FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

3D Druck der Ergebnisse einer fetalen Ultraschalluntersuchung

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Medizinische Informatik

eingereicht von

Eric Mörth
Matrikelnummer 1325306

an der Fakultät für Informatik
der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller

Wien, 27. Juli 2016

Eric Mörth	Eduard Gröller

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

3D-Printing of Fetal Ultrasound

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Medical Informatics

by

Eric Mörth
Registration Number 1325306

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller

Vienna, 27th July, 2016

Eric Mörth	Eduard Gröller

Erklärung zur Verfassung der Arbeit

Eric Mörth
Pfeilgasse 43/2/9
1080 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 27. Juli 2016

Eric Mörth

Danksagung

Ich möchte mich bei Herrn Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller und dem Insitut für Computergraphic und Algorithmen für die Chance dieses Thema bearbeiten zu dürfen bedanken. Ein weiterer Dank gilt den Mitarbeitern des Instituts, die mich bei der gesamten Ausarbeitung und Recherche in allen Belangen tatkräftig unterstützt haben. Das vorhandene Fachwissen über die Verarbeitung von medizinischen Daten und dem Visualisierungs- bzw. dem 3D Druckprozess haben mir sehr oft weitergeholfen.

Mein Dank gilt auch der Technischen Universität Wien, die mich durch die verschiedenen angebotenen Pflicht- und Wahllehrveranstaltungen im Rahmen des Studiums der medizinischen Informatik darauf vorbereitet hat, dieses Thema sachgemäß bearbeiten zu können.

Ich möchte ebenfalls meinen Eltern für die Unterstützung danken, da ich ohne sie nicht in der Lage wäre, dieses Studium bewältigen zu können. Ein weiterer Dank gilt meiner Freundin und meinen Freunden, die immer verständnisvoll und hilfsbereit sind.

Acknowledgements

At first I would sincerely like to thank Mr. Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller and the Institute of Computer Graphics and Algorithms for giving me the opportunity to work on this topic. I would also like to thank the employees of the institute, who supported me through the whole researching, programming and writing process of this bachelor thesis. The given knowledge about medical image processing and visualisation, as well as the knowledge about the 3D printing process have been a great help for me.

I also have to thank the Technical University of Vienna for giving me the possibility to study in the field of medical informatics, by providing lectures about visualisation and medical basic knowledge. Visiting obligatory and electoral lectures of the study plan enabled me to cope with the challenges of this topic.

Another big thank-you belongs to my parents, who give me the opportunity to study at the Technical University in Vienna and the given support in every other belonging. I would also like to thank my girlfriend and my friends, for supporting me and for their understanding.

Kurzfassung

Die 3D Ultraschall Untersuchung von schwangeren Frauen is heutzutage eine der Standarduntersuchungen im Bereich der medizinischen Informatik. Es gibt einige Möglichkeiten, die erhaltenen Daten zu verarbeiten. Eine davon ist es, die Daten mit einem 3D Drucker auszudrucken. Das Problem dabei ist die Konvertierung der Volumsdaten in eine Struktur, die ausgegeben werden kann. Die Drucker benötigen Objekte, die eine verbundene Oberfläche aufweisen. Die vorliegende Arbeit beschäftigt sich mit dem Problem, wie man eine solche berechnen und passend speichern kann. Die Lösung ist gegeben durch die Berechnung der „Connected Component Analysis“ und der Ausführung des „Marching Cubes Algorithm“. Der letzte Schritt der Bearbeitung besteht darin, die berechneten Strukturen in einem STL File zu speichern. Das Ergebnis der Arbeit zeigt, dass es möglich ist die 3D Ultraschall Daten zu drucken. Verschiedene Personen sind in der Lage das Gesicht des Fötus in den ausgedruckten 3D Objekten zu erkennen.

Abstract

The 3D ultrasound in prenatal diagnostics is nowadays a standard investigation in the field of medical informatics. The acquired data can be used in lots of different applications. One of them is to fabricate the fetus model using a 3D printer. The problem here is to convert the given volume data into a structure that can be printed. Current generation of 3D printers expect as an input objects defined by closed surfaces. This work handles the problem of how to calculate such surfaces. Our solution relies on the marching cubes algorithm that extracts the surface out of the volume data. The extracted surface is then refined. The last processing step is to save the data into an suitable data format. The results demonstrate that it is possible to print the fetus model from the 3D ultrasound data and that people are able to perceive the face of the fetus in the fabricated objects.

Contents

CHAPTER 1

Introduction

The 3D printing technology has tremendously improved in the last decades. In the past 3D printing has only been available for industrial prototyping or for making small batches. The last years 3D printers, which are based on an additive technology, became affordable for the private sector. This trend is described by Fasterman in his work "3D-Drucken: Wie die generative Fertigungstechnik funktioniert" [1].

3D printers are getting more mobile and can be installed in a common household or in offices. They can be used for scientific and university projects, rapid prototyping needs and for example for the architecture and industrial design. 3D printers are one of the major technologies to provide immediate data exploration in time-critical situations. One of these is preparation for an operation on the human skull, where the surgeon has to be very careful in order to prevent damaging the patient's brain.

There is a lot of possibilities for users to fabricate models on their 3D printer. One possibility is to design prototypes, spare parts or other objects in 3D sketching tools like Autocad, Google Sketchup or Blender and then convert them into files that can be processed by 3D printers.

This bachelor thesis focuses on the fabrication of fetus models from 3D ultrasound data. Using the ultrasound technology the doctor is able to analyse the pregnant woman on anomalies of the fetus, like missing or malformed limbs, deformations caused by loops of umbilical cord or facial deformations. The specialist can also determine whether the child is a boy or a girl and the expecting mother is able to see the unborn child. In order to print the fetus model, we extract the skin surface of the fetus from 3D ultrasound data. The fabricated model represents the fetus and enables the expecting parents to see and "feel" their child even before it is born. Siebert [2] has already reported about a project, where a blind woman was able to "see" her fetus for the first time.

1.1 Problem statement

The fabrication of objects with a 3D printer requires the object to be defined as one closed surface. The given input data does not contain this information. Instead we have to calculate the information. First, the data has to be classified into two classes, namely data belonging to the fetus and data belonging to the mother. The purpose of the classification is to remove mother tissues and amniotic fluid that enclose the fetus. These tissues do not belong to the fetus therefore they can be removed safely. Ultrasound investigations often have to cope with different noises, which disturb the image. In order to cope with this noises different filters can be used, which are described later in this work.

Fetal tissues have a specific data value range in ultrasonic images. The surface of the fetus can by found by analysing the regions where the values which belong to the fetus and those which don't are neighbouring. We define the surface of the fetus as the surface that encloses all data points classified as belonging to the fetus. There are different methods to compute this surface. Most commonly the Marching Cubes algorithm by Lorensen [3] and Marching Tetrahedra algorithm by Doi [4] are used. The result of both techniques is the surface of the fetus.

After the extraction, the resulting surface can be smoothed in order to remove bumps and other distortions. The printing of the fetus can be done by using different printers. The most common printer type is based on a technique called fused deposition modelling explained by Gibson [5].

CHAPTER 2

Fetus model print

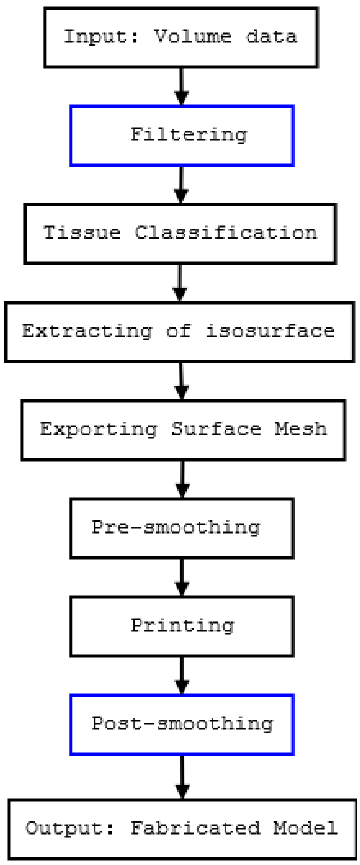


Figure 2.1: Fetus model print pipeline where black boxes are implemented and blue ones are optional.

An outline of the fetus model print method is depicted in Figure 2.1. The black boxes are implemented and used in this work and the blue ones are optional. The input data is the volume information stored in voxels. Voxel comes from vox, which means volume and el which stands for element [6]. Voxels are like Pixels but they are aligned in a three dimensional grid instead of a two dimensional one. The data is stored in bytes, so the value range is from zero to 255. We store them byte wise either. A typical dimension of input data is 276 x 184 x 156.

2.1 Volume data filtering

According to Ambardar [7], ultrasound images have high diagnostic relevance. For accurate analysis the image quality should be good. Usually, the ultrasound data contains a specific type of noise called speckles. Dainty [8] states that speckle is a phenomena where the intensity of the information in the data differs randomly. It can happen, when a coherent light propagates through a medium with a high number of different scattering particles. In our case it is the different tissue forms from the mother and from the fetus and the amniotic fluid. Ultrasound images normally contain significant amount of speckle, shadowing artifacts and also signal losses after passing through dense bones. These things make it difficult to process the images automatically. In fabricated objects these deficiencies manifest themselves as holes or cavities. Different filtering methods can be used to reduce the speckle artefacts. Usually the Wiener filter is employed, however, some authors suggest adaptive filtering. According to Zain [9] and Nadernejad [10], the Wiener filter is a good choice for removing speckle noises, compared to the mean and the anisotropic diffusion filters. This filter is also referred to as the minimum square error or least square filtering by Ambaradar [7]. It performs the smoothing by computing the local variance. The result of the Wiener filter is demonstrated in the Figures 2.2, and 2.3.

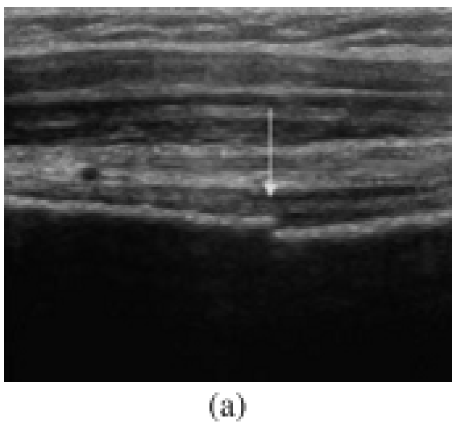


Figure 2.2: Image of proximal humerus shaft fracture [9]

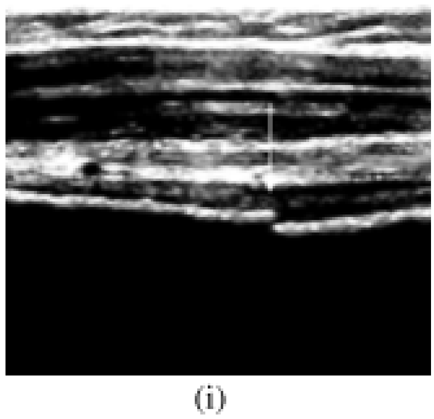


Figure 2.3: Image of proximal humerus shaft fracture after Wiener filter [9]

Rui [11] describes an adaptive filter called Nakagami multiplicative adaptive filter (NaMAF). The NaMAF has a good performance in speckle reduction and it increases the signal-to-noise ratio, which is a measure of signal strength in relation to the background noise. Rui [11] writes that this filter has a better performance in terms of speckle reduction and edges and points preservation than the standard speckle reduction filters like the median and Wiener filter. Looking at Figures 2.4, 2.5 and 2.6 the NaMAF has a better filtering result, compared to the Wiener filter.

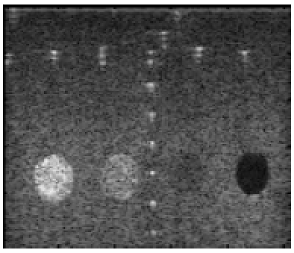


Figure 2.4: Image with Speckle noise [11]

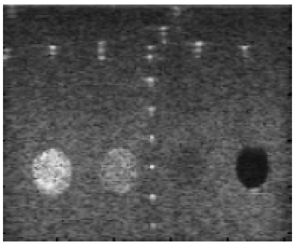


Figure 2.5: Image after Wiener filter [11]

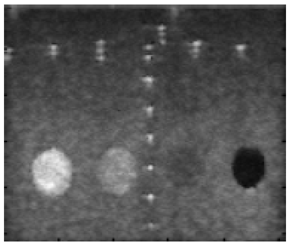


Figure 2.6: Image after NaMAF filtering [11]

2.2 Tissue classification

The value, which separates the data into fetus tissue and mother tissue is the so called threshold. Finding the exact one for extracting surface of the fetal skin is complicated due to non-normalized nature of ultrasound data, significant amount of noise, and individual variations between the patients. This task involves the user interaction, because only the human operator can find the threshold value, which represents the skin surface of the fetus. As a result, we need an interactive visualisation of the volume data coupled with the control that sets the threshold value.

For this purpose we use MeVisLab, a tool developed by the MeVis Medical Solutions AG in cooperation with the Frauenhofer MEVIS research institute [12]. In Figure 2.7 the network of used functions is depicted. It consists of three blocks. The ImageLoad block loads the volume data from the file. Another two blocks are the two different views. The 2D view block displays the standard slice view that is also used by the doctors in a normal ultrasound investigation. The last block creates 3D volume visualisation.

The impact of different threshold values is illustrated in the following pictures. The Figure 2.8 shows the surface, when the value is selected too low. As a result, some mother tissues remain, hiding mouth region of the fetus. In Figure 2.9 the ISO value is too high. This is reflected by the big cavity at the left side of the nose and the missing tissues at the mouth of the fetus. In Figure 2.10 the threshold value is just right. The lips are marked with a blue ellipse and the nose with a green one, both can be seen very clearly and there is no disturbance of other particles. The closed eye which is marked with a red ellipse can also be seen clearly. If the doctors would use better modalities, like magnet resonance

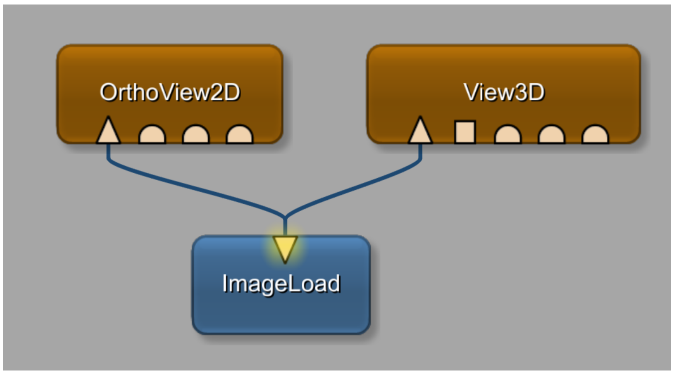


Figure 2.7: MeVisLab network used to display the data in order to find a good threshold

imaging the data could be tracked down to the skin surface of the fetus and noise and other distortions of the ultrasound investigation would play no role. The problem is that such investigations cost much more and maybe harm the development of the fetus.

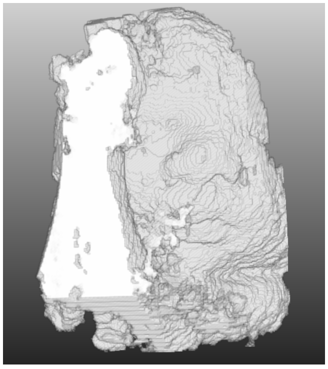


Figure 2.8: Low threshold value leads to occlusion of the mouth region by mother's tissue

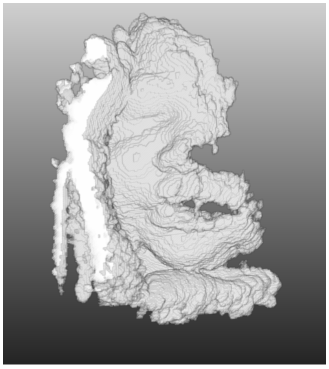


Figure 2.9: High threshold value leads to missing data at the nose and the mouth region

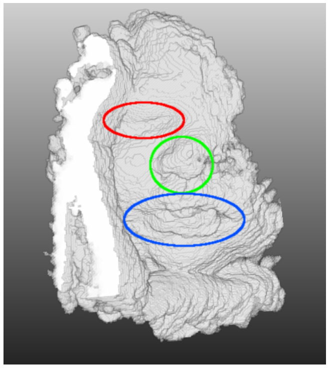


Figure 2.10: Good threshold value, the nose, the eye and the mouth can be seen very clearly

In our data mother tissue and tissue of the fetus can cause severeal connected components in the data. We use the following heuristic: the fetus is the largest connected component in the data. All other components are assumed to be mother tissues and subsequently removed.

3D printing technologies need connected components in order to print one model and not several disconnected ones. The algorithm is breadth-first search on graphs, where each voxel is a vertex and the connections between neighboring voxels are edges and is described in Algorithm 2.1.

The connected component algorithm works with a queue Q and it marks each individual component with a unique component ID. The largest component is then identified by

Algorithm 2.1: Connected Component Analysis

```
Q - queue of voxel coordinates
M - array - input mask
R - array - boolean[] processed voxel flags, initialised with false
C - array - int[], Component ID of voxel, initialised with 0
counterC - int, Component ID counter, initialised with 0
for voxel V ∈ Volume do
    if M[V] and !R[V] then
        Q.enqueue(V);
        R[V] ←true;
        while !Q.isEmpty() do
            D ←Q.dequeue();
            C[D] ←counterC;
            for voxelF ∈neighbourhood(D) do
                if M[F] and !R[F] then
                    Q.enqueue(F);
                    R[F] ←true;
                end
            end
            counterC++;
        end
    end
end
```

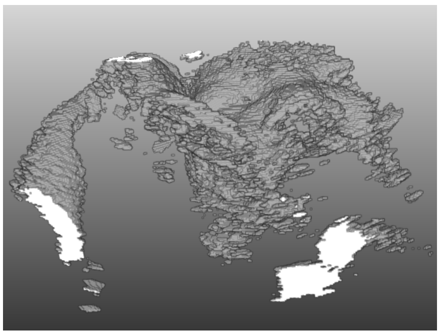


Figure 2.11: Data after thresholding contains several connected components

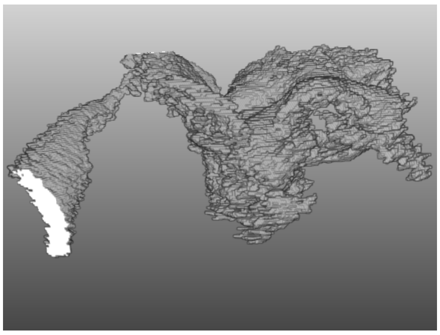


Figure 2.12: After connected components analysis, the data contains the most of the fetal tissues.

summing up the number of voxels. Having identified the connected components, we delete all components except the largest one. The results are illustrated in the Figure 2.11 and Figure 2.12.

2.3 Isosurface extraction

After filtering and classification of the data the isosurface can be extracted. Therefore different algorithm are used. Two of them are described in this section.

Marching Cubes

Marching Cubes is an algorithm, invented in 1987 by William E. Lorensen and Harvey E. Cline [3], which produces a polygonal representation of isosurface out of given voxel data. The algorithm takes scalar volume data sampled at a grid as input. The isosurface of a given isovalue is then constructed. Eight voxels are connected to each other to form a cube. The cube moves (marches) through the data and produces the triangles that represent the isosurface inside each cube location. Values below and above the isovalue are considered to be on different half-spaces of the isosurface. Each vertex and each edge inside the cube are numbered as shown in Figure 2.13 and Figure 2.14 respectively. Chernyaev [13] states, that there are 2^8 cases for the surface to intersect the cube, because there are 8 vertices of the cube and for each vertex there are two possible values, higher or the same as the isovalue and lower. Because there are eight vertices in the cube and each one can be zero or one, the data used for the triangle calculation can be saved in one byte. These cases can be rearranged into 15 unique configurations (see 2.15).

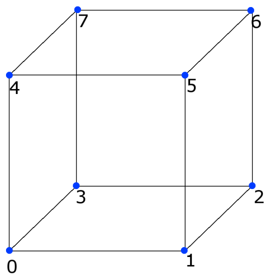


Figure 2.13: Configuration of vertices in marching cubes (reproduced from [14])

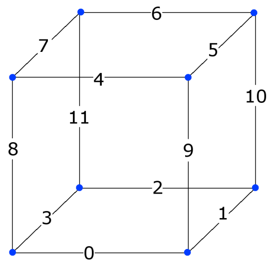


Figure 2.14: Configuration of edges in marching cubes (reproduced from [14])

One problem of the Marching Cubes algorithm is, that some configurations are ambiguous, creating holes in the resulting isosurface. This behaviour is illustrated in Figure 2.16. Chernyaev [13] states that the holes are occurring if two cubes, which have complement cases are aligned side by side. Both cubes produce different triangles and do not care about what the neighbour cube calculates. This behaviour can be fixed by letting the cubes "know" what the neighbour cube does and close the holes. In this bachelor thesis these ambiguous cases did not occur and therefore the fixing has not been implemented, because it is not necessary and would slow down the calculation.

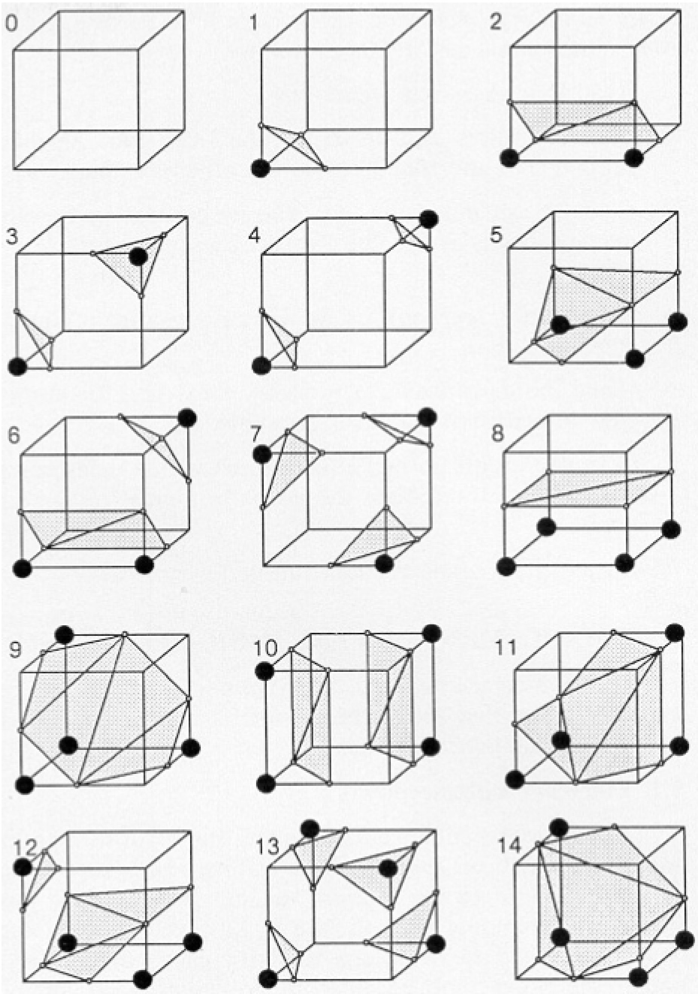


Figure 2.15: 15 unique configurations inside the marching cube [3]

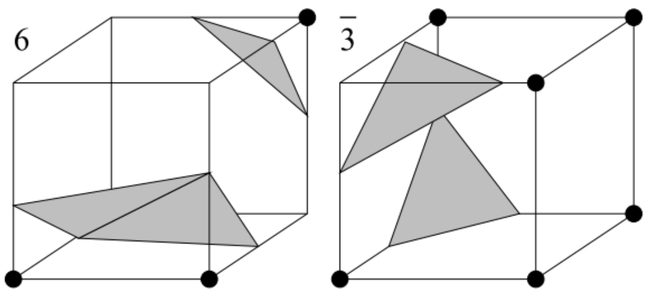


Figure 2.16: Cubes sharing faces with different triangulation [13]

Marching Tetrahedra

The Marching Tetrahedra algorithm, firstly announced by Doi und Koide [4] in 1991 works with tetrahedral cells instead of cubes. According to Treece [15] Marching Tetrahedra is a variation of Marching Cubes that solves the ambiguity problem of the Marching Cubes algorithm. When using Marching Cubes there are 256 possibilities for the isosurface to cross the cube, in relation to that Marching Tetrahedra have only 16 possible triangulations that can be reduced to three by symmetry. These three cases are illustrated in Figure 2.17

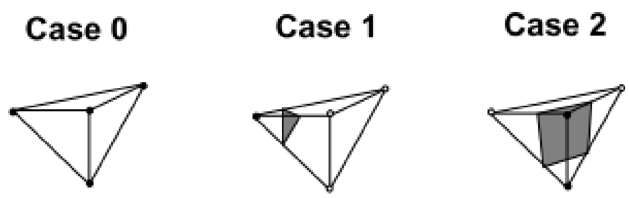


Figure 2.17: Three cases of Marching Tetrahedra [16]

Marching Tetrahedra uses 5 tetrahedrons per cube that spans over eight voxels. The tetrahedra can be aligned in severeal different ways in the given cube. Carneiro [16] states that it is important that the tetrahedra of different cubes share the same faces and edges. If every cube is filled up in the same way, the tetrahedra would not do so. If the tetrahedrons are aligned in a "zig-zag" manner, like stated by Carneiro [16], the given problem does not occur. The alignment is shown in the Figure 2.18

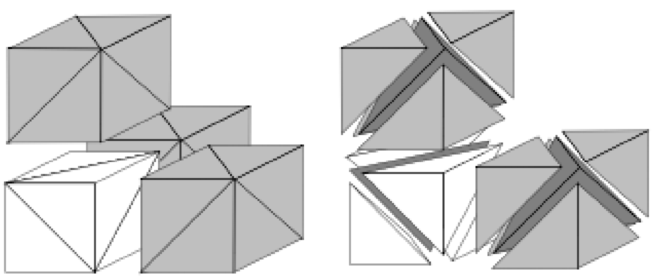


Figure 2.18: Zig Zag alignment of Marching Tetrahedra [16]

Carneiro [16] states that Marching Tetrahedra are simpler to implement, because of the smaller number of cases. It does not have the ambiguity problem of Marching Cubes and produces a more detailed representation of the surface. One problem of this algorithm is that it produces a huge number of triangles which may lead to problems in the data processing after the algorithm.

Comparison of polygonization algorithms

In this bachelor topic it has been decided to implement the Marching Cubes algorithm, because of the following arguments. The data is classified before calculating the isosurface, by a thresholding process and a connected component analysis. The analysis works with a six connectivity that means all six neighbours of each vertex are checked, if there is a connection. The data only consists of one connected component before calculating the isosurface. Marching Tetrahedra can produce up to 50 % more triangles which leads to a more detailed surface. This fact is not relevant at this thesis, because the surface data is smoothed before printing. The additional details, brought by Marching tetrahedra are irrelevant. Looking at this facts and at the results Marching Cubes is a good choice for this topic.

2.4 Data formats for 3D printers

Vatani [17] states that the stereo lithography (STL) file format is widely used in industry. The STL file describes a triangle mesh that represents the surface of the modelled object. STL files are widely used in the rapid prototyping industry, because computation of the slices, which are needed for these technologies, out of the STL is easy, fast and accurate. According to Grimm [18] the STL file is not proprietary and designed as the standard interface between CAD and rapid prototyping systems like 3D printers. According to Venuvino [19] the STL format has been invented 1989 by 3D system and the specification is shown in the following text part.

```
solid name
    facet normal n1 n2 n3
      outer loop
        vertex p1x p1y p1z
        vertex p2x p2y p2z
        vertex p3x p3y p3z
      endloop
    endfacet
endsolid name
```

Each printer has a different setup and parts, which have to be controlled in order to print an object. STL file format does not handle this task. Instead, the g-code file format is used that specifies exact printing instructions in the form of assembler language. This file format allows full hardware control of the 3D printer. However, the g-code is printer-specific and as such is used to convert the surface information in STL file into exact printing instructions for known 3D printer.

2.5 Smoothing before printing

After the surface extraction the 3D model can be smoothed. Blender is a well known 3D modelling software and therefore it is used for this task. In Blender there are different operators, which can be used for smoothing the object. The corrective smooth modifier smooths deformations like bumps, in order to reduce the number of distorted areas. There are two most important factors to define, named the smoothing amount and the number of smoothing iterations. The smoothing amount is given in a range between 0 and 1, the higher the value, the stronger the effect. A higher repetition number means that there are more distortions from the original model, which can have a good effect on the output but should be controlled. Another given parameter, which can be defined, is the smooth type. There are two options, one is the "Simple" and the other option is more complex and is called "Length Weight". Blender is an open source program therefore the sources are free to download from www.blender.com. In the source code of the corrective smooth modifier the two smoothing types are described. The "Simple" mode is an implementation of a weighted average smoothing. The surrounding vertices of the vertex of interest are weighted with the smoothing amount and the average is calculated. This value is taken for the vertex of interest. The "Length Weight" mode does not only weight the surrounding vertices with lambda, it also takes the distance of how fare the surrounding vertices are away into account. The bigger the distance to the surrounding vertex, the smaller the influence on the vertex of interest. This method delivers better results. The parameters which have been used to produce the illustrations are the following: the smoothing amount is 0.8, 80 smoothing iterations. Figure 2.19 shows the original surface data, Figure 2.20 illustrates the fetus data after smoothing with "Simple" smooth type. It can be seen that the bumps from the original data are gone, but the face looks like it is swollen and details are missing when comparing to Figure 2.21 where the "Length Weight" smoothing type has been used. Looking at this illustration the nose and the mouth seem to have much more details.

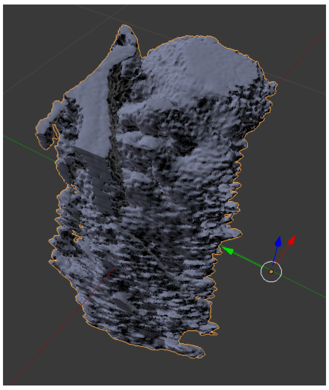


Figure 2.19: Original surface data without smoothing

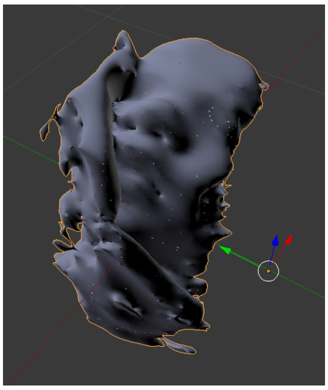


Figure 2.20: Corrective smoothing with "Simple" smoothing type

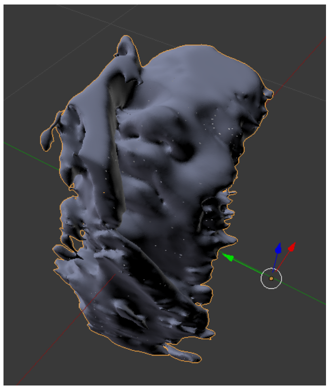


Figure 2.21: Corrective smoothing with "Length Weight" smoothing type

2.6 3D printing technology

The 3D printer used in this bachelor thesis works with a technology called fused deposition modelling. Gibson [5] states in his book that it works by pre-heating material in a chamber till the temperature where it melts in order to flow through a delivery system. This printing method uses a plastic filament, either ABS (Acrylnitril-Butadien-Styrol) or PLA (Polylactide), which is available in different diameters, like 1.75 or 3 mm. The core part of 3D printers is the extruder, which supports the material. It consists of a stepper motor. The extrusion of filament works by combining different gears and bearings. The material is then going through the hot end. There the filament melts and is squeezed through the nozzle. The temperature of the nozzle has to be regulated. The nozzle has a specific diameter typically 0.4 mm. This allows the printer to make very precise models. The scheme of the extruder is given in Figure 2.22. 3D printers need three degrees of freedom for printing 3D models. The different degrees are illustrated by the red arrows in the Figure 2.23.

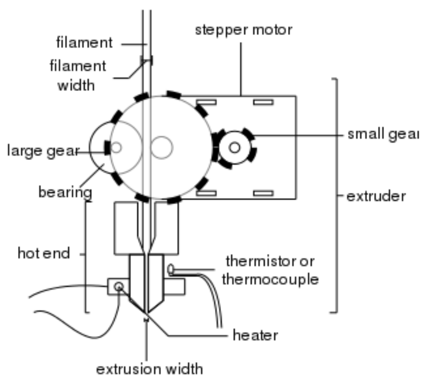


Figure 2.22: Extruder Details [20]

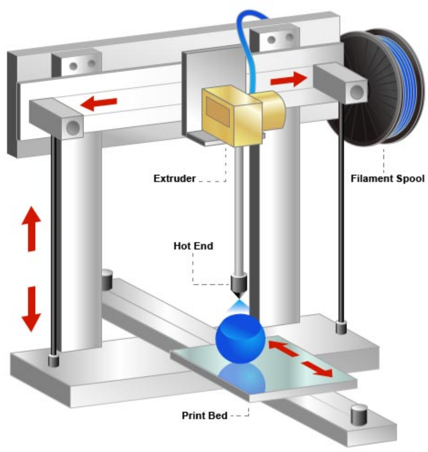


Figure 2.23: 3D printer diagram [21]

The printing process starts by placing the material onto the bed. After each printed layer, the printer proceeds by extruding the next layer on top of the printed one. The fabricated object is printed layer by layer. Each layer has to cool down before the next layer can be printed on. The printer has to be very precise in order to place every layer exactly above the before printed one. The layer thickness depends on the nozzle temperature and the diameter of the nozzle. If the printer is not calibrated, the printing result will be distorted. The printer cannot print in the mid-air. So if the fabricated model consists of floating parts, the printer has to print support structures. The support structures are removable after the printing process. Slicing software generates support structures automatically. In Figure 2.24 the support structure and the product, after the removal of the support structures is illustrated.

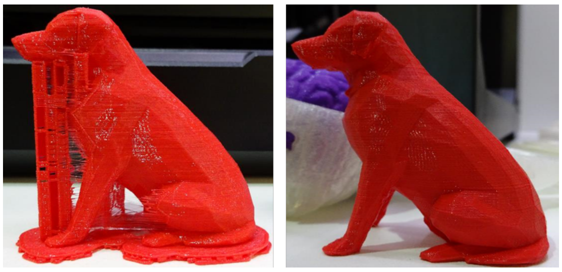


Figure 2.24: Before and after support structure removal [22]

The printing result depends on the temperature of the nozzle, the filament extrusion speed, the temperature of the printing bed, on the calmness of the air in the printing room and other parameters. Some printers, which are able to print in a nano scale need sterile and dust-free rooms, because they operate at the same size as dust particles. The 3D printing technology is not fully developed yet and therefore the deviation of these parameters has a huge impact on the printing quality.

2.7 Smoothing after printing

The surface of a 3D printed object is often very rough, because the print is built up by layers. The printer has a limited printing precision, like 0,1 mm per layer and therefore the finished print has very little steps, which can be seen and sensed. There are different methods how to get a smooth surface. One method is using sandpaper to sand down the little rills, until the surface is smooth [23]. The fabricated product has to be sanded evenly at all spots, otherwise it gets distorted. Figure 2.25 shows a printed model before sanding and Figure 2.26 afterwards.

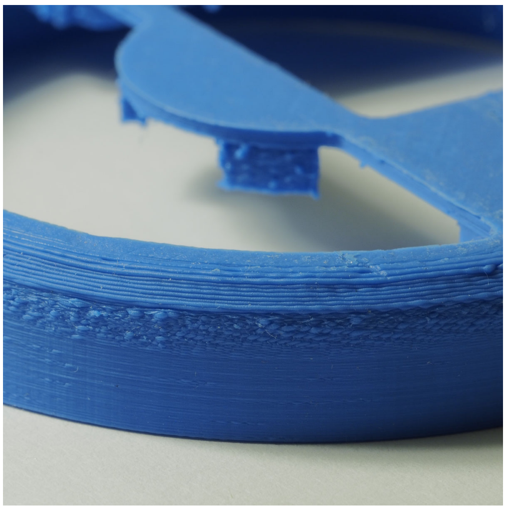


Figure 2.25: Before sanding [24]

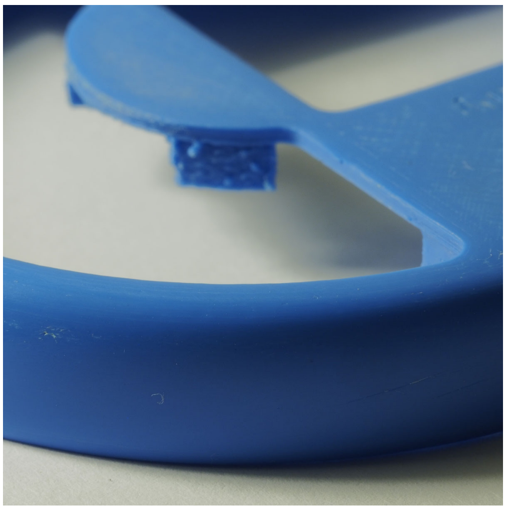


Figure 2.26: After sanding [24]

The fabricated objects can also be smoothed by using acetone vapors [25]. Liquid acetone applied to ABS printed products would melt it and deform it, but the vapour which is produced by heating the chemical can be used in order to smooth the surface of the printed products. The result is shown in Figure 2.27. The left owl object is before and the right one is after the procedure.

Figure 2.27: Left owl object is before and the right one after the usage of acetone vapour [26]

CHAPTER 3

Implementation details

This thesis focuses on the Marching Cubes algorithm in order to make volume data printable. Therefore the data does not have to be visualized, so we decided to make the computation on the central processing unit (CPU) of the computer. Marching Cubes algorithm is computationally expensive. In order to improve the performance of our software, the parallelization is done by QThreads. The Qt engine is also used for the graphical user interface, so it was near to use the supplied threading engine as well.

In this thesis the ambiguous cases of the Marching Cubes do not occur, which means the cubes are independent. Each cube can calculate its triangles on its own and does not have to know anything about its neighbour cube. Looking at this fact the volume data is separated by splitting it horizontally. The number of slices depends on the number of used threads, in our case we use eight. We tried out different settings and the best runtime has been reached by using eight threads, but this number could be dependent on which CPU the computer has. The Marching Cubes algorithm works with a lookup table, which is used from a project of Lewiner [14]. The threads have to be synchronised as well. In our case the synchronisation is very simple. As mentioned before the cubes are independent, so we are able to take the STL code from each cube directly and sum it up to build the final output. We do not have to implement a special treatment for the splitting areas.

The mentioned classification part delivers a three dimensional mask array where there are zero values, if the information of the voxel behind it is not relevant and the value one, if it is. The Marching Cubes algorithm should only calculate the surface of the voxels, which are relevant. If the algorithm only takes the mask values and calculates the surface it would be very cornered. In order to get a smooth surface the triangle points have to be calculated by using linear interpolation of the real voxel values. Looking at a cube there is for example one vertex inside the fetus and one outside, on the same edge. As an example one vertex has the value 180 and the other one has the value 10 and the threshold value is 170. The point of the triangle will be really close to the vertex with the value 180. If only the mask values would be taken, the surface would always cross in the middle of the edge.

The GUI consists of a menu, a number selection field, a check box, a progress bar and three buttons. The buttons are used to load the volume, save the data for finding the best threshold value and for carrying out the main task, named the Marching Cubes algorithm. The design can be seen in Figure 3.1. While the program is running the progress bar shows how much work has already been done in percent, as Figure 3.2 shows. The number selection field is used for the threshold. The check box determines whether the program performs connected component analysis and removes connected components except the largest one or not.

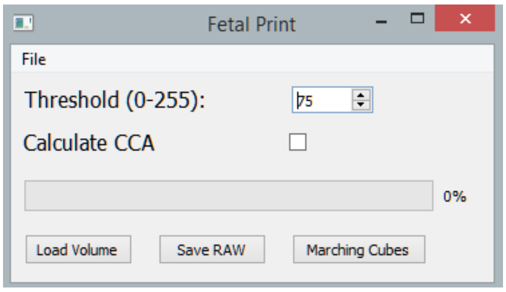


Figure 3.1: Graphical user interface start

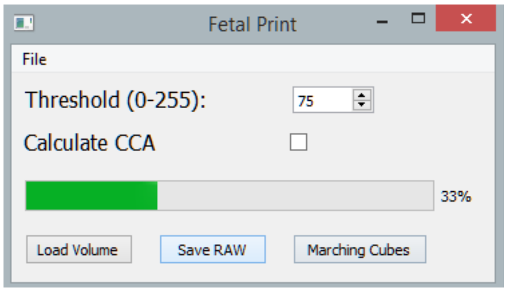


Figure 3.2: Graphical user interface after loading

CHAPTER 4

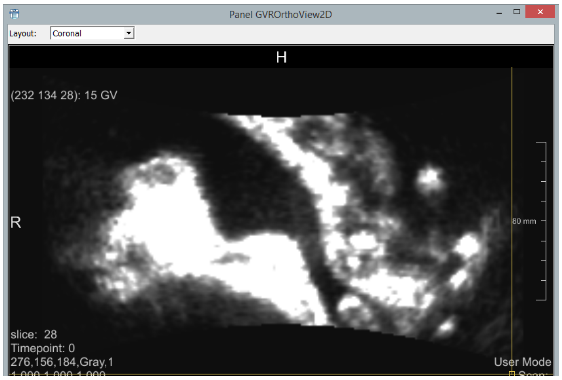
Results

This bachelor thesis aims to fabricate fetus models, therefore, the best way to present the results is actual photographic images of the fabricated models. Four datasets from ultrasonic examinations are used. Each dataset is presented with four images that show:

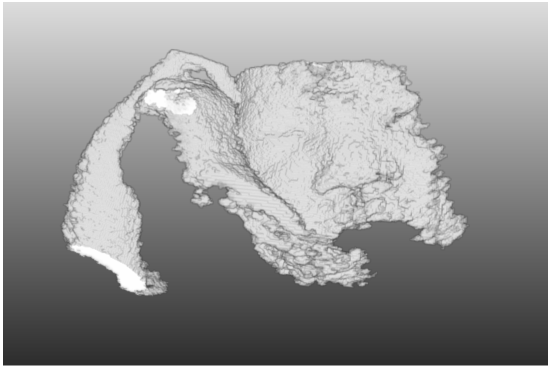
- 2D slice view of the original volume data from MeVisLab
- 3D rendering of the fetus in MeVisLab
- 3D rendering of the skin surface of the fetus in Cura prior to the printing
- Photographic image of the fabricated fetus model

The fetus model print solution works very well. It can be seen clearly that the visualisation in Cura is very similar to the fabricated model. The used 3D printer is a not that costly do-it-yourself kit. Therefore the quality of the fabricated model is not the best. Unfortunately no commercial printers were available to fabricate the models. The printing used to take very much time and it was a little difficult to break out the supporting material. A better solution would be to use a water-soluble support material which could be dissolved after the printing and does not have to be removed manually.

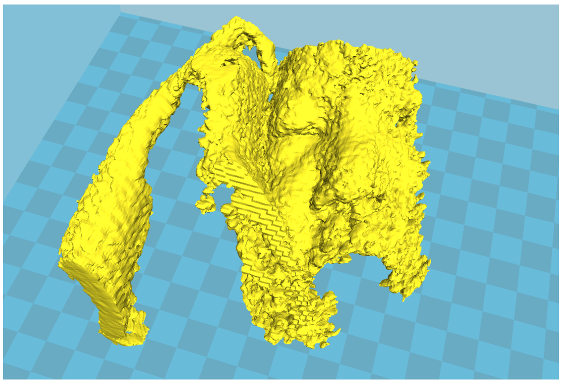
Dataset #1 shows both arms, as well as the nose and the closed eyes of the fetus. The model has not been smoothed for printing and therefore is very rough.



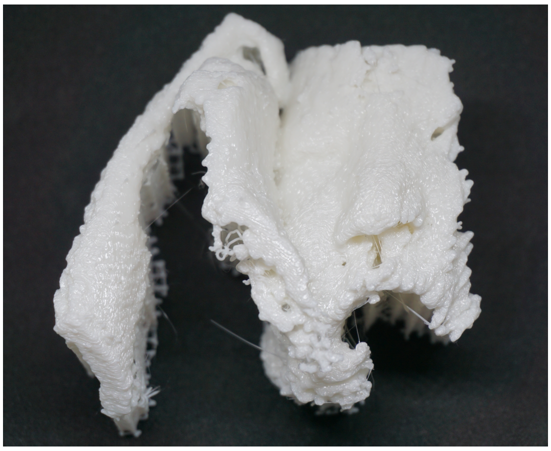
(a) slice view in MeVisLab



(b) rendering in MeVisLab



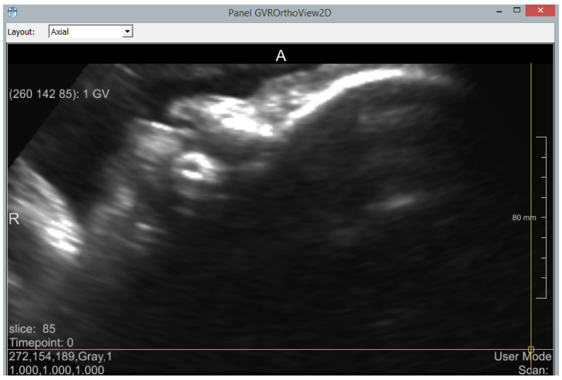
(c) visualisation in Cura



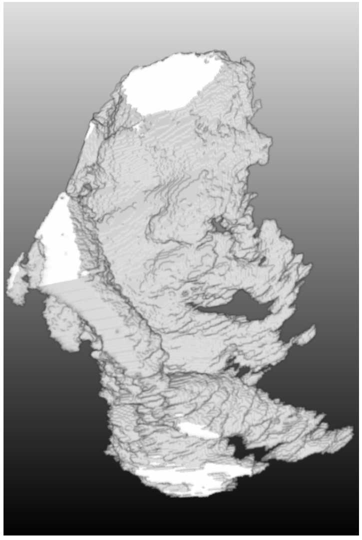
(d) printed object

Figure 4.1: dataset #1

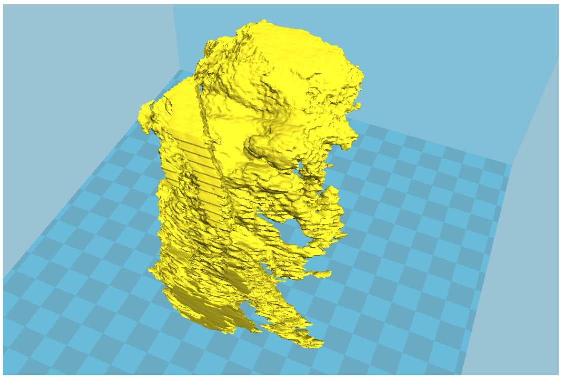
Dataset #2 is focused on the face of the fetus and is also not smoothed before printing. The data quality is not very good, therefore the mouth and the left eye is not included entirely.



(a) slice view in MeVisLab



(b) rendering in MeVisLab

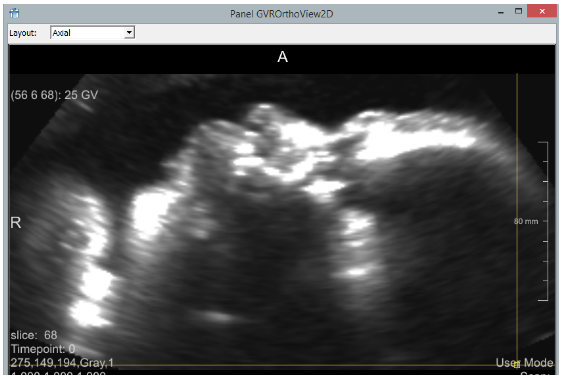


(c) visualisation in Cura

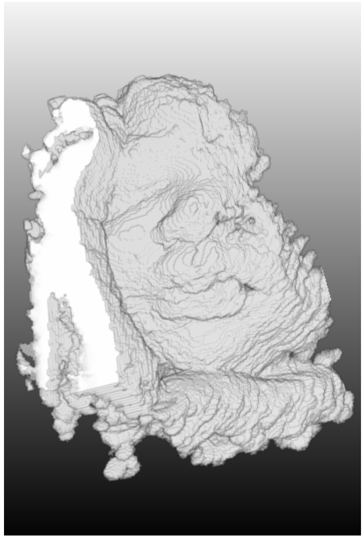
(d) printed object

Figure 4.2: dataset #2

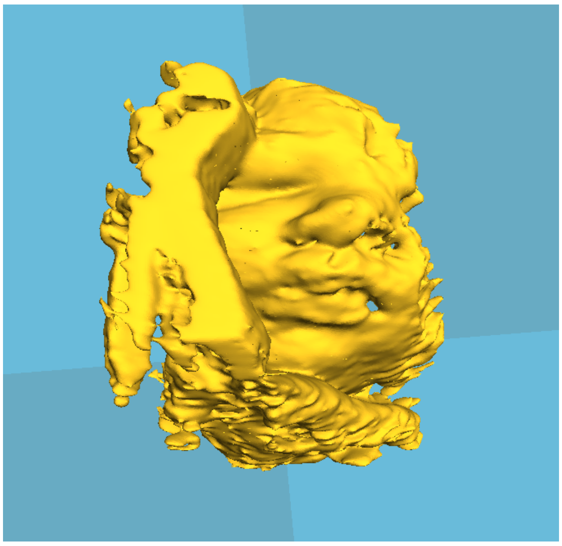
Dataset #3 is also focused on the face of the fetus and is smoothed before printing. The doctor used a different angle in the ultrasound investigation and therefore the mouth and the nose is warped. Distortions like that can also occur, when the fetus is moving during the investigation.



(a) slice view in MeVisLab



(b) rendering in MeVisLab

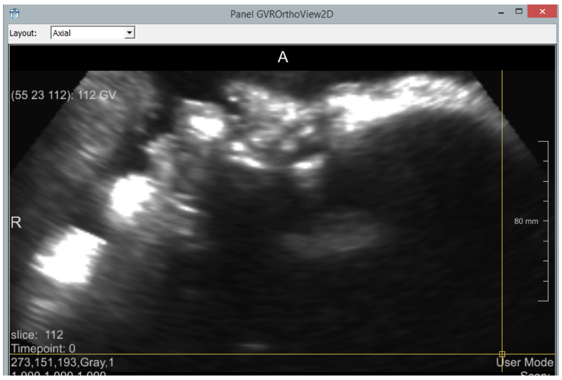


(c) visualisation in Cura

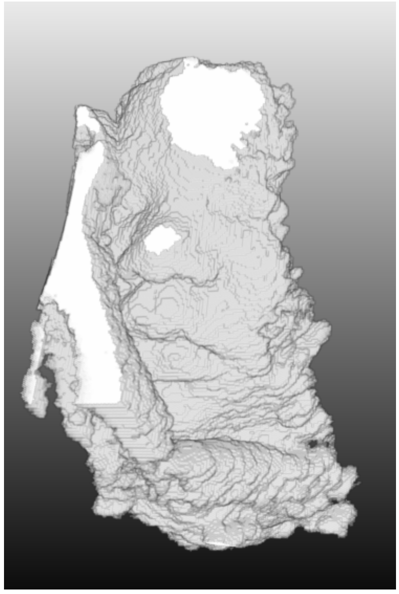
(d) printed object

Figure 4.3: dataset #3

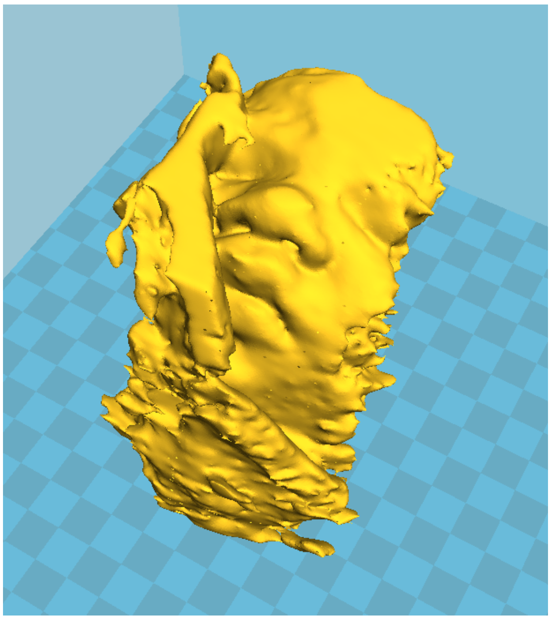
Dataset #4 is also focused on the face of the fetus and is smoothed before printing. The face can be seen very clearly, including the nose, the mouth and the closed eyes as well. This dataset has the best quality because it is very good defined.



(a) slice view in MeVisLab



(b) rendering in MeVisLab



(c) visualisation in Cura

(d) printed object

Figure 4.4: dataset #4

CHAPTER 5

Conclusion

The goal of this bachelor topic is to print the given data from a 3D ultrasound investigation. In the result section the fabricated models are illustrated and they have a high similarity to the voxel data visualisation. When people are looking at the models, they are able to perceive the nose, the mouth and the closed eyes. The surface data is smoothed before printing in order to get very smooth models, which do not have to be reworked. The models could be used when parents are not able to see the 3D ultrasound images for example if they are blind or have very bad eyesight.

There is some part of the solution which could be refined in order to improve the usability. The thresholding is done by the user and needs two different programs. It would be better, if the visualisation of the data would be included in the fetus model print program, so the user could adjust the threshold value "on-the-fly" without having to switch between programs. Another topic, to make the solution work in a productive environment would be to combine different volume datasets in order to make a model, which displays the whole fetus and not only the face of it. This task is not easy going because the fetus is moving during the investigation and the different angles used for the ultrasound investigation make it difficult to put the datasets together automatically. Cura, the program used for calculating the g-code is an open source program. It would be a good solution, if the whole process described in this thesis would be integrated into Cura as an add-on. The whole process would only need one program and not four.

There are also several future topics which could be linked to this work. One example would be 3D printing for medical intervention planning. Organ models can be extracted from MRI and CT data and fabricated. The surgeons than would gain a possibility to plan the intervention in a real world simulation. The task could also be carried out with an adapted version of the solution given in this bachelor topic. Parents maybe want to know how the child will look like, after it is born. So one could possibly try to calculate how the child will develop and print a version of how it looks like after it is born. There are lots of possibilities for future work basing on this bachelor topic.

List of Figures

List of Algorithms

Bibliography

[1] P. Fastermann, *3D-Drucken: Wie die generative Fertigungstechnik funktioniert.* Springer, 2014.

[2] V. Siebert, "Blind mother-to-be gets the chance to 'see' her unborn son for the very first time after doctors surprise her with a 3D printout of his ultrasound," 2015. [Online]. Available: http://www.dailymail.co.uk/femail/article-3070976/Blind-mother-gets-chance-unborn-son-time-doctors-surprise-3D-printout-ultrasound.html [Accessed: 2016-06-05]

[3] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *Proceedings of the 14th annual conference on Computer graphics and interactive techniques - SIGGRAPH '87*, vol. 21, no. 4, pp. 163–169, 1987.

[4] A. Doi and A. Koide, "An Efficient Method of Triangulating Equi-Valued Surfaces by Using Tetrahedral Cells," *IEICE Transactions*, vol. E74, no. 1, pp. 214–224, 1991.

[5] I. Gibson, D. W. Rosen, and B. Stucker, *Additive manufacturing technologies: Rapid prototyping to direct digital manufacturing.* Boston, MA: Springer US, 2010.

[6] J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics: Principles and Practice, 2nd Ed. in {C}.* Addison-Wesley Longman Publishing Co., 1996.

[7] S. Ambardar and M. Singhal, "A Review and Comparative Study of De-noising Filters in Ultrasound Imaging," *International Journal of Emerging Technnology and Advanced Engineering*, vol. 4, no. 8, pp. 824–831, 2014.

[8] J. C. Dainty, *Laser Speckle and Related Phenomena.* Springer, 1975.

[9] M. Luqman Bin Muhd Zain, I. Elamvazuthi, and M. Begam, "Comparative analysis of filtering techniques for ultrasound images of bone fracture," in *2010 International Conference on Intelligent and Advanced Systems.* IEEE, jun 2010, pp. 1–4.

[10] E. Nadernejad and M. Karami, "Despeckle filtering in medical ultrasound imaging," *Contemporary Engineering Sciences*, vol. 2, pp. 17–36, 2009.

[11] Li Rui, Sun Zhuoxin, and Zhang Cishen, "Adaptive filter for speckle reduction with feature preservation in medical ultrasound images," in *2008 10th International Conference on Control, Automation, Robotics and Vision.* IEEE, dec 2008, pp. 1787–1792.

[12] MeVis Medical Solution, "MeVisLab," 2016. [Online]. Available: http://www.mevislab.de/ [Accessed: 2016-06-05]

[13] E. Chernyaev, "Marching cubes 33: Construction of topologically correct isosurfaces," *Institute for High Energy Physics, Moscow, Russia,*, pp. 1–8, 1995.

[14] T. Lewiner, H. Lopes, A. W. Vieira, and G. Tavares, "Efficient Implementation of Marching Cubes' Cases with Topological Guarantees," *Journal of Graphics Tools*, vol. 8, no. 2, pp. 1–15, 2003.

[15] G. Treece, R. Prager, and A. Gee, "Summary for Policymakers," in *Climate Change 2013 - The Physical Science Basis*, Intergovernmental Panel on Climate Change, Ed. Cambridge: Cambridge University Press, 1998, vol. 1, pp. 1–30.

[16] B. Carneiro, C. Silva, and A. Kaufman, "Tetra-cubes: an algorithm to generate 3D isosurfaces based upon tetrahedra," *Proceedings of IX Brazilian symposium on computer, graphics, image processing and vision (SIBGRAPI)*, pp. 205–210, 1996.

[17] M. Vatani, A. Rahimi, and F. Brazandeh, "An enhanced slicing algorithm using nearest distance analysis for layer manufacturing," *Proceeding of World Academy of Science, Engineering and Technology*, vol. 3, no. 25, pp. 721–726, 2009.

[18] T. Grimm, *User's guide to rapid prototyping.* Dearborn, Mich. : Society of Manufacturing Engineeers, 2004.

[19] P. K. Venuvinod and W. Ma, *Rapid Prototyping.* Boston, MA: Springer US, 2004.

[20] T. Halterman, "ScrapRap," 2015. [Online]. Available: https://3dprint.com/70287/build-3d-printer-cheap/ [Accessed: 2016-06-05]

[21] Linkyo Insights, "Beginner's Guide to 3D Printing," 2013. [Online]. Available: http://blog.supermediastore.com/article/nitty-gritty-3d-printing [Accessed: 2016-06-05]

[22] PatentYogi, "Near instantaneous 3D printing technology from Disney," 2016. [Online]. Available: http://patentyogi.com/american-inventor/near-instantaneous-3d-printing-technology-from-disney/ [Accessed: 2016-06-05]

[23] A. K. France, *Make: 3D Printing: The Essential Guide to 3D Printers.* Maker Media, 2013.

[24] M. Matisons, "Sand Your Way to Smoother 3D Printed Parts," 2016. [Online]. Available: https://3dprint.com/113714/sandpaper-3d-printed-parts/ [Accessed: 27.07.2016]

[25] J. Horvath and R. Cameron, "3D Printing," in *The New Shop Class.* Berkeley, CA: Apress, 2015, pp. 31–46.

[26] Sink Hacks, "Building an Acetone Vapor Bath for Smoothing 3D-Printed Parts," 2016. [Online]. Available: www.sinkhacks.com [Accessed: 2016-06-05]