$0 \leq h \leq 1$.

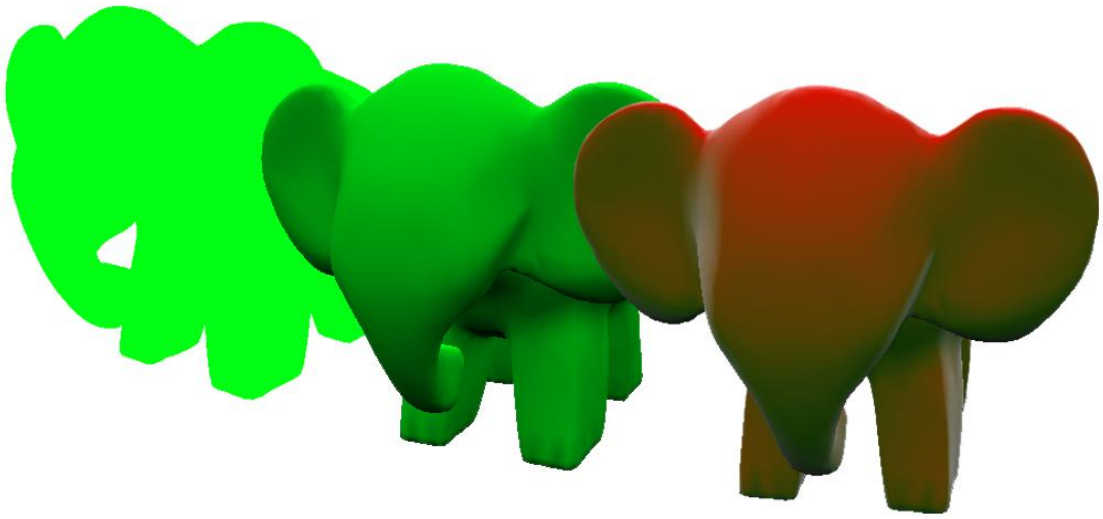


figure 4-5 – Elephant 3D model rendered in different way. From left to right: just base color; lightning model described above; the same, with additional red glow from a above and rim lightning added, this rendering method is not used in this project.

In this chapter, we defined the problem that was the software in the demonstrator must solve in mathematical terms. The next chapter will describe how this model in translated to actual software. Furthermore, the lighting model that is used to render the 3d model of the tumor was described. We will look at how this model can be made to run in real time on 3d accelerator hardware in the next section as well.

5 SOFTWARE IMPLEMENTATION

The software system faces the problem of generating images of the tumor that make its geometry appear undistorted to the user, despite the uneven surface that it is projected on. The software takes as inputs: (visualized in figure 5-1)

- 3D models of the tumor and the mannequin. The tumor geometry is used to create the image, the mannequin geometry used to warp the image so that the tumor will seem undistorted when this image is projected on the mannequin.
- The position and orientation of the users head and the mannequin in the world.
- The extrinsic en intrinsic parameters of the projector. Extrinsic parameters describe its position and orientation, while intrinsic parameters are the parameters of its model, such as the horizontal field of view.
- The position inside the mannequin at which we want the virtual tumor to appear.

The software uses this information to render the required image, and sends this to the projector. It projects it in turn onto the mannequin, where the user can see it.

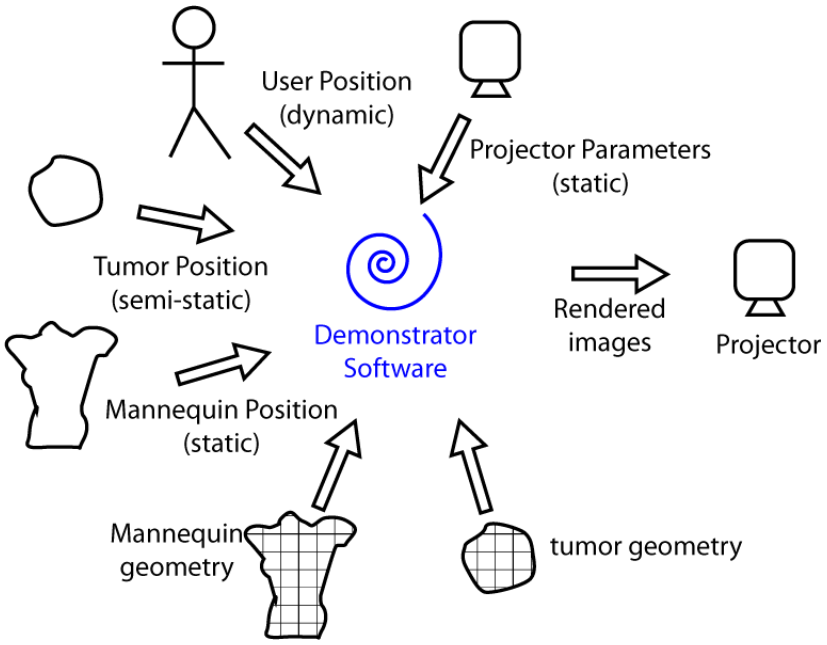


figure 5-1 – Demonstrator software inputs and output.

Calibrating the demonstrator consists of inputting the position and orientation of the mannequin and the projector parameters in the global coordinate system. Due to time constraints, these were measured and inputted by hand. A possible improvement to the demonstrator is measuring these more accurately using the electromagnetic tracker that is used to track the user. The calibration was checked using a test object of known geometry (a sphere). By checking if its projection accurately covered the surface of its real-world counterpart, the accuracy of the intrinsic projector parameters could be checked. The user position is measured using the electromagnetic tracker described before. All measured positions are relative to the tracking beacon, so they have to be converted to the global coordinate system. Measuring and inputting the position of the beacon is therefore a part of calibration as well, and was likewise performed manually.

The software generates the image in several steps; these are visualized in figure 5-2. First an image of the tumor is rendered from the position of the users eyes, let's call this image *A.* A is exactly what we want the user to see, despite the fact that it is projected on an uneven surface. To achieve this, a few

extra steps have to be taken. First A is projected from the eye position onto the geometry of the mannequin. The result is then rendered from the projectors position, let's call the resulting image *B.* Intuitively, we can see that by if the supplied positions, projector model and 3D models match their real-world counterparts, projecting B results in the user observing A.

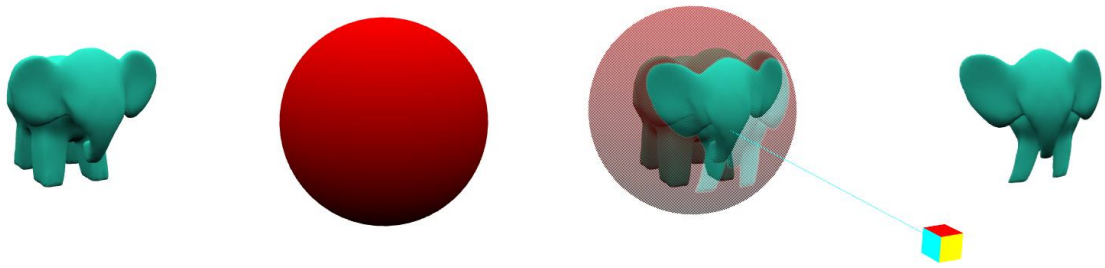


figure 5-2 - The solution to the projection illustrated. The elephant represents the tumor, the red ball represents the mannequin, the colored block represents the users viewpoint *o*.

5.1 3D MODELS

This chapter describes how the 3D models used in the demonstrator are acquired. The surface model of the mannequin was created by scanning the physical object using structured light. This is described in chapter 5.1.1. This approach was chosen because the 3D model has to match the physical object as closely as possible. This is because its geometry is the basis for the deformation we apply to the rendered image. The tumor models were modeled by hand, since no real patient data could be found. The results are shown in chapter 5.1.2.

5.1.1 3D SURFACE MODEL OF THE MANNEQUIN

To get a surface model of the mannequin, it was scanned with the use of structured light. The setup that was used was realized by E. Schippers [11]. Structured light is based on stereo triangulation. Needed are a projector and a camera with known intrinsic and extrinsic parameters. Both should be aimed towards the same point in the scene to be measured. The beamer projects a pattern onto the scene. In the image that the camera captures points can be associated with corresponding points in the projected pattern. Because the position and orientation of the projector and camera are known, these points define lines through the points on the image planes. This situation is shown in figure 5-3.

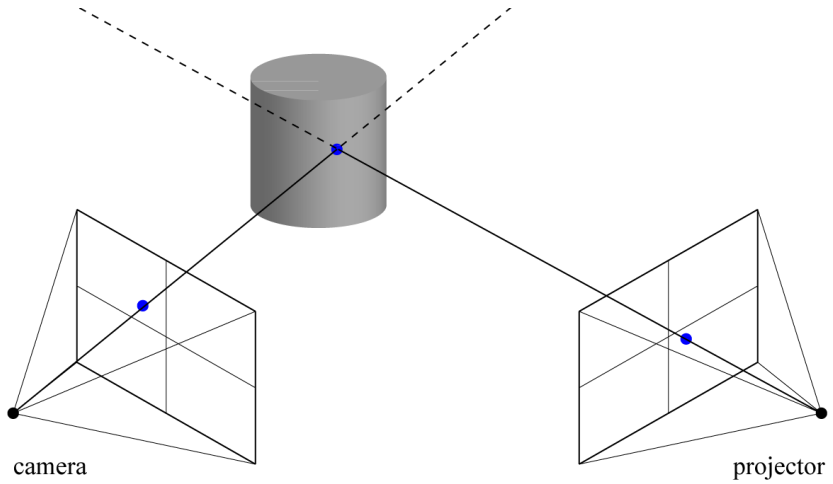


figure 5-3 – Theoretical scanning process using structured light. The blue point from the projected pattern is detected in the camera's image. The depth of the scene of the intersection of the defined lines becomes known.

In order to find a correlation between points in the projected pattern and the captured camera image Schippers' system uses binary time codification. In this method a series of stripe patterns is projected sequentially. The first pattern is half light, half dark. For every pixel in the camera's image we can now detect in which part of the projected pattern they are positioned. By subdividing each half, the area that defines possible locations for the point is halved as well. Only after every pattern is projected a final relation is found between points. Therefore the relative position of the beamer, the camera and the scene cannot change during this time. The connection between an observed value at a certain point, the projected pattern and the possible locations of that point is shown in figure 5-4.

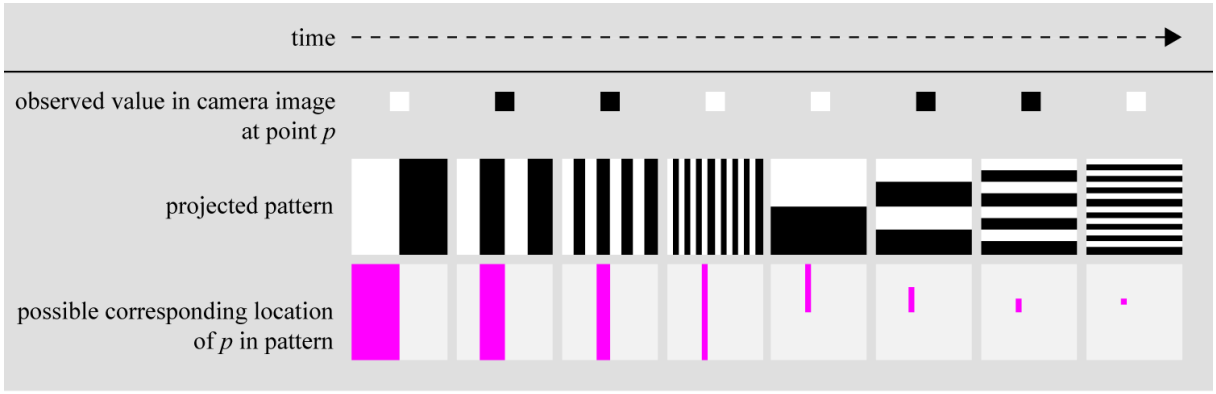


figure 5-4 – Binary time-coded structured light.

The scanning process for the mannequin is shown in figure 5-5. The mannequin could not be scanned in one pass due to limitations of the field of view of the physical rig. Therefore the point clouds had to be aligned, after which a surface over the point clouds could be calculated. The exact procedure is detailed in Appendix A.

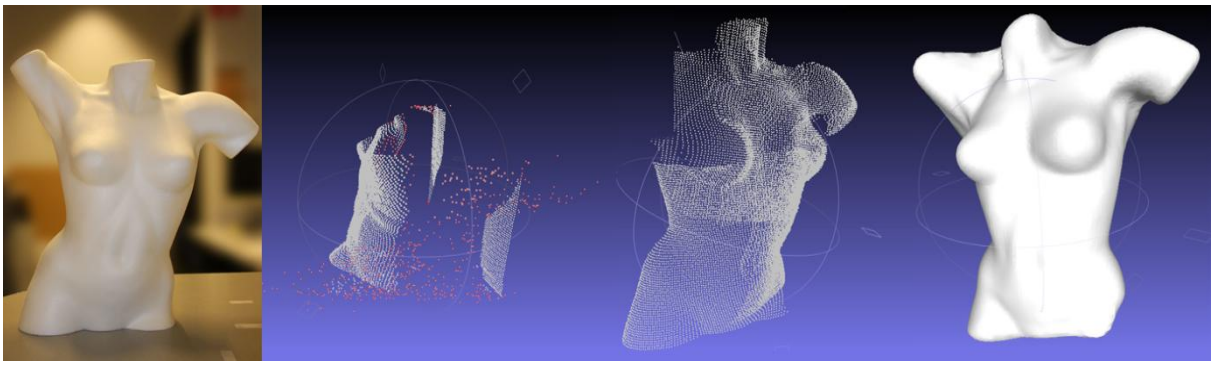


figure 5-5 – The scanning process for the mannequin. From left to right: The mannequin; A single point-cloud with outliers and unwanted surfaces; The point clouds cleaned and aligned; The final 3D model.

5.1.2 THE TUMOR 3D MODELS

Two different 3D models are used to represent the tumor in the system. They were modeled using the Blender 3D software. One is modeled after real-life tumors, to give a realistic impression. The other one is designed in a way to maximize the stereoscopic depth effect. To achieve this, the model must have an open structure, to allow the user to see the complete depth of the model at a glance. A Gordian knot answers to these description. Both models are showcased in figure 5-6.

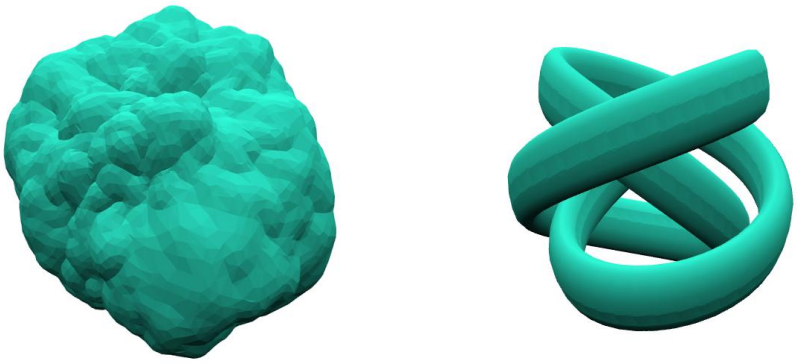


figure 5-6 – The two tumor models used in the demonstrator.

5.2 DEMONSTRATOR IMPLEMENTATION

This chapter describes the implementation in software of the radiometric and geometric model. To create a software system that generates 3D images in real-time, the use of a 3D accelerator card through a 3D API is recommended. The OpenGL API was chosen for this purpose. OpenGL supports three types of graphical primitives, being point, lines and convex polygons. The coordinate system OpenGL uses is shown in figure 4-1. Points and lines are rendered with constant size, irrespective of distance to the camera. This makes them mostly unsuitable to application in our system, so they won't be discussed further. All polygons are converted to triangles internally, in a process called triangulation. A triangulation of a polygon is shown in figure 4-2. This immediately clarifies why only convex polygons are allowed, since their conversion to triangles is trivial.

While defining vertices it is possible to define extra parameters for these vertices. The parameters used in this system are:

- Color.
- Normal.
- UV coordinates.

When OpenGL first came out, it was configurable to a limited extent. Several choices could be made that influenced the rendering process, but no new lightning and shading processes could be implemented. This was called the *fixed function rendering pipeline*. In the last few years this has changed with the rise of the so-called *programmable rendering pipeline*. It enables uploading of small programs that directly influence the rendering process. Such programs are called *shaders*. Every modern OpenGL implementation allows at least two types of shaders, vertex shaders and fragment shaders. Shaders must be written in GLSL (OpenGL Shading Language), a programming language based on C. The main program that uploads the shaders is referred to as the *host program*.

A vertex shader is run once per vertex, and get as inputs: the 3d position of the vertex; the defined parameters (color, normal, uv coordinates); variables describing the current OpenGL state like the camera-matrix **C**; and finally all variables that the host programs wishes to expose to the shaders, called uniforms. The vertex shader is required to output a coordinate that represents the position of the vertex in screen space. In a basic case, this can be calculated by multiplying the input position with **C**. Furthermore, the vertex shader can output any number of variables for use in the fragment shader.

For every triangle supplied, the vertex shader is first run for all three vertexes. Next it is determined which pixels in the final image are covered by the triangle. A small part of a triangle represented by one pixel is called a *fragment* in OpenGL. As such the fragment-shader is run once for every one of these pixels. The fragment shader gets as inputs: the variables that were outputted by the vertex-shader, linearly interpolated over the surface of the triangle. As output the fragment shader is required to supply the color and transparency of that pixel.

5.2.1 IMPLEMENTATION RADIOMETRIC MODEL

The combination of the vertex shader shown in code fragment 5-1 and the pixel shader shown in code fragment 5-2 implements the radiometric model consisting of the color and lightning model described before. An example of a model rendered with these shaders can be seen in the center of figure 4-5.

code fragment 5-2 - Vertex shader for geometric model

```
#version 120

varying vec3 vNormal;
varying vec4 vColor;

void main(void)
{
	// Project this vertex using the projection matrix:
	gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;

	/* Output variable for use in the fragment shader: */
	// Pass surface normal.
	vNormal = gl_Normal;
	// Pass model base color.
	vColor = gl_Color;
}
```

code fragment 5-3 - Fragment shader for radiometric model

```
#version 120

varying vec3 vNormal;
varying vec4 vColor;

void main(void)
{
      // Define a vector pointing up.
      vec3 up = vec3(0,1,0);
      // Calculate brightness based on the normal.
      float v = (1.0 + dot(up, vNormal)) / 2;
      // Output color based on brightness times the base color.
      gl_FragColor = v * vColor;
}
```

5.2.2 IMPLEMENTATION GEOMETRIC MODEL

This section describes the implementation of the geometric model in software. The projector model can be implemented in OpenGL using the projection matrix shown in equation 4-4. Although the matrix can be created by hand, helper functions are available for creating it. The model is defined in terms of the display ratio (height/width ratio of the image) and vertical field of view. We know the height/width ratio of this projector to be ¾, measurements confirm this. The vertical field of view was determined through measurements to be 19.25º. Using these parameters, we can use the following code snippet to calculate the projector matrix. This matrix will be available to shaders as a variable named *gl_ProjectionMatrix*.

code fragment 5-4 – setting up the projector matrix

```
GLU.gluPerspective(camFovY , displayRatio, 0.1f, 1000.0f);
GLU.gluLookAt(position.x, position.y, position.z,  target.x, target.y,
      target.z, up.x, up.y, up.z);
```

A possible solution to the projection problem is to calculate the projection of every point T on **b** in the host program, in the CPU. By sending these transformed points to OpenGL we can directly generate the desired images. This method is however too slow in practice to execute in real-time, since use of the 3D accelerator card is circumvented. A solution that does the required calculations on the 3D card, in a shader, is therefore preferable. A problem in this case is, however, that shaders only have access to the position of the current vertex, making it impossible to calculate intersections with a surface. To circumvent this limitation, we render the final image in two steps, with a bitmap image as a temporary buffer for the intermediate result.

- Firstly an image *i* is rendered of the tumor *t* from the position of the users eye *o*, using a projection matrix *O*.
- Next the 3D of the body model is rendered from the projectors position *p* using the projectors projection matrix **C**.
    - To determine the color at a certain vertex **x** in the body model, we sample the color in image *a*, at position $\mathbf{x} * \mathbf{O}$. This process is explained visually in figure 5-7.

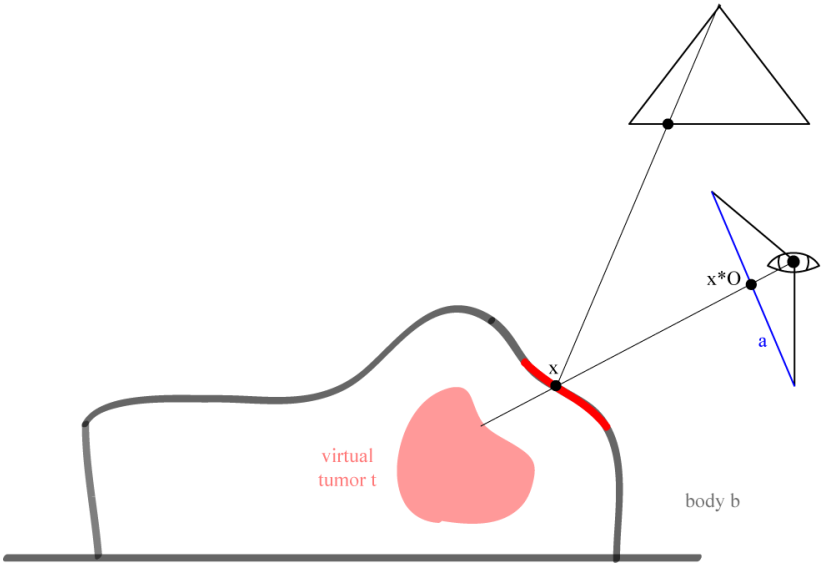


figure 5-7 – Implementation of projection problem using intermediate image *a*.

5.2.3 IMPLEMENTATION STEREOSCOPIC 3D

In 3D projection, both eyes receive a different image, thus enabling the illusion of 3D.

Using standard projectors, one possible modality that can achieve this separation is the light spectrum. In this scheme, both images are projected simultaneously in a different color, for example red and cyan. The user wears glasses with a red filter in front of the left eye, and cyan in front of the right. Therefore, the red image only reaches the left eye while the cyan image only reaches the right, thus achieving the desired effect. Because of the use of color as a separation modality, it is very hard to achieve full-color images using this method.

Another modality we can use is time, i.e. sequentially projecting frames destined for the left and the right eye. The user wears so-called shutter glasses, which are synchronized to the projector and block out (shut out) the images not meant to be seen by making each lens opaque in rapid transitions. However, due to the relatively low refresh rate in common projectors (60 Hz, resulting in only 30Hz per eye), special-purpose projectors will have to be used. Furthermore since the shutter glasses are active, they will need a source of power and a synchronization signal, leading them to be unwieldy.

Using additional hardware, polarized light can also be used to separate images. This scheme was used by Rasker at the University of North Carolina. [12] He used two completely overlapping projections, one for each eye. Polarizing filters in front of the projectors and in glasses worn by users were used to make sure each image only reached the correct eye.

The solution chosen uses the color-separation method. To generate the images using OpenGL, the tumor is rendered twice to two different bitmap images. The eye position for each rendering is adjusted accordingly, as well as the tumors primary color, which is either pure red or pure green. By sampling the two images and adding the two colors the final image is obtained. An example rendering can be seen in figure 5-8.

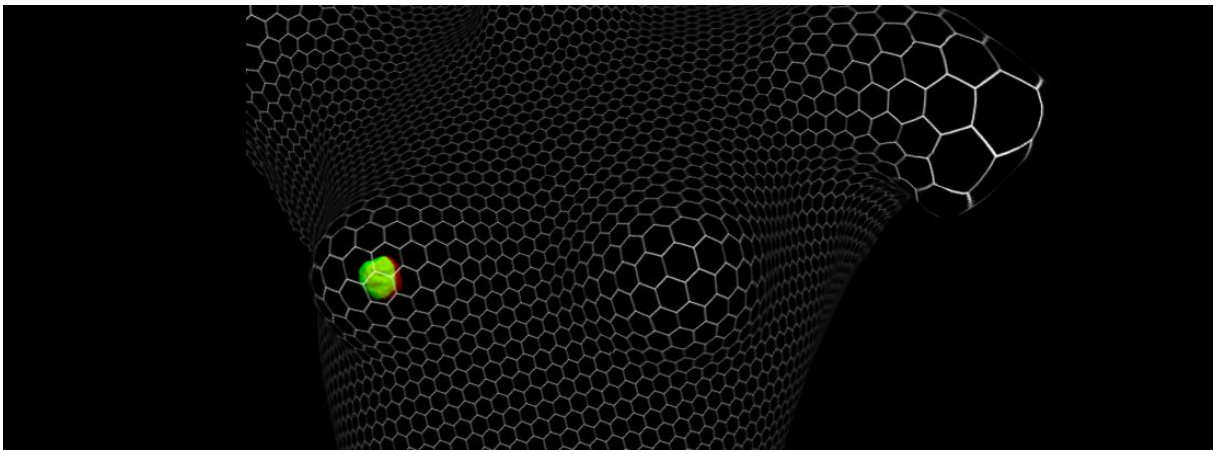


figure 5-8 – A stereoscopic rendering of the tumor model projected on the body, shown with guidance grid.

5.3 PERFORMANCE

Several factors decrease the accuracy of the measurements taken in the user test. First of all, the body might have shifted a little from its ideal position. Because the table the body rested on wasn't secured, the body might not have been in the exact position the software expected it in. This leads to some error in the projection of the tumor model, as shown in figure 5-9. Calculating the exact error is complex as it depends on the amount of shift, the position of the user, and the geometry of the mannequin.

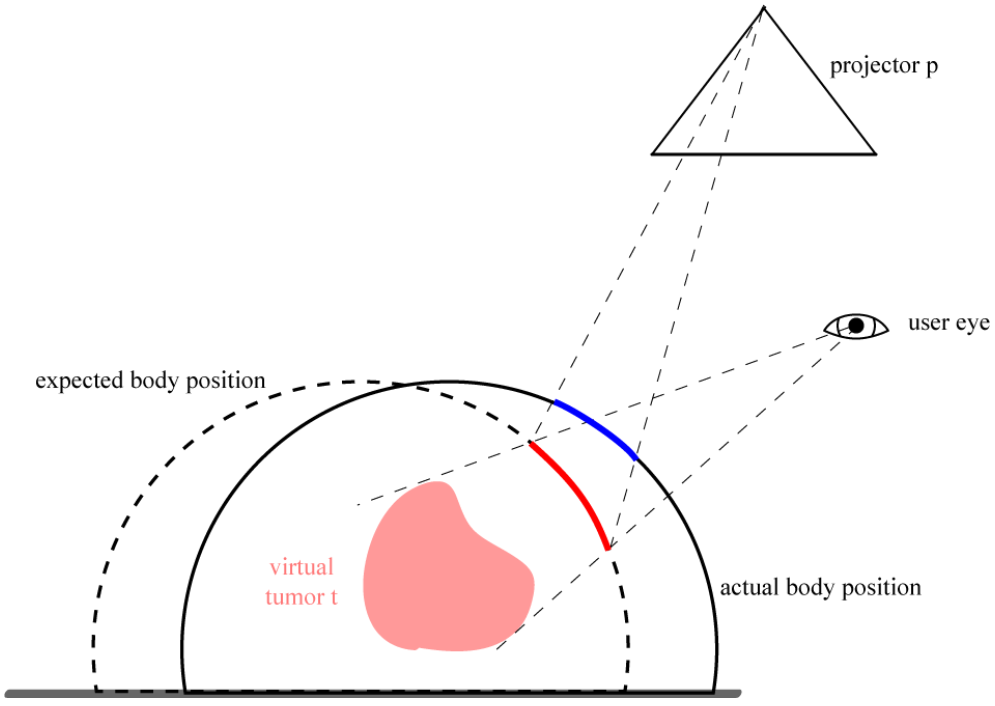


figure 5-9 – Error due to body shift. Correct projection shown in red, actual projection shown in blue.

Furthermore, the relative position of the beamer and the electromagnetic beacon was measured by hand, introducing a small, but constant error. Also the tracking solution has some uncertainty in its output. The documentation specifies that output values are within 1.4mm of the actual location, with 95% confidence. However, the measured position might jitter around the actual position, leading to a

dynamic error and a model that seems to be ‘swimming’ a bit. The latency in the electromagnetic tracker contributes to this effect as well.

The projector model used is also a (small) source of error, since advanced aspects such as lens-distortion aren’t modeled. [13] A more accurate model could offer improvements, although that might be hard to implement in real time in OpenGL since more complex camera models are not modeled by a simple projection matrix. The error depends on the quality of the lens in the projector. No attempts have been made to quantify this error.

The use of bitmaps in the OpenGL implementation is also a source of error. If the normal of a body’s surface patch strongly deviates from the direction towards the projector, the projected resolution (dots per inch) is noticeably low. . This causes some aliasing artifacts, in which pixels become visible, and lines are no longer smooth.

5.3.1 CORRECTING FOR SYSTEM ERROR

Because of the above system errors, the positions on the surface that are closest to the virtual tumors are not fully known. Therefore, it doesn’t make sense to directly compare the positions that users indicated to these ideal positions. However, because the system error is the same for each scenario, we only compare differences between scenarios. After all, the users experience the same error in each scenario. This approach essentially eliminates the influence of the system error.

To further increase the accuracy of the measurements, every location the users indicated is projected onto the virtual mannequin. That is, the closest point to the location that is on the body’s surface is used instead of the actual location. Since every point indicated is supposed to be on its surface, this method is bound to decrease error. This compensates for the times the position was recorded slightly before or after the user pressed the sensor on the surface, and also for the fact that the recorded position lies in the middle of the sensor instead of at the point of contact.

6 USABILITY TESTING

6.1 GOAL

User tests were done to determine the influence several factors have on the user's accuracy and speed in a path planning task. The factors tested are the use of stereoscopic 3D, the display of a guidance grid, and the use of the different tumor models. The task was trying to pick a point on the mannequin's surface closest to a virtual tumor model displayed within the model. The goals are formalized in the following hypothesis, which are rejected or accepted based on the data gathered.

Hypothesis 1. The use of stereoscopic 3D allows users to pick the closest point with greater *accuracy* then in the case without stereoscopy.

Hypothesis 2. The use of stereoscopic 3D allows users to pick the closest point with greater *speed* then in the case without stereoscopy.

Hypothesis 3. The display of a stereoscopic grid and stereoscopy allows users to pick the closest point with greater *accuracy* then in the case without the grid but with stereoscopy.

Hypothesis 4. The display of a stereoscopic grid and stereoscopy allows users to pick the closest point with greater *speed* then in the case without the grid but with stereoscopy.

6.2 PROCEDURE

19 users were tested. Each user was assigned either one of the two tumor shapes for the entire duration of the test. The users were shown the tumor model in 3 scenarios, each scenario consisting of 15 different locations for the tumor. For each tumor location, the user was asked to pick a point on the mannequin's surface closest to it. This position was recorded. The scenarios were as follows:

- Scenario *s1*: Tumor shown **without** guidance grid and **without** stereoscopic 3D.
- Scenario *s2*: Tumor shown **without** guidance grid and **with** stereoscopic 3D.
- Scenario *s3*: Tumor shown **with** guidance grid and **with** stereoscopic 3D.

Each user is expected to do a little better on the second scenario, and even better on the third, due to him/her getting used to the equipment and the task; an influence we dubbed *the learning effect.* To "even out" this effect over the complete data set, the subjects were subjected to the scenarios in a randomized order. Furthermore, the order in which the locations were shown in each scenario was randomized, to try and prevent users from learning and recognizing locations from previous scenarios. The data that was recorded was the position that each user picked for each of the locations in each of the scenarios, and the time it took the user to pick it.

6.3 MODEL

The following variables are defined:

$\mathbf{l_1}, \mathbf{l_2}, \dots, \mathbf{l_{15}}$ $-The\ virtual\ locations\ of\ the\ 15\ tumours.$
$\mathrm{d}(\mathbf{x}, \mathbf{y})$ $-The\ distance\ between\ point\ \mathbf{x}\ and\ \mathbf{y}$
s_1, s_2, s_3 $-The\ scenarios\ described\ above.$
$u_1, u_2, \dots, u_{19}$ $-The\ different\ users.$
$\mathbf{p}_{s_y, l_z}^{u_x}$ $-The\ location\ the\ user\ u_x\ picked\ in\ scenario\ s_y\ when\ the\ tumor\ was\ shown\ in\ location\ \mathbf{l}_z.$
$\mathbf{c}_{l_z}$ $-The\ point\ on\ the\ body's\ surface\ closest\ to\ \mathbf{l}_z$

We define the following sets: (see equation 6-1)

$$\mathrm{P}^{x} = \left\{ \frac{d(\mathbf{p}^{u_a}_{s_{x-1},l_b} - \mathbf{l}_b) - d(\mathbf{p}^{u_a}_{s_x,l_b} - \mathbf{l}_b)}{d(\mathbf{c}_{l_z} - \mathbf{l}_b)} \middle| a = 1, \dots, 20 \; b = 1, \dots, 15 \right\} \; x = 2 \; or \; 3 \qquad \textbf{equation 6-1}$$

An element from P^{x} indicates the relative difference in path lengths between the tumor and its chosen location between two scenarios. This difference is relative to the actual shortest length. The relative difference has been used, rather than the absolute difference, in order to weigh the deeper situated tumors more heavily than the tumors that are near the surface (and thus easier to localize). To test hypothesis 1, we need to have data on the relative performance of users in scenario *s2* compared to *s1*. This data is contained in the point setP^2. Likewise for hypothesis 3 the data is contained in P^3. The data for the hypothesis on users speed, 2 and 4, is found in a completely analogous way. Because a minimal time to pick a point cannot be defined, we don't divide the time difference. So, we define:

$t^{u_x}_{s_y,l_z}$ – *The length of time it took user x to pick a location when the tumor was shown in location* l_z *in scenario* s_y.

and build the following sets: (see equation 6-2)

$$\mathrm{S}^{\mathrm{x}} = \{ t^{u_a}_{s_{x-1},l_b} - t^{u_a}_{s_x,l_b} | a = 1, \dots, 20 \; b = 1, \dots, 15 \} \qquad \textbf{equation 6-2}$$

The data needed to accept or reject hypothesis 2 is contained in S^2, the data for hypothesis 4 in S^3.

6.4 STATISTICAL ANALYSIS

The student t test [14] is used to analyze the results. For each hypothesis:

- The applicable dataset is chosen from the sets P and S.
- A normal distribution is assumed for these points: $\mathrm{N}(\mu, \sigma^2)$, with the mean μ and the variance σ^2 unknown. Since we don't know the true values of μ and σ, we use estimators $\overline{X}$ and S in a student distribution.
- We split our hypothesis into two cases, the null hypothesis H_0 and an alternative hypothesis H_1. If H_0 is rejected, then H_1 must be accepted.

$$\mathrm{H}_0 : \mu = 0$$

$$\mathrm{H}_1 : u > 0$$

- The t-value for the student t-test is defined as follows.

$$T = \frac{\overline{R}}{S/\sqrt{\mathrm{n}}} \qquad \textbf{equation 6-3}$$

$$\overline{R} = \frac{1}{n} \sum_{i=1}^{n} R_{\mathrm{i}} \qquad \textbf{equation 6-4}$$

$$S^2 = \frac{1}{n-1} \sum_{i=1}^{n} \left(R_i - \overline{R} \right)^2 \qquad \textbf{equation 6-5}$$

- We test the hypotheses with a confidence level of α. That is, we determine a constant *c* such that $P(T_{n-1} \geq c) = 1 - \alpha$. The t-distribution is tabulated in many textbooks on statistics, e.g. [10]. For α=99.5%, we find *c*=2.58.Thus.
  $P(T_{281} \geq 2.58) = 0.005$
- If *T* is larger than *c*, H_0 is rejected, and H_1 is accepted. This proves with the chosen confidence level that the hypothesis is true.

If the hypothesis is proven true, the 95% confidence interval can be calculated, shown in equation 6-7.

$$T = \frac{\overline{R}}{S/\sqrt{\mathrm{n}}} \qquad \textbf{equation 6-6}$$

$$(\overline{R} - \frac{cS}{\sqrt{n}}, \overline{R} + \frac{cS}{\sqrt{n}}) \qquad \textbf{equation 6-7}$$

with

$$P(T_{n-1} \geq c) = 1 - \frac{\alpha}{2} \qquad \textbf{equation 6-8}$$

This gives a lower and upper bound on the 'true' value of the tested variable, with the chosen confidence level.

6.4.1 RESULTS

Hypo-thesis	**Data set**	H_0	H_1	$\overline{P}$	**S**	**T**	**c**	**95% Confidence interval**	H_0 rejected
1	P^2	$\mu = 0$	$u > 0$	0.174	0.8155	3.593	2.58	(0,079, 0,269)	**yes**
2	S^2	$\mu = 0$	$u > 0$	3.793	11.266	5.654	3.29	(2.478, 5.108	**yes**
3	P^3	$\mu = 0$	$u > 0$	0.076	0.5830	2.215	1.69	(0.008, 0.145)	**yes**
4	S^3	$\mu = 0$	$u > 0$	0.480	7.4801	1.077	1.64	n.a.	**no**

The result from the statistical analysis tabulated here show that using stereoscopic 3d shortens the path users choose between circa 8% and 27%, with 95% confidence. It also allows users to pick points with greater speed. Using a guidance grid offers between 0.9% and 14% improvement in chosen path length over stereoscopy without grid, but doesn't offers any speed improvement.

6.5 PLOT

Figure 6-1 shows the data points obtained from the different scenario's plotted. The points plotted are the recorded user picks projected on the body, relative to the tumor locations which are placed at the origin. The z axis is dismissed for clarity.

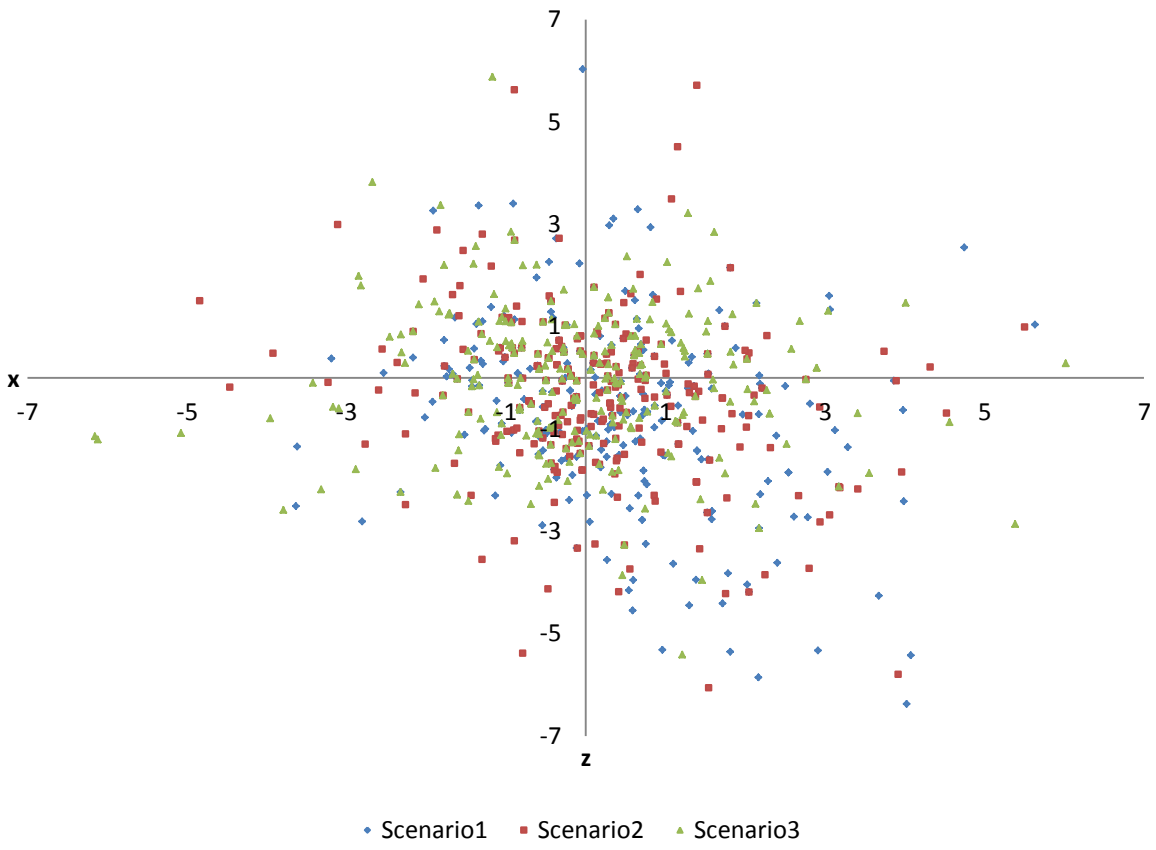


figure 6-1 – Data points from the different scenarios plotted.

6.6 USER PREFERENCE

Every user was asked for a personal preference as to which of the three scenarios they preferred. As this only delivers 19 data points, we will refrain from doing statistical analysis, and just state the numbers. Out of 19 users:

- 2 Users preferred scenario 1, the situation without stereoscopic 3d and without grid.
- 3 Users preferred stereoscopic 3d, but were undecided between scenario 2 and 3.
- 4 Users preferred scenario 2, stereoscopic 3D without guidance grid.
- 10 Users preferred scenario 3, stereoscopic 3D with guidance grid.

This is in line with the results of our statistical analysis.

6.7 3D TUMOR MODEL

In the user tests, users were shown either one of the above tumor models. Due to the varying system error described above, it becomes impossible to compare user results in any meaningful way. Therefore, we can't firmly conclude if this factor has a significant influence on the users' performance in the test.

7 CONCLUSION

In this project the technical feasibility of a real-time projected augmented reality setup with a focus towards the application in a medical setting was proven. A hardware rig was assembled simulating an operating room situation. A mannequin represents the patient, a projector allows projecting images on the mannequin's surface, and an electromagnetic tracker tracks the user's eye position, while also allowing him/her to pick out points on the mannequin's surface.

The software system written for the demonstrator allows generating and projecting images of a tumor model onto the mannequin's uneven, curved surface in a way that makes the tumors geometry seem undistorted to the user. OpenGL was selected to generate the images in real-time. A simple lighting model was designed and implemented using shaders to clearly display the tumor model. Stereoscopic 3D using red-green glasses offer true depth perception, while an optional projected grid accentuates the mannequin's surface.

There are some static and dynamic errors in the system. Manual calibration and a simple projector model may lead to a slight shift in the projected images from their intended location. The electromagnetic tracking has a small dynamic error of up to 1.4mm, as well as some latency, leading to the images being slightly but visibly delayed.

Furthermore, a usability study was held with 19 users to test the various parameters in the projection. This proved that the application of stereoscopic 3D improves both users' speed and accuracy in a simple shortest path picking exercise. Accuracy is improved by 8 to 27 percent, time required to pick a point is shortened with 2.5 to 5 seconds. The display of a guidance grid improves accuracy by 1 to 14 percent, but not speed.

8 FUTURE WORK

This section describes possible improvements in the demonstrator, as well as directions for further research.

First of all, a flaw in the user test conducted is the user-dependent system error that was present. To make the results more meaningful and sound, this error should be eliminated. This can be achieved by measuring the complete setup using the electromagnetic tracker, securing both table and mannequin to a fixed position, using a more accurate model for the projector, and improving the 3D model created from the mannequin.

The user test that was held compares several variables in the projection, but doesn't offer any meaningful comparison to the current workflow. To make this comparison possible, a workflow that shows the generated images on a monitor next to the model instead of projecting them could be added to a user test. In this way a fair comparison to the current workflow can be made.

To improve accuracy and flexibility of the projection, many steps can be taken. Multiple projectors could be combined to make sure that the user is no longer able to obscure the image of the tumor. The blending of these projector images could be a mix between the regular overlap functions and the beamer that is most "in focus" at that given point.

Intensity correction is another possible improvement. The image is now less bright on places where the display surface is steeply sloped. By adjusting the brightness of the projection based on the normal of the mannequin, a more accurate projection can be achieved. The method is described in [8].

Finally, steps can be taken to take the system toward real-world usability. The position of the body will have to be detected in real time, adding a depth sensor to the setup could make this possible. This work could be based on the work of Grimson et. al: [4]. As a more advanced step the 3D model of the body could be automatically deformed based on the readings from this depth sensor, as real bodies aren't static. A realistic model of the human body could aid in this step.

9 WORKS CITED

[1] Ronald T Azuma, "A Survey of Augmented Reality," *Teleoperators and Virtual Environments*, vol. 6, no. 4, pp. 355-385, August 1997.

[2] Ludwig Adams et al., "CAS - a navigation support for surgery.," *3D Imaging in Medicine*, vol. 60, pp. 411–423, 1990.

[3] William Lorensen and Ron Kikinis, "Enhancing Reality in the Operating Room," in *Proceedings of Visualization '93*, Los Alamitos, 1993, pp. 410-415.

[4] Grimson et al., "An automatic registration method for frameless sterotaxy, Image guided surgery, and enhanced reality visualisation.," *IEEE transactions on medical imaging*, vol. 15, no. 2, pp. 129-140, April 1994.

[5] J.P Mellor, "Enhanced Reality Visualization," MASSACHUSETTS INSTITUTE OF TECHNOLOGY, MASSACHUSETTS , Msc. Thesis 1544, 1995.

[6] Ramesh Raskar, Greg Welch, and Wei-Chao Chen, "Table-Top Spatially-Augmented Reality: Bringing Physical Models to Life with Projected Imagery," in *2nd IEEE and ACM International Workshop on Augmented Reality*, 1999, p. 64.

[7] J.-P. Tardif, S. Roy, and J. Meunier, "Projector-Based Augmented Reality in Surgery without Calibration," *The IEEE Engineering in Medicine and Biology Society*, pp. 548-551, 2003.

[8] Hanhoon Park, Moon-Hyun Lee, Sang-Jun Kim, and Jong-Il Park, "Surface-Independent Direct-Projected Augmented Reality," *Lecture Notes in Computer Science*, vol. 3852/2006, pp. 892-901, 2006.

[9] Jens Garstka and Gabriele Peters, "View-dependent 3D Projection using Depth-Image-based Head Tracking," University of Hagen, Hagen, 2010.

[10] Gregory Baratoff and Scott Blanksteen. Tracking Devices. [Online]. http://www.hitl.washington.edu/scivw/EVE/I.D.1.b.TrackingDevices.html

[11] E. Schippers, "Solving ambiguity problems in phase based profilometry," University of Twente, Enschede, Masther Thesis 2010.

[12] Ramesh Raskar et al., "Multi-projector displays using camera-based registration," in *Visualization '99. Proceedings* , San Francisco, CA, USA , 1999, pp. 161 - 522.

[13] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

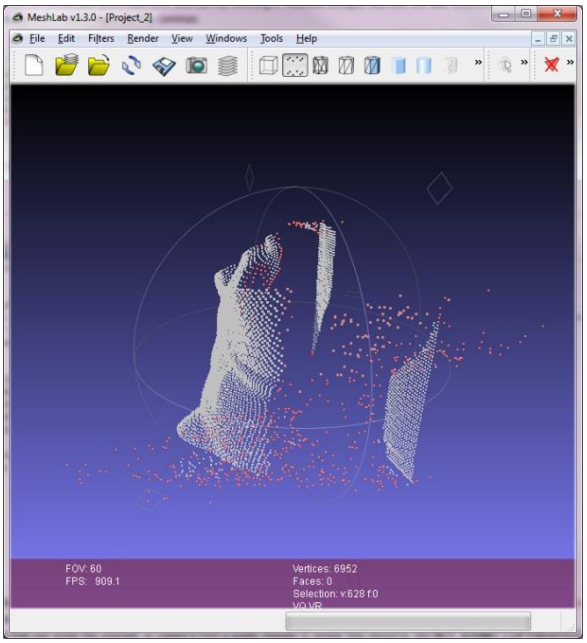
[14] K Poortema, *Inleiding statistiek*. Enschede, Overijssel: Universiteit Twente, 2006.

10 APPENDIX A: GENERATING A 3D MODEL FOR THE MANNEQUIN

To process the point cloud that the structured light process delivered, the open source Meshlab software package was used. The exact process is detailed below.

CLEAN POINT CLOUDS

If you've got some noise in your source data you probably want to get rid of that first. If the noisy points are separated from the rest a bit, as in this example (noise indicated in red):



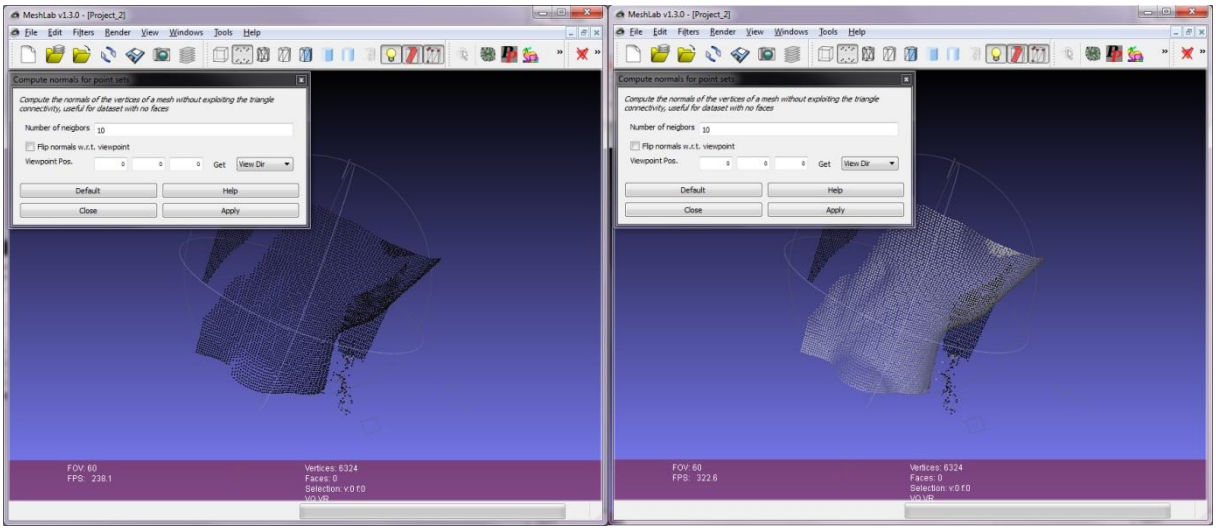
You can select and delete them by:

- Using the vertex select tool directly.
- Estimate a radius for your point by selecting **Filters > Point set > Estimate radius from density**. Default parameters are fine.
    - Select the point you don't want (with a big radius) by selecting **Filters > Selection > Conditional vertex selection.**

Delete your selection with the rightmost delete button in the toolbar.

FIX NORMALS

To fix the normals for your point cloud first turn on the light using the button on the toolbar. Points that are 'facing' you are white, others are black. To fix the normals use the **Filters > Point set > Compute normals for point sets**.



ALIGN POINT CLOUDS

Load all point clouds at once. Use Meshlab's alignment tools to iteratively align the separate point clouds until you are happy with the result.

GENERATE MESH

To finally generate a triangle mesh first flatten all layers (if you have more than one). Then select **Filters > Point set > Surface Reconstruction (Poission)** to generate a triangle mesh. Be sure to change your viewmode to Smooth or Flat to view the results.