PALESTINE POLYTECHNIC UNIVERSITY

DEANSHIP OF GRADUATE STUDIES AND SCIENTIFIC RESEARCH

MASTER OF INFORMATICS

Efficient Generation of Transfer Function for Direct Volume Rendering

Submitted by

Afnan M A Salah

A thesis submitted in Partial fulfillment of the Requirements for the Degree Master of Informatics

May 2024

APPROVAL OF THE PPU BOARD

Date: 04/05/2024

The undersigned hereby certify that they have read and recommended to the Deanship of Graduate Studies and Scientific Research at Palestine Polytechnic University the acceptance of a thesis entitled:

Efficient Generation of Transfer Function for Direct Volume Rendering

Submitted by: **Afnan M A Salah**

In Partial fulfillment of the Requirements for the Degree of Master of Informatics.

Graduate Advisory committee:

Dr. Zein Salah,

Assist. Prof. of Computer Science, Palestine Polytechnic University

Signature: ..... Date: .....

Dr. Alaa Halawani,

Assoc. Prof. of Computer Engineering, Palestine Polytechnic University

Signature: ..... Date: .....

Dr. Ahmad Ewais,

Assoc. Prof. of Computer Science, Arab American University Jenin

Signature: ..... Date: .....

Thesis Approved

Dr. Nafez Nasereddin

Dean of Graduate Studies and Scientific Research

Signature: ..... Date: .....

CERTIFICATE

This is to certify that the thesis entitled **"Efficient Generation of Transfer Function for Direct Volume Rendering"**, submitted by **Afnan M A Salah** to **Palestine Polytechnic University**, is a record of bona fide research work under my supervision and guidance, and I consider it worthy of consideration for the award of the degree of *Master of Informatics* of the Institute.

Dr. Zein Salah

Signature: .....

Date: .....

DECLARATION

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree at Palestine Polytechnic University or at any other educational institute, except where due acknowledgment has been made in the thesis. Any contribution made to the research by others, with whom I have worked at the college or elsewhere, is explicitly acknowledged in the thesis. I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Afnan M A Salah

Signature: .....

Date: .....

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for the master degree in Informatics at Palestine Polytechnic University, I agree that the library shall make it available to borrowers under the rules of the library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made. Permission for extensive quotation from, reproduction, or publication of this thesis may be granted by my main supervisor, or in his absence, by the Dean of Graduate Studies and Scientific Research when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Afnan M A salah

Signature: .....

Date: .....

Dedicated to,

To my deceased father, and lovely mother, may God protect her , and to my dear family.

To those who gave their lives so that we could live in dignity.

ACKNOWLEDGMENTS

I would like to thank mom, the source of inspiration, who gave me strength when I thought of giving up. My husband and my beloved family, without their endless love, support, and patience, I would not have been able to achieve this. Also, my brothers, sisters, and friends, for their advice, encouragement. Finally, I would like to thank my supervisor Dr. Zein Salah for his help and support throughout the work on this thesis.

Abstract

The process of transfer function generation is a crucial step in the direct volume rendering pipeline. The transfer function is responsible for setting visual properties such as color and transparency for each voxel in the volumetric data set. Therefore, it is considered the most important part of the direct volumetric rendering process, as it is a complex process and takes a long time. That is why the transfer function is the process that potentially determines the efficiency of the volume rendering process as a whole.

In this thesis, we proposed an automatic design of a transfer function based on the similarity of features of volumetric data, where the features of volumetric data are extracted through the similarity between iso-surfaces. Then, we classified these features through the affinity propagation algorithm to automatically extract the optimal number of clusters that best reflect these features.

The efficiency of the proposed system is demonstrated by its accuracy in exploring the features of volumetric data and its ability to classify them automatically without the need of user intervention or pre-define the number of clusters.

الملخص

يعتبر اقتران التحويل (ترانسفير فنكشن) خطوة بالغة الأهمية في عملية العرض ثلاثي الأبعاد للبيانات الحجمية. وظيفة اقتران التحويل تقوم على تعيين خصائص بصرية كاللون والشفافية لكل نقكة في الصورة المراد عرضها. لذا تعتبر هذه الخطوة الأكثر أهمية كونها تحدد ةكفاءة عملية العرض ككل.

في هذه الأطروحة قمنا باقتراح طريقة لتصميم وحساب اقترانات التحويل بشكل أوتوماتيكي، يقوم بشكل مبدئي على ثماثل الخصائص في الصورة، حيث يتم استخراج هذه الخصائص اعتماداً على التماثل بين سطوح القيم المتساوية (آيزو سيرفيس). في خطوة لاحقة يتم تصنيف هذه الخصائص من خلال خوارزمية افينيتي بروباجيشن من أجل استخلاص العدد الأمثل من العتاقيد التي تعكس تلك الخصائص، وذلك بشكل تلقائي.

تتجلى كفاءة النظام المقترح من خلال دقته في استخلاص الخصائص الهامة من البيانات الصورية، وقدرته على تصنيفها بشكل تلقائي دون الحاجة إلى تدخل المستخدم او تحديد عدد العناقيد مسبقاً.

Table of Contents

Approval of The PPU Board i

Certificate ii

Declaration iii

Dedication v

Acknowledgements vi

Abstract vii

Arabic Abstract viii

Table of Contents ix

List of Figures xii

List of Tables xiv

Chapter 1 Introduction 1

1.1 Volume Visualization 2

1.2 Research Problem and Objectives 3

1.3 Contribution of this Thesis 4

1.4 Thesis Structure 5

Chapter 2 Background 7

2.1 Volume Data 8

2.2 Data Visualization 9

2.2.1 Data Visualization Pipeline 12

2.3 Volume Data Visualization 13

2.3.1 Decomposition Methods 14

2.3.2 Indirect Volume Rendering 15

Marching cubes . . . 16
2.3.3 Direct Volume Rendering . . . 17
Ray Casting . . . 17
2.4 Volume Classification . . . 19
2.4.1 Transfer Function Generation Methods . . . 21
Data driven methods . . . 21
Image driven methods . . . 21
2.4.2 Automation of Transfer Function . . . 22
2.5 Related Works . . . 22
2.5.1 Histogram-Based Methods . . . 23
2.5.2 Feature-Based Methods . . . 26
2.5.3 Graph-Based Methods . . . 28

Chapter 3 Data and Methods 29

3.1 Data Preparation . . . 30
3.2 Volume Classification . . . 31
3.2.1 Similarity Matrix . . . 31
3.2.2 Classification using Affinity Propagation . . . 34
3.3 Rendering . . . 36

Chapter 4 Experiment and Results 38

4.1 System Platform . . . 38
4.1.1 Implementation Details . . . 40
4.2 Tools . . . 41
4.2.1 Visualization Toolkit . . . 41
VTK Pipeline . . . 42
4.3 Results . . . 43
Testing Data . . . 43
Real Data . . . 43
4.3.1 Experiment 1 . . . 44
4.3.2 Experiment 2 . . . 53
4.4 Method Evaluations . . . 60

Chapter 5 Conclusions and Future works 64

5.1 Limitations and Challenges . . . 64

5.2 Conclusions 65
5.3 Future Works 66

References **67**

List of Figures

2.1 The pixel in a grid point of image. 8
2.2 The voxel and the cell in 3D grid. 9
2.3 Example of data visualization techniques. 10
2.4 Example of medical data visualization [34]. 11
2.5 The Visualization Pipeline. 12
2.6 The deference between indirect volume rendering and direct volume rendering[27]. 14
2.7 The concept of tiny cubes method [22]. 15
2.8 The 15 possible intersection cases in marching cube algorithm [35]. 17
2.9 Ray casting algorithm [7]. 18
2.10 The volume rendering pipeline. 18
2.11 Transfer function for volume rendering. 20
2.12 The concept of histogram. 23
2.13 The relationship between the gray value and gradient value, and the concept of gray-gradient histogram [15]. 24

3.1 The concept of isosurface similarity calculation. 32
3.2 The concept of octree partitioning [11]. 33

4.1 System Platform. 39
4.2 The VTK pipeline. 42
4.3 Samples of similarity matrix for an image of two isovalues. 44
4.4 Clustering result of two-values image similarity matrix. 45
4.5 Rendering result of two-value image. 46
4.6 Rendering result of foot Dataset. 47
4.7 The resulting classes of foot dataset. 48
4.8 Rendering result of Head MRI Dataset. 49
4.9 Inner and outer tissues of brain. 49
4.10 Different tissues of brain 50

4.11 Rendering result of a abdomen dataset. 51
4.12 Colon tissues rendering. 51
4.13 Different tissues of abdomen dataset. 52
4.14 Clustering results of a pre-defined four intensity values image. 53
4.15 Rendering results of a pre-defined four intensity values image. 54
4.16 Rendering result for the foot dataset. 55
4.17 Various clusters renderings for the foot dataset using the median value. . 56
4.18 Rendering result for the brain MRI dataset 57
4.19 Various clusters rendering for the foot data set using the median value. . 58
4.20 Rendering result for the abdomen dataset. 59
4.21 Various clusters renderings for the abdomen data set using the median value. 60
4.22 Rendering result of two different clusters of head MRI dataset. 61
4.23 Rendering result of two different clusters of abdomen dataset. 62

List of Tables

4.1 Performance of the pipeline , measured in terms of the computation time of each step, in the classification the minimum value has been used as a preference. . . . 63

4.2 Performance of the pipeline , measured in terms of the computation time of each step, in the classification the median value has been used as a preference. 63

1

Introduction

In general words, visualization refers to any way or technique of creating images, diagrams, animations or any representations of an object, situation, or set of information in order to make it more accessible and understandable. This way of visual imagery has been an efficient way of communicating ideas since the dawn of humanity. And these days, with the increase in data-driven world, there has become an urgent need to find more effective methods to view and understand data. This led to the emergence of the data visualization science.

Data visualization is a branch of science that is concerns with displaying data in graphical representations using visual elements like charts, maps, graphs and other methods, in order to offer insights into data which ease the understanding and analyzing the data. The main goal of data visualization is to help in understanding, analyzing, inter-

acting with, and gaining insight into data.

There are many fields the data comes from; some are computed and some are simulated. Such data can be three dimensional data like medical images such as computed tomography (CT) or magnetic resonance imaging (MRI) which stores physical measurements, often at a high resolution. This type of data needs to be presented in a graphical form in order to facilitate extracting interesting features and better to analyze and gain insights and inspections from it, thus leading to better diagnostics, decision making, treatment plannings, etc.

The process of visualizing three dimensional data into two dimensional graphical representations is called volume visualization, which is concerned with creating visual representations of the scalar data in three-dimensional space.

1.1 Volume Visualization

Volume visualization is a branch of science that is concerned with producing a two-dimensional graphical representation from a raw three-dimensional scientific datasets to improve the interpretation of data and to get insights into it.

There are two major types of volume visualization, indirect volume rendering (iso-surface rendering) and direct volume rendering (DVR) [8]. The difference between indirect and direct volume rendering is that the indirect volume rendering converts the volume data to an intermediate representation (polygonal surface) and subsequently renders it using some rendering methods [25][32]. On the other hand, direct volume rendering directly maps volume data to optical properties without employing any intermediate representations [25][32].

The major step in the direct volume rendering pipeline is the classification step. In this step, color and opacity values are assigned to each voxel in the data depending on its properties. This mapping is done by the so - called *Transfer Function* [32][25][22][17].

There are many types of classification that can be categorized according to different criteria. Some are automatic that generate a transfer function automatically and some are not. Some are data-driven that depend on the properties extracted from the data and some are image-driven that depend on input image slices.

A great amount of research and work focused on automatic and semi-automatic transfer function generation [24][29][17]. Sometimes it is desirable to allow the user to change in transfer function even if it is generated automatically. So the description of automatic or semi automatic depends on the ability of the method to generate the initial transfer function with or without the intervention of the user.

1.2 Research Problem and Objectives

A crucial step in the DVR is the generation of the transfer function which maps the scalar values of the volume dataset to optical properties like color and opacity. According to that, the design of the transfer function determines the efficiency of the whole rendering process. Since the generating of the transfer function is a non-intuitive and time-consuming task, the challenge is how to develop methods to efficiently generate the transfer function from the data properties such as intensity and gradient.

In this thesis, we implement a full volume rendering pipeline. Within this pipeline, a fully automatic classification approach is introduced. which is based on a combination of a similarity matrix classification algorithm and affinity propagation clustering.

1.3 Contribution of this Thesis

Similarity matrix has been used for the volume classification [18]. In essence, it represents the input to an IGM-histogram based classification algorithms. One restriction of this approach is the need to pre-define the number of clusters to be computed, since this algorithm uses a k-medoid method.

On the other hand, the affinity propagation algorithm has been used for classification [36], with the advantage that the number of clusters need not to be given as input. However, it was applied to classify IGM-Histograms.

In our thesis, we adapt a combination of both methods. Mainly, we initiate the classification process by the similarity matrix. This method is advantageous in the context of medical images, since it is based on the content of the image, and therefore, the borders between different tissue types are taken into consideration.

To further automate the process, we do not continue the classification using the K-medoid algorithm, but instead, we feed the similarity matrix into the affinity propagation algorithm to avoid having to define the number of clusters, which in most cases cannot be guessed initially.

In order to test and evaluate the previously described approach, the following activities have been carried out in the frame of this thesis:

- Implementation of the marching cube based similarity matrix algorithm with proper adaptation for the next pipeline steps.
- Implementation of the affinity propagation algorithm, and testing the effect of various options and parameters.

- For visual inspection of the classification, we developed a rendering environment that allows for testing various settings and inputs and generating proper interactive visualizations of input images.

1.4 Thesis Structure

The rest of this thesis is organized as follows: Chapter 2 presents the theoretical background of topics that are related to the thesis problem. Section 1 of this Chapter is concerned with defining the volume data and the differences between 2D and 3D data. Section 2 presents a definition of data visualization and depicts the visualization pipeline. Section 3 describes the volume data visualization and its techniques. Section 4 describes the process of volume classification and the methods of defining transfer functions. Finally, Section 5 shows a set of the related studies that are dealt with the main aspects of our thesis.

Chapter 3 shows the details of the proposed system in three sections. Section 1 describes the nature of the data used in the project. Section 2 gives the details of the volume classification process including the similarity matrix creation method, and classification algorithm and gives a detailed description of the affinity propagation algorithm. Section 3 shows the rendering details.

In Chapter 4, we highlight our experiments and results, including the system platform and implementation details in Section 1. In Section 2, the VTK tool that is used in this project is briefly described. Section 3 shows the experiment results of the used datasets. Section 4 concludes the Chapter with a discussion and evaluation of the proposed method.

Finally, in Chapter 5, a summary and conclusion of the thesis is presented, including the challenges and future work.

2

Background

In this chapter, an overview of theoretical background will be presented. As the subject under consideration deals with volume data, we start with a brief description of this type of data. We then explain the meaning of data visualization and scientific visualization and the main goals of scientific visualization. After that, the description of the visualization pipeline is presented. Then, we describe the volume visualization techniques and the algorithms related to each technique. Finally, the related works and previous studies are presented.

2.1 Volume Data

A digital image is a two-dimensional array of a finite number of elements called picture elements or *Pixels* [32]. Each element has a particular value at a particular location.

As shown in Figure 2.1 an image is defined as two-dimensional function F(x,y), where x and y are spatial coordination, and the amplitude of F is called the intensity. i.e the pixel value is the intensity or the gray value in the 2D image [27][8]. In the 3D datasets, the 3D image is a three-dimensional ordered set of volume elements called *voxels* [8].

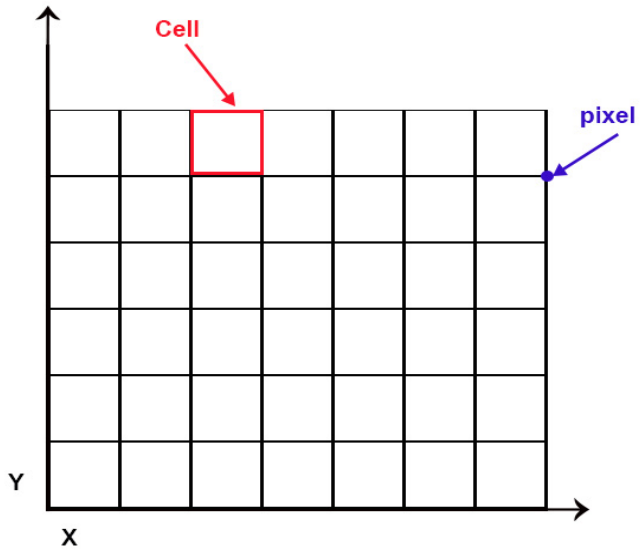


Figure 2.1: The pixel in a grid point of image.

In contrast to pixel, the voxel is usually determined by its position relative to other voxels rather than its value and its location [27].

Figure 2.2 shows the volume as a three dimensional ordered set of voxels.

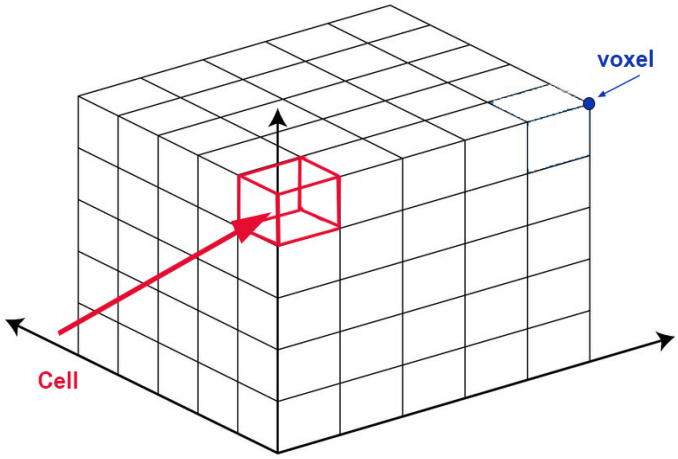


Figure 2.2: The voxel and the cell in 3D grid.

As shown in Figure 2.2, the cell in the Cartesian grid is defined by the volume falls between eight neighboring grid points. And the image in 3D can be defined as a function F(x,y,z), where x,y and z are the indices of the voxel.

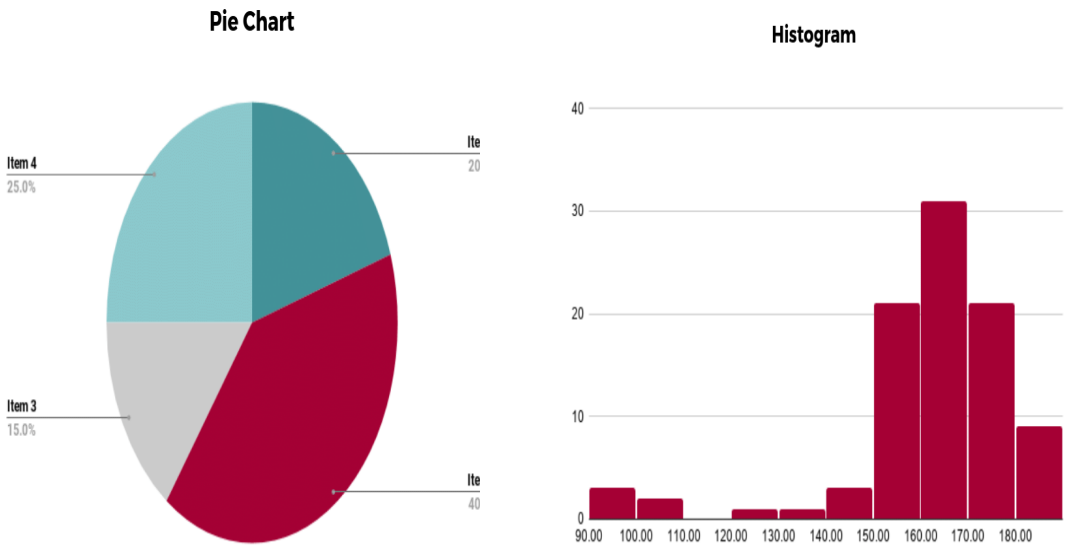
From another viewpoint, the 3D image can be seen as a stack of 2D images called slices, and can be defined as a function of F(x,y) within the slice and z is the index of the slice within the volume[27].

2.2 Data Visualization

Data visualization is a part of science that is concerned with the representation of the data in a visual form [32][19]. It contains a set of methods for displaying data graphically in some presentations that helps to understand and analyze data.

The input of data visualization comes from different areas and is usually described by a *data model*, which defines the nature of the data as well as its attributes and relations[9].

The common sources of data can be a direct measurement of physical quantities, a database of information system, or a result of software simulation. Accordingly, the applied visualization can be classified as either *information visualization* or *scientific visualization*[21]. In general, information visualization aims at facilitating a visual inspection of abstract data, which is usually non-physical and non spatial data, that might lead to, e.g., proving or disproving a hypothesis about data, or presenting a comparison of specific data attributes for some purpose [4]. Usually, different types of charts are utilized to serve different purposes. Common examples are pie charts, bar charts, histograms, scatter plots, etc. An example is shown in Figure 2.3.



(a) Pie chart (b) Histogram

Figure 2.3: Example of data visualization techniques.

On the other hand, scientific visualization is a branch of data visualization that is concerned with producing a graphical representation of raw scientific data, usually physically based data, e.g. medical images, to improve interpretations of large datasets

and gain insights into the data [19].

The idea of scientific visualization is that it generates graphical images from data. These images are resulted by rendering models that are extracted from the raw data. For this purpose, several techniques from computer vision, image processing, computer graphics, and animation can be adopted, depending on the nature of the data and the targeted application.

One major field of scientific visualization is medical visualization [25]. This is largely motivated by the developments in medical imaging methods and scanners. The need for more effective visualizations of the data increases for the purposes of disease diagnosis, treatments planes, educations and many other purposes.

Medical images are one of the best examples of volumetric data [27], which is considered as a stack of aligned images or slices with the same resolution and adjacent position. An example of medical data visualization is shown in Figure 2.4.

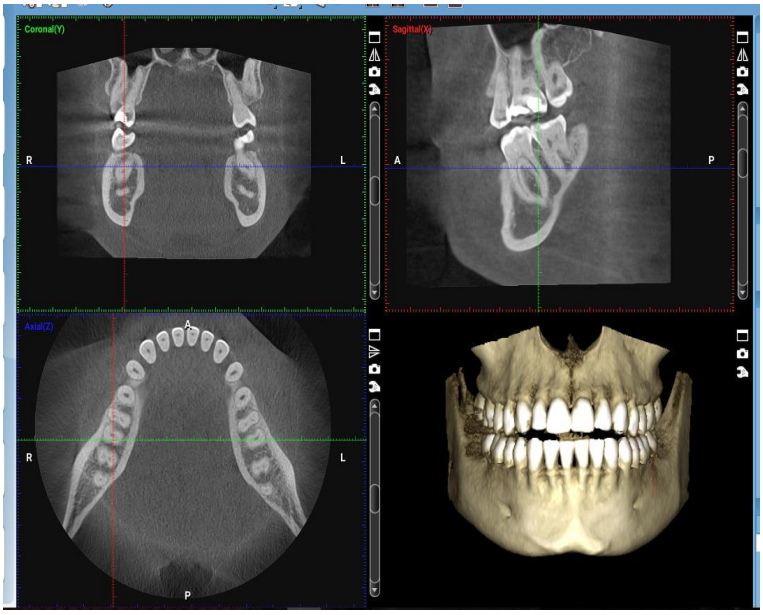


Figure 2.4: Example of medical data visualization [34].

The main goals of visualization in general can be summarized as [25]:

- Explore data and information and enhance the understanding of the data and gaining new insights.
- Representing a scientific phenomena and features.
- Control and making decisions of measurements and simulations.

2.2.1 Data Visualization Pipeline

The visualization process consists of several stages [32][25], The input of each stage is the output of the previous stage and each stage is modeled by a specific data transformation operation. This data flow is called the *visualization pipeline*.

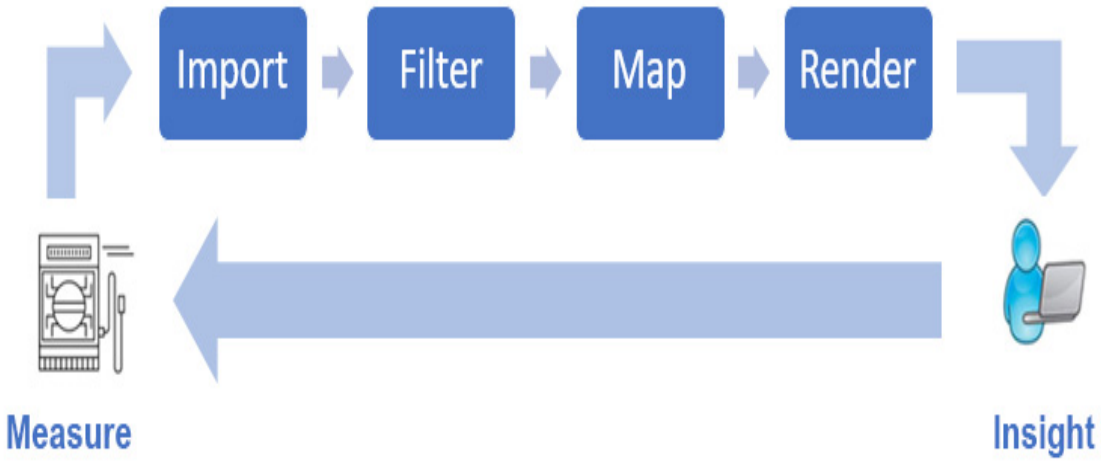


Figure 2.5: The Visualization Pipeline.

As shown in Figure 2.5, the visualization pipeline often consists of the following stages [32]:

- **Import data**: By choosing the original information that was previously measured

or simulated and representing it in a suitable data format.

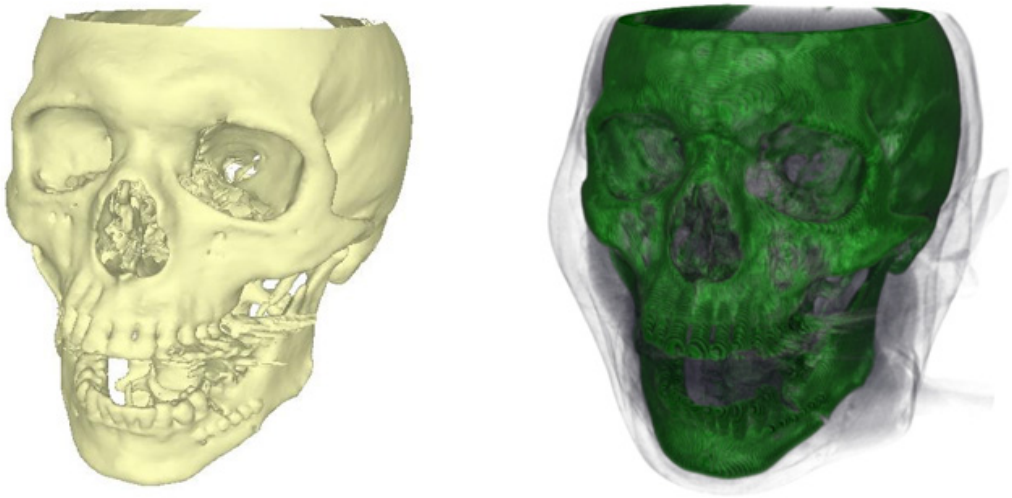
- **Filtering**: Usually the data must be filtered and enhanced to more appropriate representations to get the important features.
- **Mapping**: After filtering, the data elements must be mapped to visual primitives.
- **Rendering**: The last step of the visualization pipeline, which takes the 3D scene created by the mapping process and renders it into the final image using the user-specified viewing parameters.

2.3 Volume Data Visualization

One field of scientific visualization is volume visualization, which is concerned with generation of visual representation for three dimensional data, i.e, two-dimensional graphical representation of three-dimensional scalar datasets called volumetric data [9].

In general, there are three fundamental techniques of volume visualization [32][25][27], The first one that decomposes the volume into some elements and visualizes these elements individually. The second is concerned with extracting surfaces with target elements and then rendering these surfaces, this technique is called indirect volume rendering. The last technique is the direct volume rendering that assigns optical properties directly to the points that are sampled along rays casted through the volume.

Figure 2.6 shows the deference between direct and indirect volume rendering, where the image generated from the direct volume rendering (b) looks like a multi-layer image in a semi-transparent medium, and the image generated from the indirect volume rendering (a) looks like a solid surface image.



(a) Indirect volume rendering of the skull. (b) Direct volume rendering of the skull.

Figure 2.6: The deference between indirect volume rendering and direct volume rendering[27].

2.3.1 Decomposition Methods

The idea of decomposition methods is to arrange voxels in a Cartesian grid and render them using some primitives like spheres or cubes. Best examples of this method are the tiny cubes and vanishing cubes methods [27][22]. Figure 2.7 shows the concept of the tiny cubes method. Where in the tiny cubes method, a cube (object) is placed at each sampled point and an open space is left between cubes to enable seeing through the cubes.

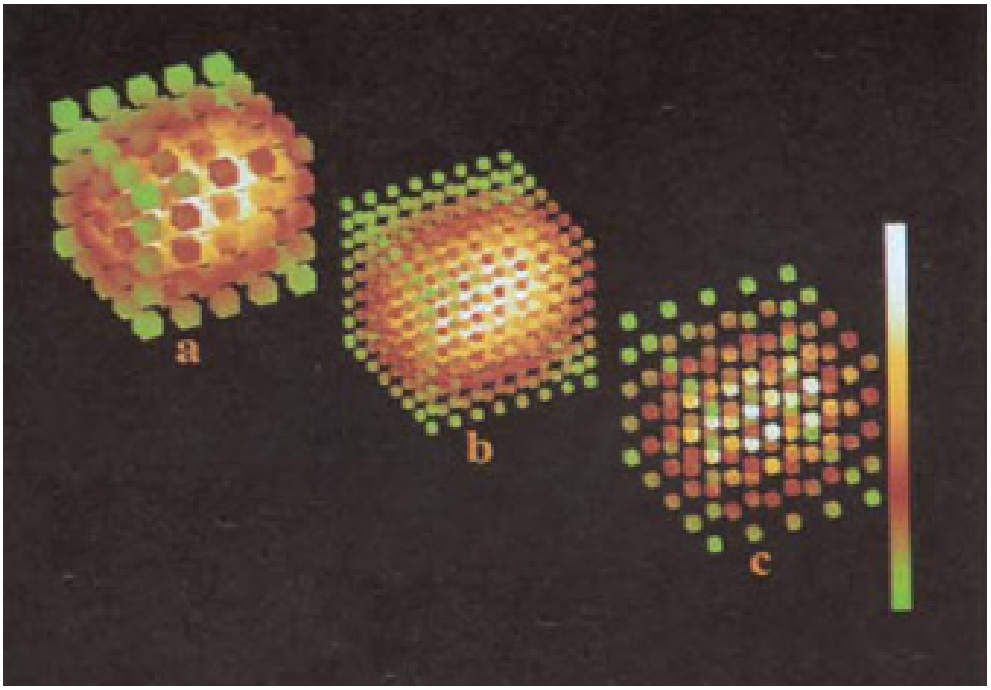


Figure 2.7: The concept of tiny cubes method [22].

2.3.2 Indirect Volume Rendering

Indirect volume rendering refers to the way of converting the volume dataset into intermediate representations and then rendering these representations using traditional rendering techniques. These intermediate representations are called surfaces, or more precisely, iso-surfaces with a specific iso-value [32][31].

The essential step in the indirect volume rendering is the creation of the polygonal mesh that represents the target surfaces, and this can be done using many different methods, such as contour tracing, Opaque Cubes (Cuberille), Marching Cubes, Marching Tetrahedra, and Dividing Cubes [25][27][23].

In this section we will discuss the marching cube method in more detail as it is the most commonly used algorithm and it is of most relevance to our thesis.

Marching cubes

In general words, marching cube algorithm creates a surface from 3D dataset by extracting a polygonal triangle mesh from 3D discrete dataset.

The algorithm proceeds through the volume cube by cube. Hence, the cube term comes from the process of taking eight neighboring points at a time, forming an imaginary cube. Iterating "marching" over a uniform grid of cubes and checking if all 8 vertices of the cube are positive or all the 8 vertices are negative, then no triangle is emitted, otherwise, a triangle will be generated crossing this cube.

The main steps of the marching cubes algorithm are [28]:

1. Read the dataset and generate a cube from two slices and take four neighbors from each slice.
2. Determine the index of each cell by checking the intensity value at the cube vertices.
3. Determine the intersection edges.
4. Compute the intersection by means of linear interpolation.
5. Generate a triangle of the intersection.

The mesh is generated from the union of all the triangles after iterating over all cubes. Figure 2.8 shows the possible summarized 15 intersection cases, where in fact there are 256 cases of intersections.

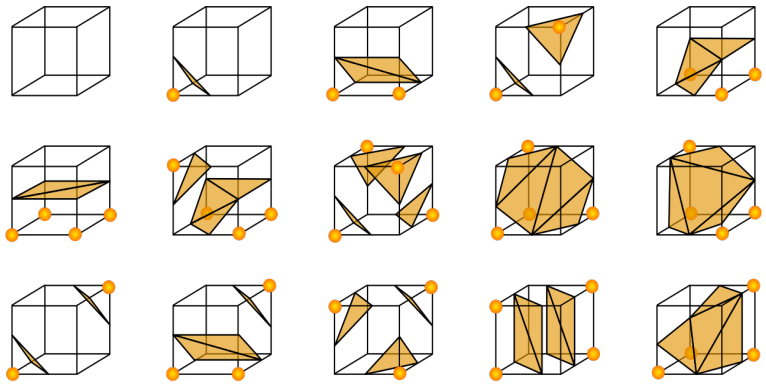


Figure 2.8: The 15 possible intersection cases in marching cube algorithm [35].

2.3.3 Direct Volume Rendering

Direct volume rendering (DVR) generates a projected image directly from the volume data in a semi-transparent medium without the need for intermediate polygonal representation, by accumulating the contribution of each sampled point along the viewing ray to generate the final image [32][25][31].

Unlike surface rendering, each point in DVR should be mapped to optical properties such as color and opacity avoiding the binary classification of the points, whether the point passes through a surface or not.

The key step behind a successful volume rendering is the right choice of the function that maps the whole set of scalar values along the ray to a single pixel in the resulting 2D image.

Ray Casting

Ray Casting is one of the best algorithms for DVR, where the image accumulates, at every pixel, the scalar data along a ray passing through that pixel into the volume data

[31][13].

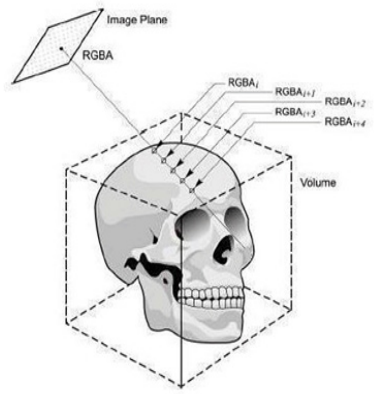


Figure 2.9: Ray casting algorithm [7].

As shown in Figure 2.9 a ray is cast through the volume and samples are taken at equal distances. At each sampled location, the color and opacity are accumulated along each ray to determine the contribution of each voxel to the final image. This happens through a set of processes called pipeline shown in Figure 2.10, these steps are [17]:

Figure 2.10: The volume rendering pipeline.

1. Sampling along the ray usually at equal distances and the sampling points are interpolated from the voxel grid using the Nearest Neighbor algorithm or tri-Linear interpolation.

2. Classification, where the color and transparency are mapped to the sampled points, which is defined by the *Transfer Function*.

3. Accumulate values (compositing), and render them using some rendering methods.

2.4 Volume Classification

Volume classification is the most important and challenging process in volume rendering because it is prone to error more than it is a complex and time-consuming process.

In the classification process, a so called transfer function (TF) is used to assign meaningful optical properties to each voxel in the volume data. These properties will determine the contribution of the voxel to the final image. So, the main role of the TF is to define what part of data should be depicted and how to depict it [17].

In other words, classification is the process of designing and applying TF that distinguishes the different types of components based on their scalar values [32]. Where the TF is a special case of classification that determines that a certain regions in the volume belong to the same materials, so it will have the same color and opacity value in the final image [1][12].

Figure 2.11 shows a TF for volume rendering. According to this TF, intensity value equals to 175, e.g., is assigned an opacity value of 0.7 and given the color yellow.

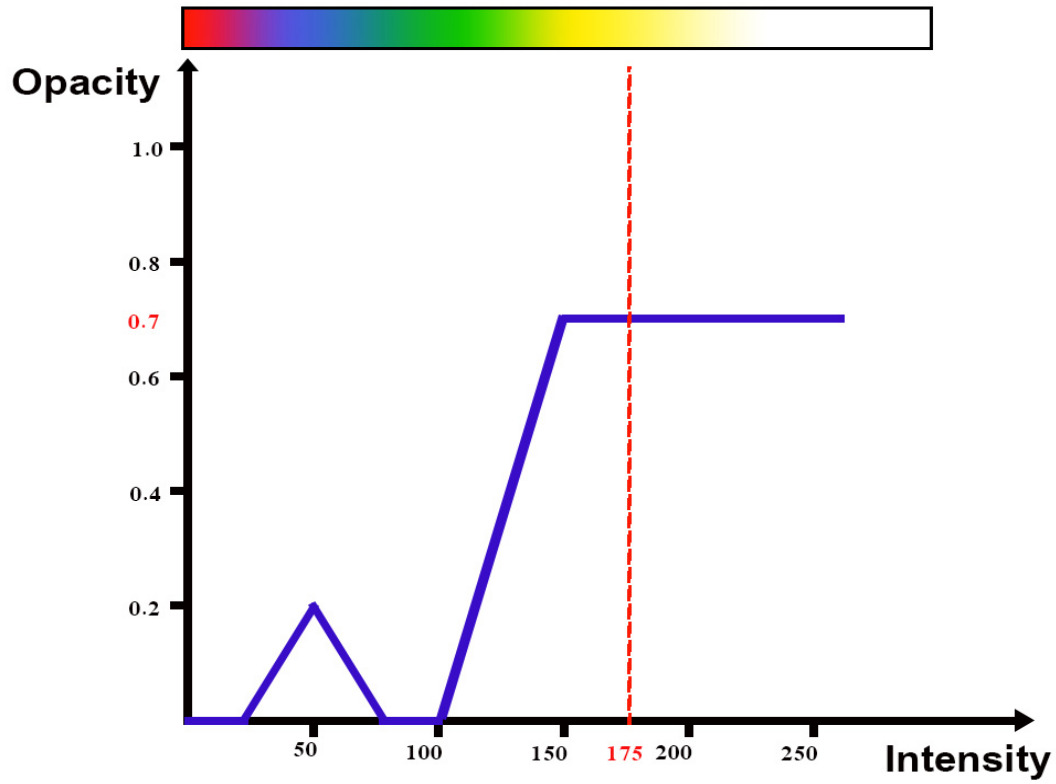


Figure 2.11: Transfer function for volume rendering.

In the ray casting algorithm, the ray function is used to *combine* the scalar values along the ray. I.e., the resulting RGB value for each pixel in the output rendered image is computed by accumulating the previously assigned color and opacity values. In technical terms, the key role of the TF is the estimation of the two functions forming the emission-absorption equation [17]:

$$I = \int_a^b q(s) e^{-\int_a^s k(u)du} ds \tag{2.1}$$

where,

- I is the intensity results from traversing the ray between the points a and b in the volume.

- *q(s)* is the light contribution of the point s along the ray.
- $\int k(u)$ is the attenuation of the light contribution.

2.4.1 Transfer Function Generation Methods

The existing TF generation methods can be divided into two categories:

Data driven methods

These methods analyze the volume data in a position-independent manner and extract various features from the volume. and use these features and available information to generate the TF. The main features include data properties like intensity, and derived attributes like gradient magnitude [26][12].

Image driven methods

Instead of dealing with histogram widgets, the image driven methods propose initial suggestions of rendered slices of images to the user so that the user interacts with the data of the images itself, analyzes the rendered images, modifies on these slices to derive an optimal set of parameters and generates the final image rendering instead of having to manipulate the TF itself [9][3].

The main drawback of such methods includes the limited applicability of such methods as these methods depend on image-related parameters like image resolutions and viewpoint, also depends basically on the user interaction, so it is difficult to automate the process using this type of generation methods.

2.4.2 Automation of Transfer Function

Since designing the TF is complex and time consuming, this represents a bottleneck regarding the efficiency of the volume rendering process as a whole.

While the manual design of the TF is challenging, a huge amount of recent research is focusing on the efficiency of designing a good TF, and a lot of techniques are proposed to facilitate the design of the TF [20][36]. Some of these techniques automatically generate the TF or propose an initial TF that can be modified by the user.

As mentioned above, the image driven generation methods basically depend on the user's interaction with the image slices or rendered images, so it is practically tedious to automate this type although.

In contrast, the data driven approaches could be, at least partially, automated since it depends on the data extracted from the image, but at the same time it needs more computation and has more complexity than image driven approaches.

2.5 Related Works

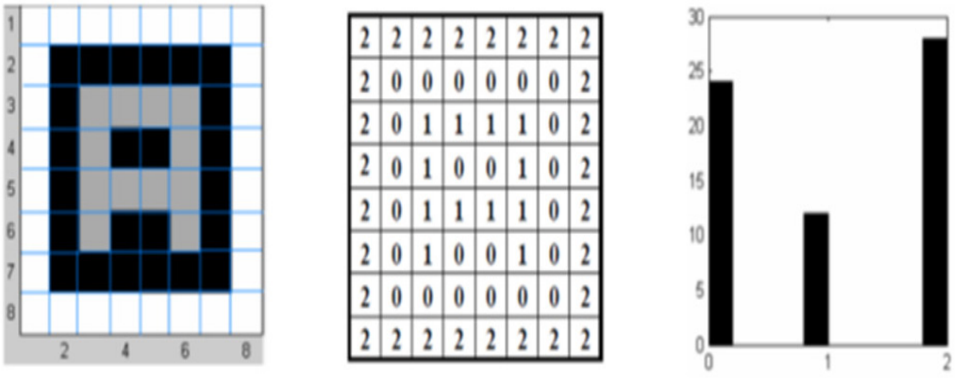
A great amount of researches has been done focusing on the generation of TF either the image driven approach or the data driven approach.

In 1998, Kindlmann and Durkin improved a method to automatically detect the boundaries in the volume by computing the first order derivative of the scalar values. So that the opacity can be assigned automatically to the boundaries [13]. This work formed a base for other researches, and had great attention from other researchers, so many researchers built their studies on this achievement.

In this section we reviewed a set of these studies that are most related to our research.

2.5.1 Histogram-Based Methods

A *Histogram* is a special case of bar chart that is used to explore the volume dataset [36][15]; where each bin in the histogram represents a set of voxels that have the same value. Figure 2.12 illustrates the concept of histogram. where (a) is an image with multiple scalar values, (b) shows how the scalar values are distributed through the image, and (c) is the statistical histogram of each scalar value where each bin represents the number of cells contain the scalar value of that bin.



(a) (b) (c)

Figure 2.12: The concept of histogram.

The use of histograms is popular in generating the TF because the histogram is a statistical method that shows the distribution of data values and other measures through the dataset which is useful for analyzing data.

A frequency distribution histogram was used by Wangjun et al. [10], where a voxel model is constructed for the scalar field at the beginning and a 3D grid of the scalar field is generated according to the voxel model. The frequency distribution histogram is then

generated according to the range frequency in the histogram. Color bands were initialized and assigned to each range in the histogram according to the voxel ratios, which is the ratio of the range frequency value to the total number of voxels. The color bands were generated and set to the ranges in cyclic mode using linear interpolation, where the end color in each rang is the starting color in the next one, and the concentrated regions gain more color grading. A light ray cast is performed in the resampling phase and then the classification is performed in a texture dependent manner to obtain color and opacities.

A Gray-Gradient mode histogram is another type of histogram used by Yish et al. [15]. Figure 2.13 shows the concept of the gray-gradient histogram, where the peaks represent different objects regions and the valleys represent the transition points between adjacent objects.

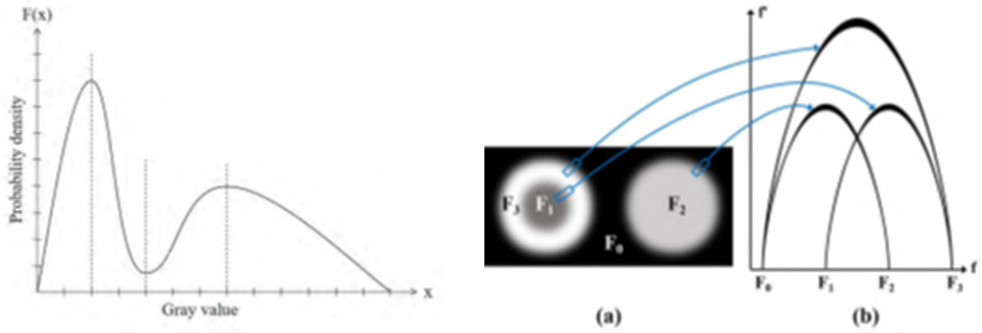


Figure 2.13: The relationship between the gray value and gradient value, and the concept of gray-gradient histogram [15].

Based on the fact that different tissues of the human body have different ranges of CT values, a pre-defined look up table was used to define the gray values of different tissues. This table is used to assign colors and opacities to the sampled points independently. Suppose that different tissues have different waves, the clustering is performed according to the similarity between objects, where the similarity is calculated using the euclidean

distance.

The visibility histogram has also been used as quantitative measurement for visibility distribution of a visibility function, where the visibility of a voxel from a viewpoint represents its contribution to the final image, which is determined by its opacity and the transparency of the remaining voxels before it. The visibility distribution describes the distribution of all voxels, in terms of the probability of each viewpoint.

Marc et al. [26] proposed a method that finds the optimal TF using informational divergence, which is defined by the distance between the projected visibility distribution, captured by a set of viewpoints, and the target distribution, proposed by the user, or the distribution that reflects the importance of the data.

Bramon et al. [2] extend this approach to multimodal volume visualization. The multimodal visualization aims at combining different volume datasets in a single one that reflects as much information as possible using information theory. In this method, two main information measures are used, predictability and entanglement. The predictability reflects how one dataset affects the other and the entanglement tells that a large value from one dataset relates to large information from the other one.

Using these two measures, Broman et al. [2] propose to fuse color and opacity automatically from two volume datasets into one. The opacity obtained via an optimization procedure that minimizes the divergence between projected visibility distribution and target distribution using pairs of intensity values for each voxel from the two datasets instead of one to generate 1D TF's from both datasets to generate a 2D TF that is used to fuse the gradient values in one value, and the color of two TFs into a single color. The problem with color fusion is that the color that is generated from the fusion strategy is a new color, so, this may cause misinterpretation.

Cai et al. [3] depended on the Intensity-Gradient magnitude Histogram (IGM Histogram) to match observed visibility with a target visibility using Newton's method, where the observed visibility is achieved by computing all probabilities of all bins from all viewpoints. The target is then determined by the importance of each bin in IGM histogram. According to the information theory, the regions with low occurrence carry more information, and are more important than the regions with high occurrence. From that, bins with low occurrence correspond to highly important features. Cai et al. [3] used Newton's method to match observed visibility with target visibility to achieve faster convergence because it is a second order derivative method.

2.5.2 Feature-Based Methods

Considering the histogram-based method, which ignores the spatial information of the volume, the feature-based methods raised to consider this problem and take spatial information into account.

Kindlmann et al. [13] assumed that the boundary is located at the high variations of the scalar values between two areas in the volume. Mathematically, the boundary is located at the point that has a maximum first order derivative and second order derivative equals to zero. The boundary thickness is defined according to the value of standard deviation and defined as equal for the whole volume.

Mesquita et.al [20] built their work on Kindlmann's works and tried to improve on it by suggesting the use of the third order derivative so the boundary can be detected in terms of the three directional derivatives, where the boundary can be detected by the negative values of the third derivative. Mesquita also suggested determining the value of the standard deviation for every voxel as it is different among the volume.

Bo Ma et al. [18] proposed a feature similarity method for clustering volume data based on iso-surface similarity. The similarity between two iso-surfaces is determined by the number of intersected cells between them. Based on that, a similarity matrix contains the similarities between each pair of iso-values is created and clustered by the K-Medoid clustering algorithm to extract the various regions in the volume. Instead of treating the volume voxels separately, Bo Ma. et al. [18] treated the features as a whole object and redefined the notion of visibility considering that the similar features contribute to each other's visibilities, and occlude each other's if they are dissimilar. Accordingly, the opacity is computed based on the similarities between iso-surfaces and weighted by the distance and assigned to each object.

Tianjin et al. [36] depended on assuming that the voxels belonging to the same structure have the same intensity and gradient magnitude and are located together in the IGM histogram. Based on that, they proposed to cluster IGM histogram bins using the affinity propagation algorithm depending on the intensity and gradient magnitude as well as the spatial information of voxels and their location on the histogram. Each bin in the histogram contains all voxels in the volume that have the same intensity and gradient magnitude. The mean and variance are calculated to eliminate the noisy and non-sufficient voxels.

After constructing the IGM histogram, the affinity propagation (AP) algorithm was used to cluster bins of histogram. Where the advantage of AP algorithm is that it does not need to identify the number of clusters, so it supports the automation of the process. The message passing AP algorithm depends on two main messages, responsibility and availability. The responsibility is sent from data point to the cluster's center to decide whether its suitable to be a cluster center or not, and the availability message is sent from

center to data point telling whether to choose it as a center or not. Using the similarity matrix, the responsibility and availability messages continue passing and updating until convergence. The similarity matrix is created based on Euclidean distance of bins in IGM histogram and spatial location of voxels which depends on the number of direct neighborhood of bins in IGM histogram. After classification, the opacity and colors assigned to voxels depending on IGM histogram.

2.5.3 Graph-Based Methods

Sharma et al. [30] proposed a graph-based method for material deduction and used it for an automated TF generation. In this method a graph-based framework is used to create a material definition that can map a solid material interior to a color value. This is done in two steps; the first is the graph deduction by creating a set of materials and using a distance matrix to reduce materials using parent-child hierarchy and merge all children with their respective parents, and the second step is constructing the valid graph by merging similar materials together and merging repeating interface into one.

The optimal graph can be achieved by maximizing the separation between adjacent materials and minimizing the intra-material variance. For the TF widgets, the colors are assigned by using dissimilar colors to adjacent materials and transparencies are assigned to materials using the occlusion spectrum, where the occlusion value indicates how deep the material is in the volume. The user can later change colors and transparency to hide some parts.

3

Data and Methods

As we mentioned in the previous chapter, the volume rendering pipeline consists of three main steps starting from sampling data along the ray using the ray casting algorithm, then, the classification step to map the color and transparency to the sampled points using the transfer function, and finally, the composition step of the values and rendering it using various rendering methods.

In this chapter our proposed system of designing the transfer function will be described in details as it is the most challenging step in the volume rendering process. First, we will describe the nature of the used data and how it was prepared. Then we will describe the volume classification process starting from creating the similarity matrix using the isosurface similarity, then a detailed description of the similarity matrix classification using the affinity propagation algorithm. Finally, the details of the

rendering process will be described.

3.1 Data Preparation

Our data was originally available in the DICOM format, which is the international standard for transmitting, storing, retrieving, printing, processing, and displaying medical imaging information. This type of files is hard to be dealt with directly without implementing or including an application-specific DICOM importer.

To facilitate working with input data in our project, we used the MRIConverter software tool to convert the DICOM files into MetaImage format[1]. The MetaImage format has an easy structure and supports multi-dimensional images. In essence, it consists of two files, the Header file and the Raw file. The Raw file that contains the image dataset itself, and the header file that contains the meta information of the data including the image dimensions, the size of the image, the pixel intensity and other information as cleared in the following example.

```
ObjectType = Image
NDims = 3
ElementSpacing = 0.683594 0.683594 0.936056
DimSize = 256 256 150
ElementType = MET_UCHAR
ElementDataFile = CT01b_Head.raw
```

[1] https://itk.org/Wiki/ITK/MetaIO/Documentation

3.2 Volume Classification

The classification process involves two main steps; the similarity matrix creation process which describes how the iso-surfaces are similar to each other, and then the classification of the similarity matrix to cluster the isovalues into various clusters that reveal the various structures of the volume.

3.2.1 Similarity Matrix

The idea of the isosurface similarity matrix is to create a set of cells for every isovalue (in our case, 256 isovalue), each set consists of all cells that are crossed by that isovalue, and determines the common cells between each pair of iso-surfaces to determine the similarity between the two.

Based on the marching cube algorithm of the indirect volume rendering, the isosurface $S\mu$ crosses the cell when μ falls between the minimum and maximum values of the grid points that form that cell. Therefore, the isosurface crosses the cell when its isovalue μ falls between the cell's minimum and maximum intensity values. According to that, the voxelization of the isosurface can be achieved by collecting all the cells that are crossed by the isosurface together. As a result, the similarity between two isosurfaces can be determined by the number of intersected cells between these isosurfaces.

Given a volume V, each isovalue μ has a set of cells CellSet(V,μ) that contains all the cells that crossed by that isovlaue.

The similarity between each pair of isovalues is calculated using the equation [18]:

$$Sim(\mu_1, \mu_2) = \frac{Cellset(V, \mu_1) \cap Cellset(V, \mu_2)}{Max(Cellset(V, \mu_1), Cellset(V, \mu_2))} \tag{3.1}$$

The normalization factor in the denominator is the maximum surface between the two isosurfaces. Where in the case of significant size deference, the larger isosurface is used to assign low similarity value to the two isosurfaces. So, the normalization factor is used to normalize the similarity value to the range of [0,1].

In order to illustrate the concept of the proposed isosurface similarity calculation, Figure 3.1 represents three isosurfaces in a two-dimensional grid. From the Figure, we can calculate the similarity between the red and the green isosurfaces as (49/81), the similarity between the blue and the red isosurfaces as (19/81) , and the similarity between the green and the blue isosurfaces as (1/82).

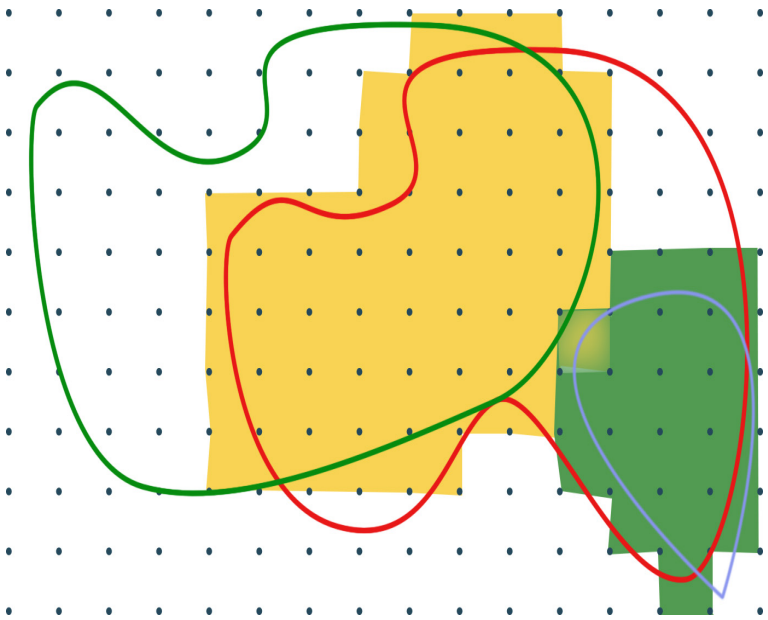


Figure 3.1: The concept of isosurface similarity calculation.

Another important issue in calculating the similarity between isosurfaces is the cell's size. Where the large cell size has a higher probability of being intersected by an isosurface than the smaller, so, we used an octree partitioning to build a hierarchy of cells and calculate the similarity in each level and aggregate the similarity value of all levels to achieve the final similarity.

The octree structure can be defined by enclosing the volume in a cube and then divide the cube by bisection in the three dimensions into eight octans, and then subdivide each octan. This process of division continued to specific level [14]. Figure 3.2 shows the concept of the octree partitioning.

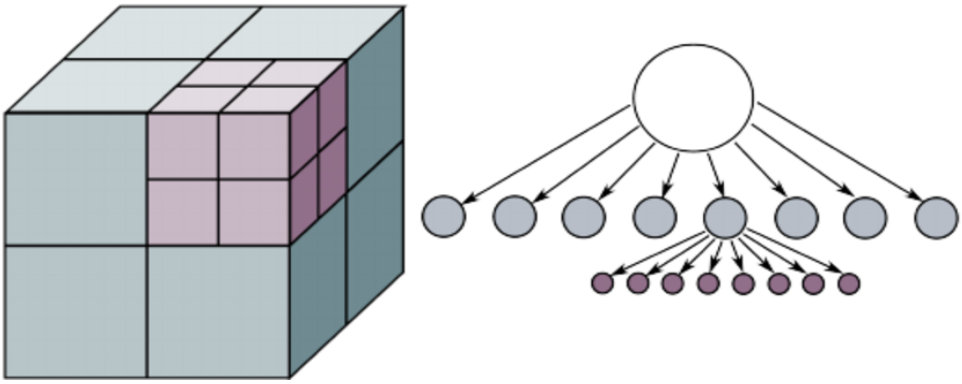


Figure 3.2: The concept of octree partitioning [11].

In our project, we start by defining the whole volume as one cell and at each level we divide each cell into eight cells and calculate the similarity at each level and, at the end, aggregate the similarities of all levels to get the final similarity.

The final similarity between each pair of isosurfaces is defined as [18]:

$$Sim(\mu_1, \mu_2) = \frac{1}{K} \sum_{h=1}^{n} w_h Sim(\mu_1, \mu_2) \tag{3.2}$$

n: is the height of the octree.

$W_h = h$ is the weight factor used to balance the contribution at each level. We set it to be the octree level.

$\frac{1}{K}$: is the normalization factor. Where $K = \sum_{h=1}^{n} w_h$.

Finally, the distance between each pair of isosurfaces is defined as [18]:

$$1 - Sim(\mu_1.\mu_2). \tag{3.3}$$

3.2.2 Classification using Affinity Propagation

The generated similarity matrix must be entered in the clustering process in order to classify the volume into various clusters, each cluster shares common features.

As the goal of this research is to classify the volume automatically without the intervention of the user, we propose using the Affinity Propagation classification algorithm as it doesn't need to pre-define the number of clusters.

The AP clustering algorithm is a message-passing algorithm introduced in 2007 by Frey and Dueck [36]. It can be understood by imagining that each point sends a message informing its targets of each target's relative attractiveness. Each target then replies to all senders informing them its availability to associate with the sender. The sender then replies with a message of the target's revised relative attractiveness. This procedure continues until an optimal set of examples and relative clusters emerges.

There are two types of messages exchange through the AP algorithm, the first message is the *"Responsibility"* message, which is sent from the data point to a candidate exemplar, reflecting how well suited for the exemplar, to serve as an exemplar, for that data point.

The second message is the *"Availability"* message, which is sent from the exemplar to the point and reflects how appropriate it is for the point to choose that candidate exemplar as its exemplar [6][33].

At the beginning, the similarity matrix that is generated from the previous step, which is constructed from the number of the intersected cells between each pair of isovalues, should be negated and the diagonal elements *S(i,i)* should be replaced by the

"*Preference*" value of all items.

As the AP algorithm does not need to pre-define the number of clusters, at the beginning, it suggests that all data points are equally suitable as exemplars, this is done by replacing the diagonal elements in the similarity matrix *S(i,i)* with the preference value, which always represents the median or the minimum value of the similarity matrix [6].

After that, the availability matrix and responsibility matrix are iteratevility updated using the following equations[6][33]:

$$r(i,k) \leftarrow s(i,k) - \max_{k's.t.k' \neq k} \{a(i,k') + s(i,k')\} \tag{3.4}$$

$$a(i.k) \leftarrow \min \left\{ 0, r(k,k) + \sum_{i's.t.i' \notin \{i,k\}} \max \{0, r(i',k\} \right\} \tag{3.5}$$

where i and k, respectively, refer to the rows and columns of the dataset.

The diagonal elements in the availability matrix are calculated differently using the *"self availability"* equation [6]:

$$a(k.k) \leftarrow \sum_{i's.t.i' \notin \{i,k\}} \max \{0, r(i',k\} \tag{3.6}$$

The criterion matrix is constructed to combine availability and responsibility in order to monitor the exemplars, where the exemplars identified by the positive criterion values [33].

$$c(i.k) \leftarrow a(i,k) + r(i,k) \tag{3.7}$$

Finally, the algorithm terminates after a fixed number of iterations, the message falls below a threshold, or when the decision is not changed for a fixed number of iterations, and the rows that share the same exemplar are set in the same cluster.

To avoid numerical oscillations, a damping factor λ should be used when updating the availability and responsibility, taking λ times from the previous iteration and $1 - \lambda$ from the current iteration. Where λ is between 0 and 1.

3.3 Rendering

After the volume is classified into clusters, it is the time to translate the clusters into various features or components. Where each point of each cluster should have the same color and opacity.

In the rendering step, many issues should be addressed. The first thing is that the AP algorithm generates a great number of clusters although it generates good clustering. So, in order to address this problem, we decided to rearrange the clusters according to the number of voxels fall in each cluster and render the highest 15 clusters.

The color and opacity are set to each cluster with a default value at the beginning, and give the user the ability to change color and opacity values in order to give a more realistic and aesthetic rendering.

Note that the automation of this step is not desired; the reason behind that is human perception, and the ability to understand and interpret the resulting clusters into more realistic images. For this purpose, we visually inspect the generated clusters and assign colors that suit the underlying tissue type in the input image (e.g., bone, muscle, air-filled

space, etc..

Summary

In this chapter we presented the method and the data used to achieve the goals of the project. The experiment and the results will be discussed in the next chapter.

4

Experiment and Results

After discussing all the aspects related to the research in the previous chapter, this chapter discusses the experiments carried out through the research and the results of the experiments.

4.1 System Platform

The system is customized to receive a 256-intensity range medical image and can be extended to any image range with any medical image modality including MRI, CT, etc. and can work on any computing system that includes a c++ compiler and the VTK toolkit.

The implementation of the system was divided into three sub-programs as shown in

Figure 4.1:

1. **Similarity matrix creation sub-program**: creates the similarity matrix and stores it in a file.

2. **Classification sub-program**: performs the classification process and stores the result in a file.

3. **Rendering sub-program**: performs the rendering process and generates the final image.

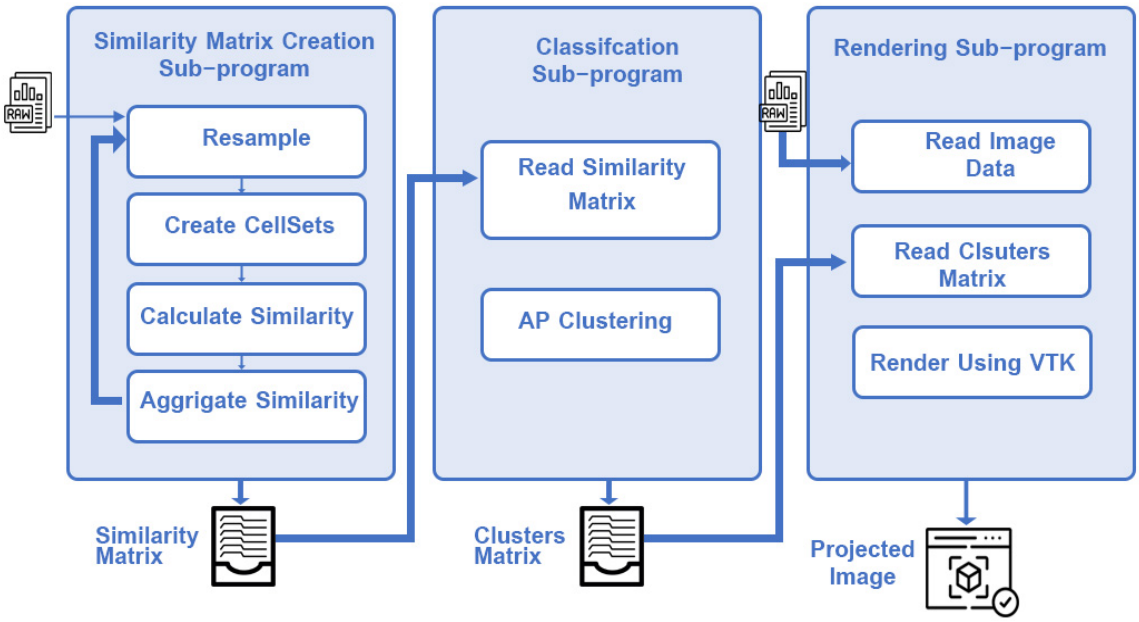


Figure 4.1: System Platform.

The similarity matrix creation sub-program takes a medical image as an input, re-samples it. After initiating the image, the program calculates the similarity matrix between each pair of isovalues in a hierarchical octree structure and aggregates the similarity of all levels to form the final similarity matrix and then stores it in a file to be used in the next step.

The classification sub-program takes the 256*256 similarity matrix as an input and then performs the classification process automatically using the Affinity Propagation clustering algorithm to expose the various clusters of the data that reveals the various structures of the input image and stores the result in a file.

The rendering sub-program takes the 1* 256 clusters matrix as an input and uses it to define the transfer function to be used by the rendering module that generates the final rendered image. As required by the VTK toolkit, these are defined in the form of color and opacity lookup tables.

4.1.1 Implementation Details

The process of similarity matrix creation started with reading a 256*256*256 image and defining the whole volume as one cell. A cellset is then created for every isovalue (256 isovalue). In the first iteration, all the isovalues had the same similarity value equal to 1 as they all shared the single cell that represents the whole volume.

In the second iteration, the cell is divided into 8 cells and then a cellset created for every isovalue and the similarity is calculated using equation (3.1).

This is done during 8 levels of partitioning of the octree. At the final iteration, the final similarity is calculated using the equation (3.2).

The weight of each level was fixed to be the height of the octree at that level, as the higher levels of the octree have a discrimination more than the lower levels. Finally, the distance matrix is created using equation (3.3) and stored in file to be used in the classification level.

The classification step starts with reading the similarity matrix and keeping all the

items and replacing the diagonal items with the preference value. In our research we set the preference value through two experiments and monitor the results in each. We set the preference value to the minimum value in the first experiment and the median value as the preference in the second experiment.

In order to get the optimal classification result, the responsibility and the availability matrices are updated during 190 iterations when we used the minimum value as a preference value in the first experiment, and 100 iterations in the second experiment when we used the median value as a preference value. This decision was taken according to the convergence of the process as the results were converged at 190 iterations when using the minimum value as a preference, and it was converged at 100 iterations when using the median value as a preference.

The damping factor λ was fixed to be 0.5 in both experiments.

4.2 Tools

4.2.1 Visualization Toolkit

The visualization toolkit (VTK) [1] is a software system used for data processing and visualization, including image processing and rendering, geometry computations, medical image analysis and many other areas of data presentations. The role of VTK is to take the data and transform it into visual representations (visualizations) that can be understood by humans. This is done through a data flow pipeline that includes data obtaining, processing, representation and rendering.

[1] https://vtk.org/

VTK Pipeline

The VTK contains a processing and rendering pipeline that consists of three basic classes of objects; objects to represent data (`vtkDataObjects`), objects to process, transform, filter and map data objects from one form to another (`vtkAlgorithm`), and objects to execute the pipeline (`vtkExecutive`).This pipeline consists of the following steps as shown in Figure 4.2:

Figure 4.2: The VTK pipeline.

- **Source**: to create a source data file, either by reading it from a stored file like `vtkMetaImageReader` that reads an image file, or by creating a source file like, e.g., `vtkConeSource` that creates a 3D cone.

- **Filter**: to transform the output of the source or another filter from one form to

another, like `vtkCutter` that cuts the output of the previous filter.

- **Mapper**: that transforms the data to graphical primitives like `vtkPolyDataMapper` that maps polygonal data to graphics primitives.
- **Actor**: represents the object rendered in the scene, including properties and position.
- **Renderer**: displays the rendering on the screen.

4.3 Results

In order to test the system and ensure its performance and effectiveness we use two types of data; testing data and real data.

Testing Data

In the volume rendering there are no well-defined benchmarks to compare the results with, so in order to test the effectiveness of the system we used testing images, which are images with a pre-defined number of isovalues, and monitor the results to insure the correctness of the result before using the real complex datasets.

The testing images that were used were a two-values image with a specific 2 intensity values, a three-values image, a four-values image and others.

Real Data

The real data images used in the experiments include brain MRI image, abdomen MRI image, and foot CT image and others. These datasets were acquired from our clinical

partners .

To test the effectiveness of the classification system we re-sampled all the used images into (256x 256x 256) images, and displayed each class separately by setting the opacity of the other classes to 0.

Note that in all images result's we ignore the first cluster as it represents the background of the image.

For an image of two certain isovalues (50,127), the generated similarity matrix contains just three values according to the values of each voxel, if it is less than 50 or if it is in the range of [50-127] or it is greater than 127. as shown in Figure 4.3.

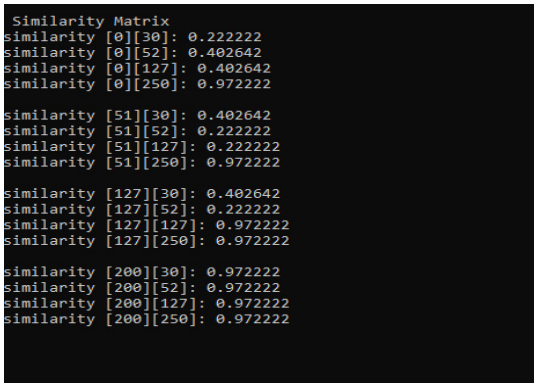


Figure 4.3: Samples of similarity matrix for an image of two isovalues.

The classification process was carried out through two experiments in which the preference value changed. The results of the experiments are summarized as follows:

4.3.1 Experiment 1

In the first experiment we set the minimum value as a preference for the classification process. We found the algorithm converges after 190 iterations.

For the two-value image, the result of the classification process, depicted in Figure 4.4, shows how the isovalues are distributed through the clusters, and what the values of isovalues in each cluster are.

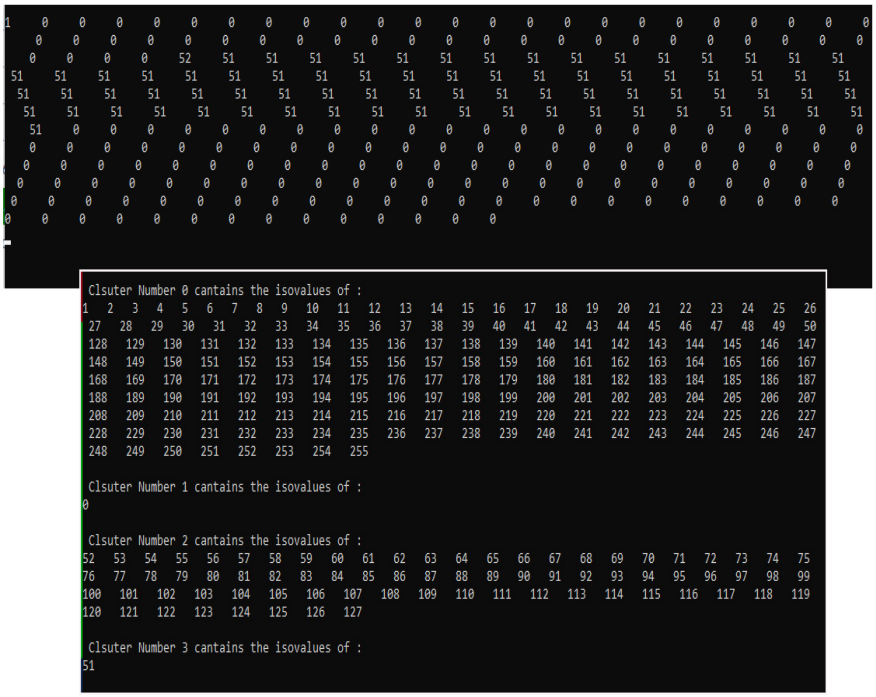


Figure 4.4: Clustering result of two-values image similarity matrix.

The rendering result of the image shown in Figure 4.5 shows how the image represents the separation of the two values.

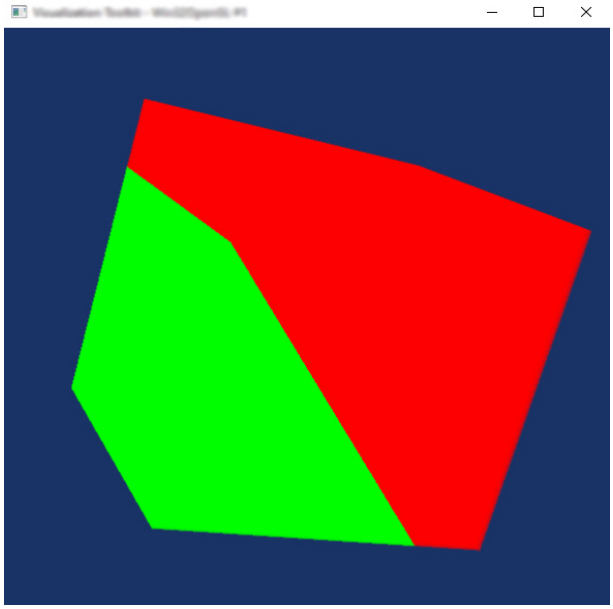


Figure 4.5: Rendering result of two-value image.

For the real data, the results come as follows:

1- Foot Dataset

Figure 4.6 shows the rendering result of the foot dataset.

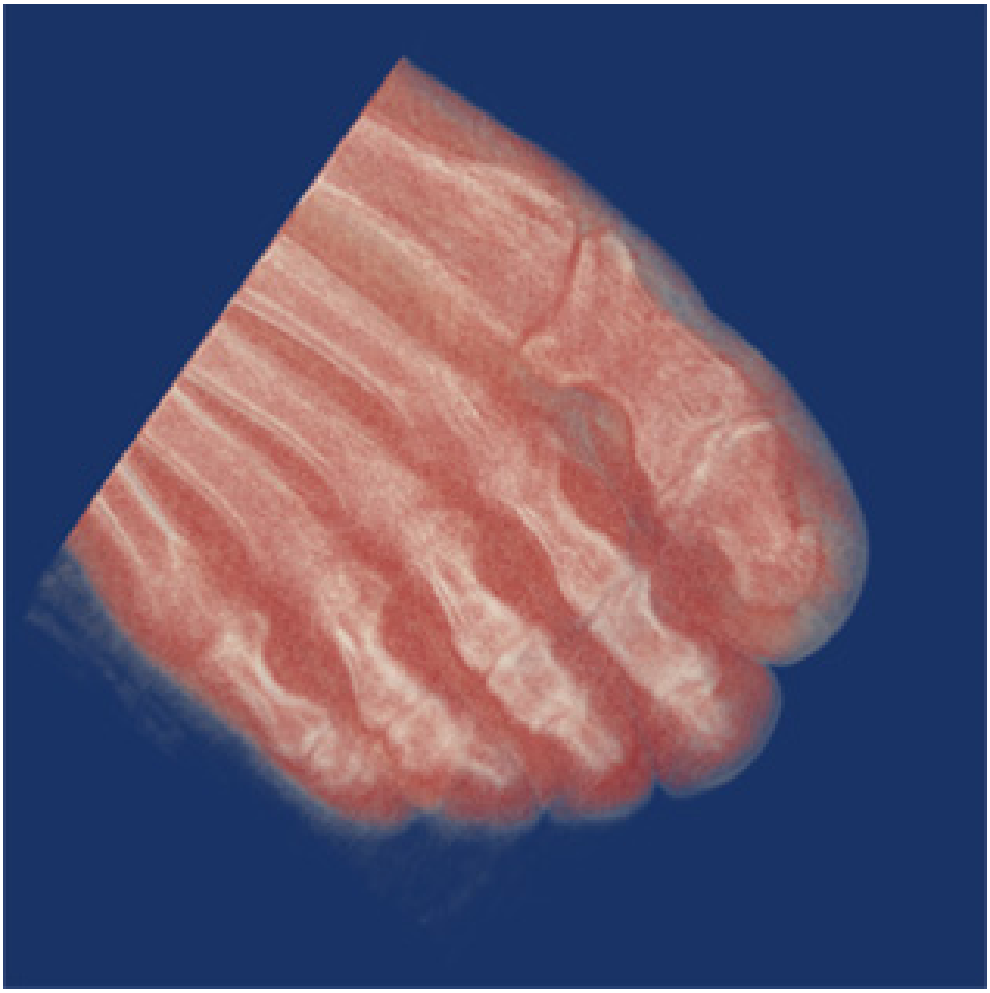
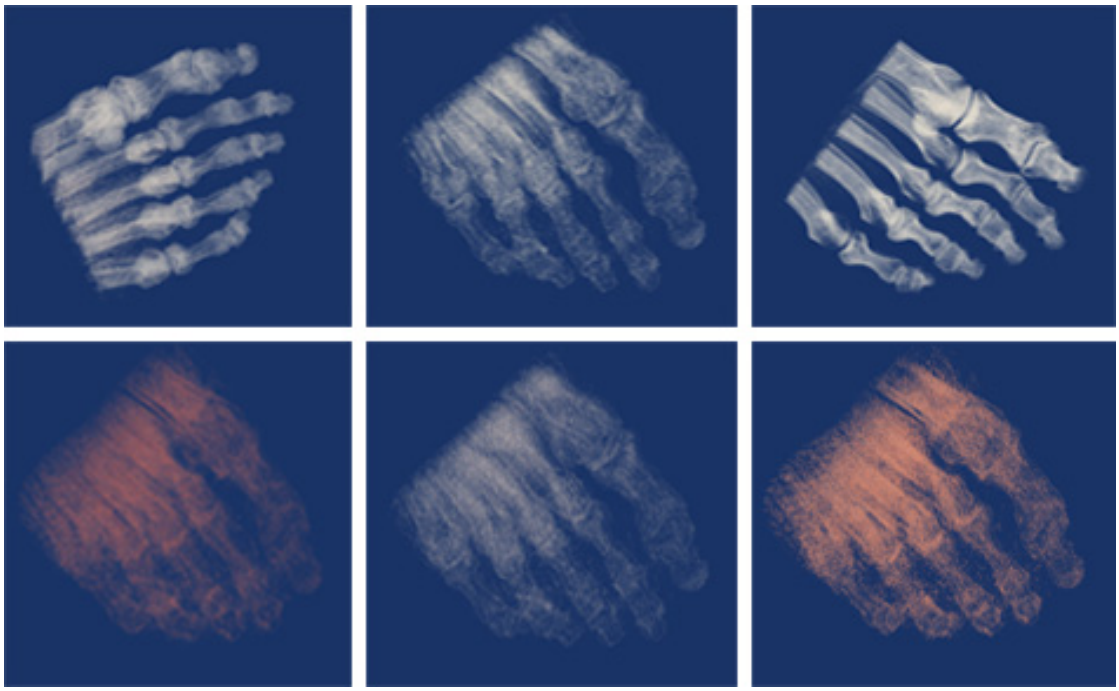
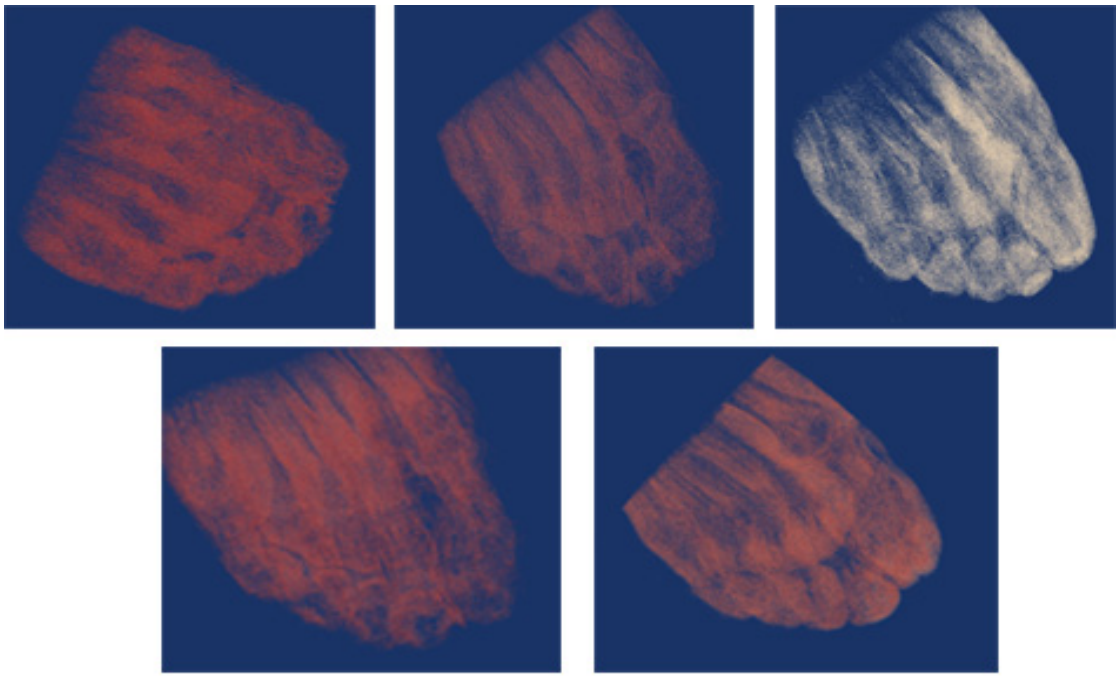


Figure 4.6: Rendering result of foot Dataset.

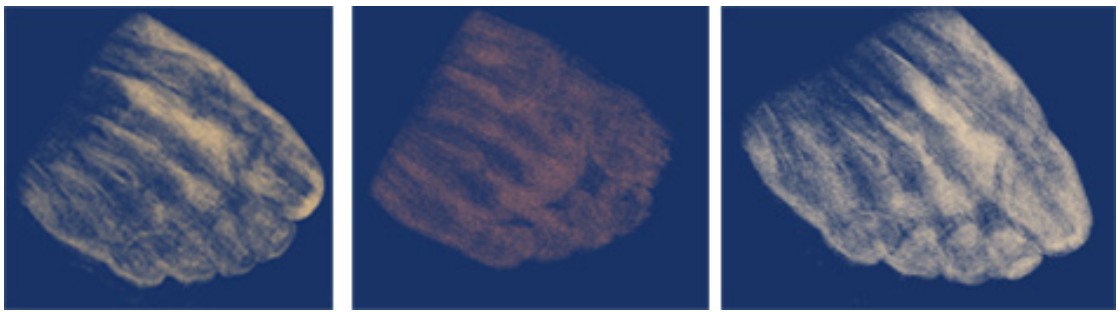
Figure 4.7 shows the resulting classes of the foot dataset, where Figure 4.7a shows the bones and Figure 4.7b shows the various inner tissues and muscles, and Figure 4.7c shows the skin and outliers.



(a) Classes representing bones.



(b) Classes representing various tissues and muscles.



(c) Classes representing outliers.

Figure 4.7: The resulting classes of foot dataset.

2- Brain MRI Dataset

Figure 4.8 shows the rendering result of the brain MRI dataset.

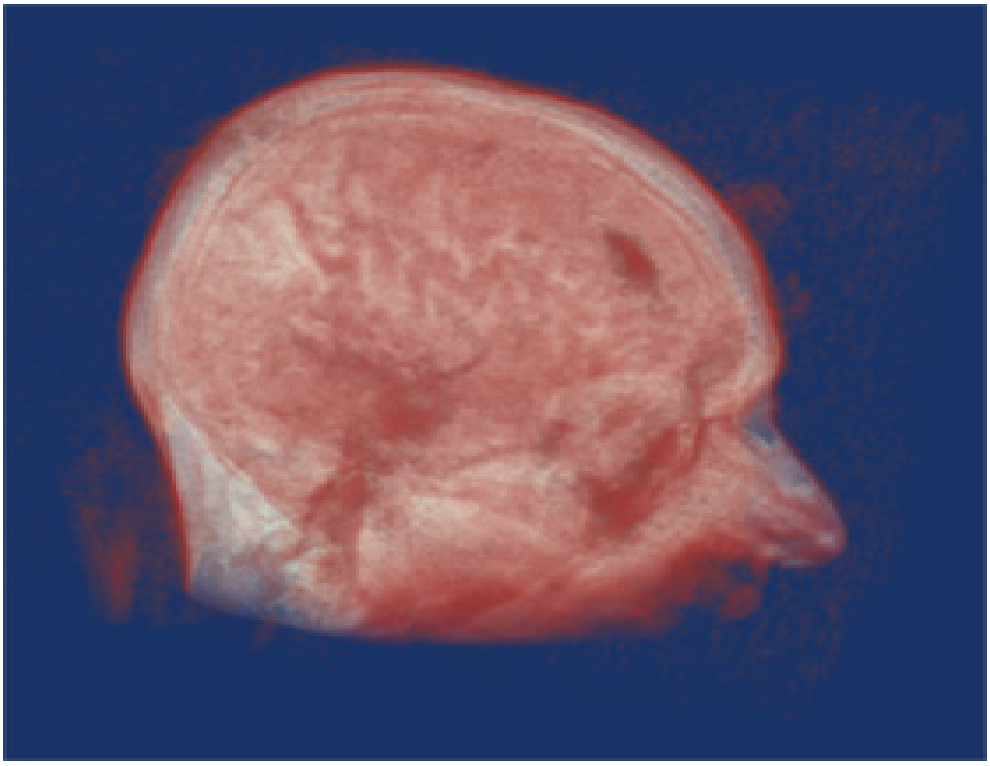
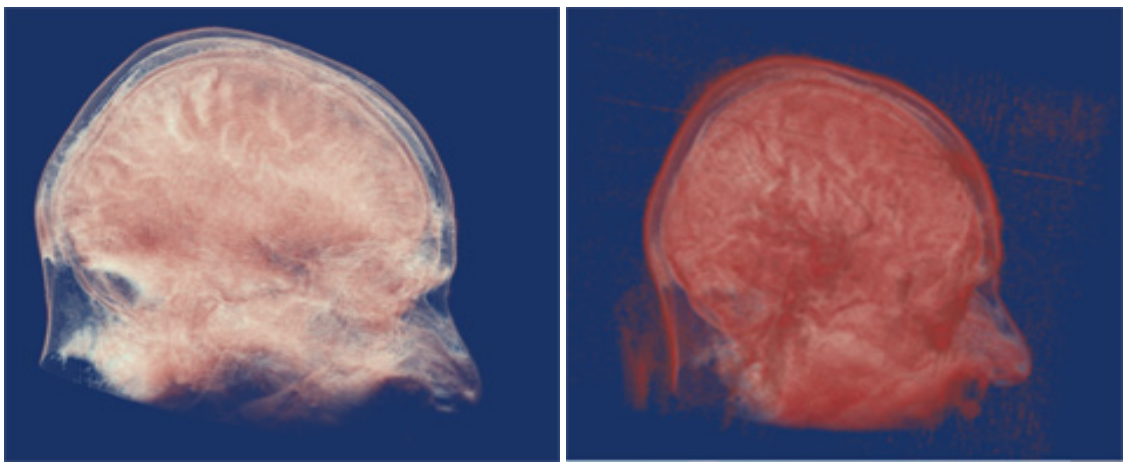


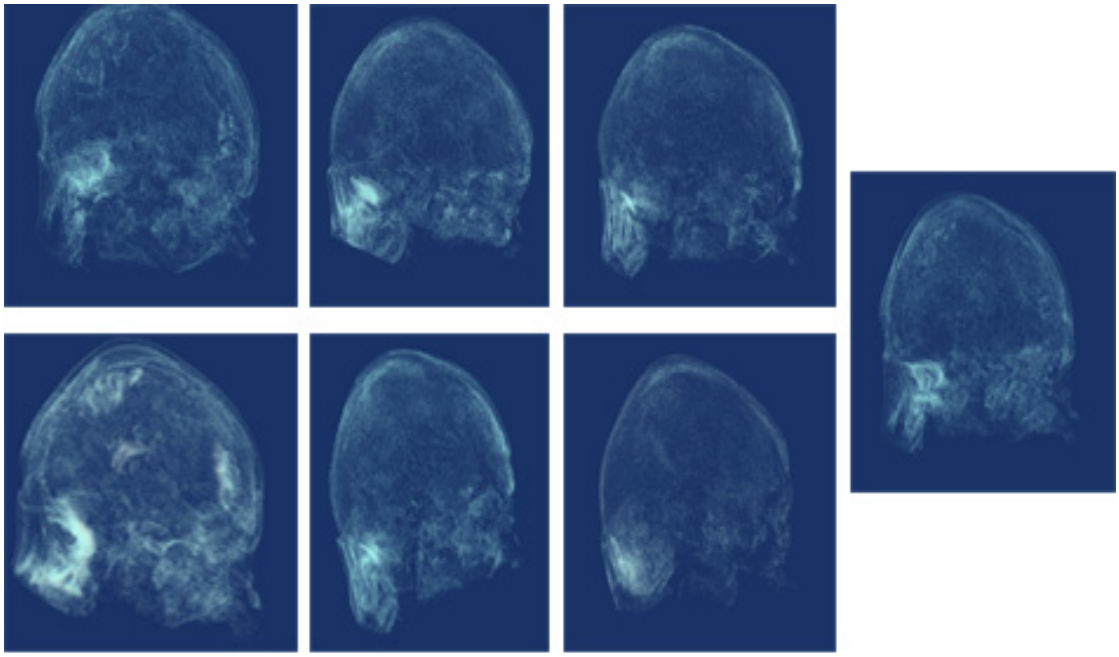
Figure 4.8: Rendering result of Head MRI Dataset.

Figure 4.9 shows the resulting classes of inner and outer tissues of brain, and Figure 4.10 shows the various inner tissues and skull bones.

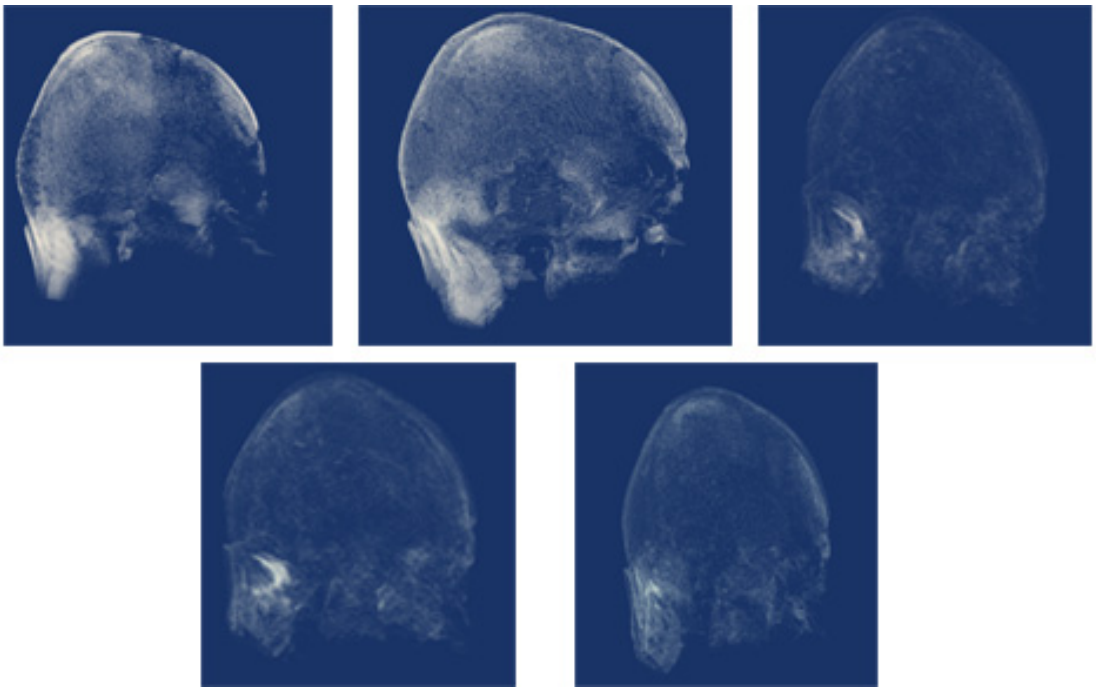


(a) Inner tissues of brain. (b) Outer tissues of brain.

Figure 4.9: Inner and outer tissues of brain.



(a)



(b)

Figure 4.10: Different tissues of brain

3- Abdomen Dataset

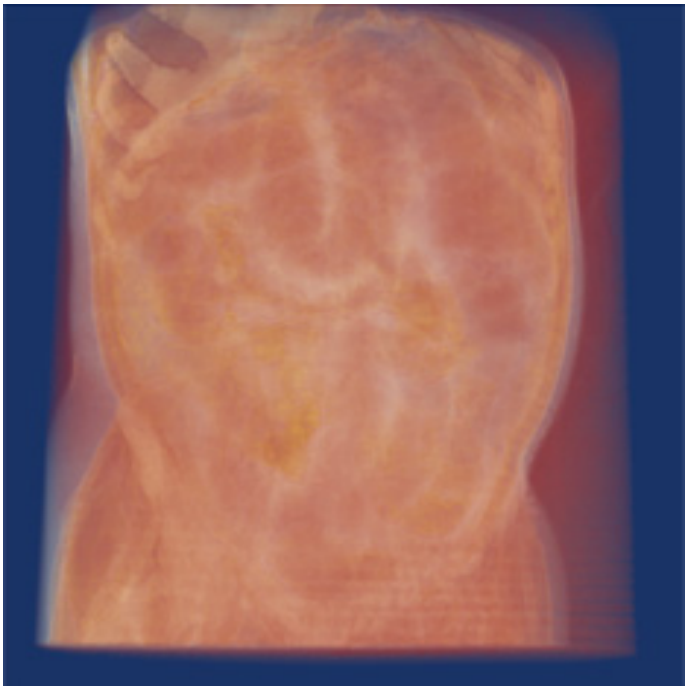


Figure 4.11: Rendering result of a abdomen dataset.

Figure 4.11 shows the rendering result of the abdomen image. Figure 4.12 shows the colon, Figure 4.13a shows the various organs, Figure 4.13b shows the vertebral column and pelvic bones.

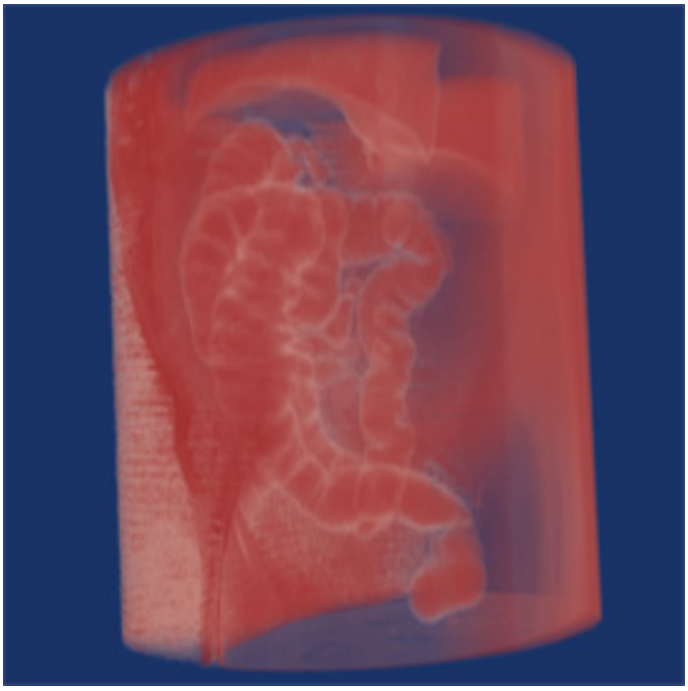
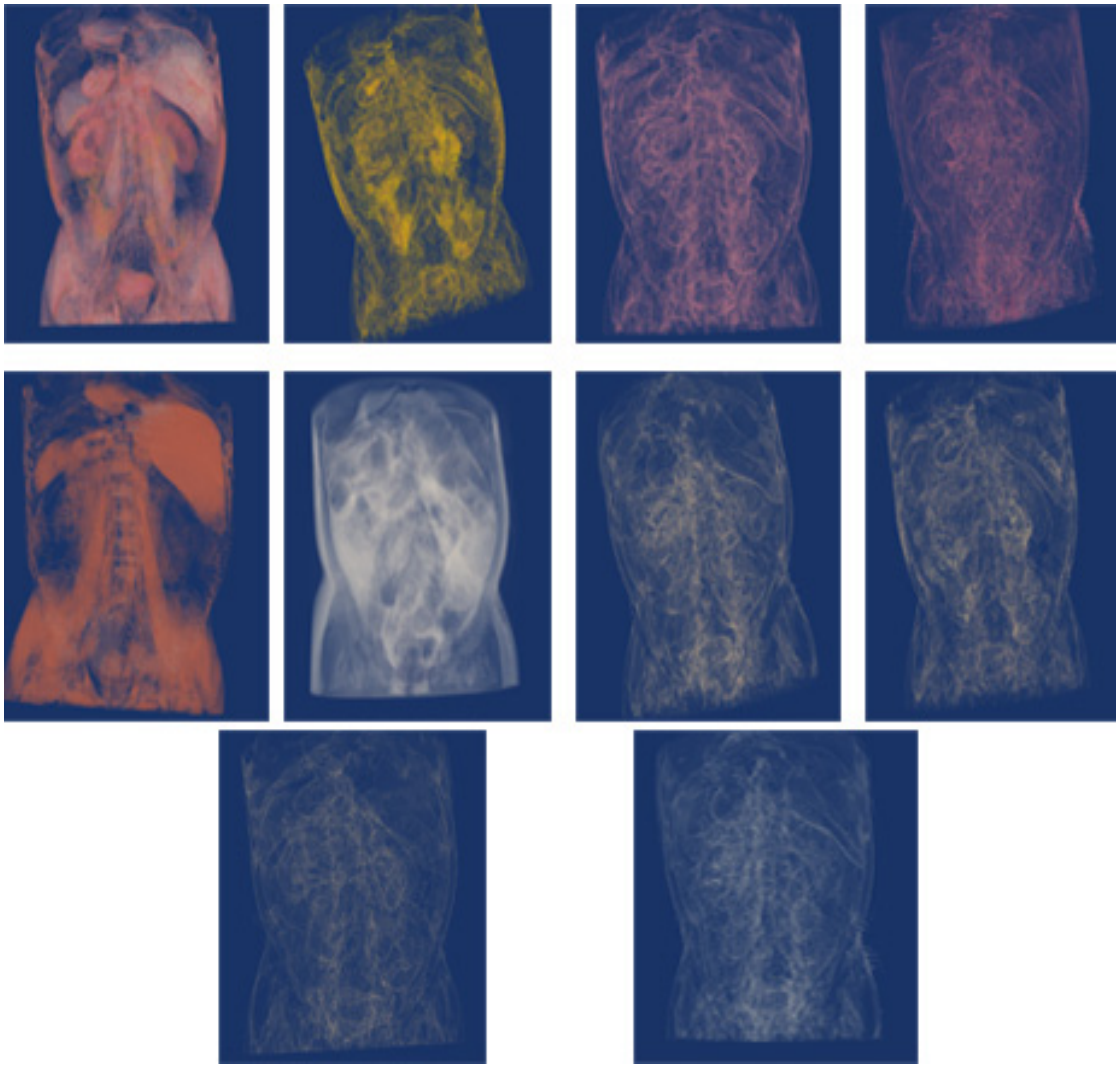
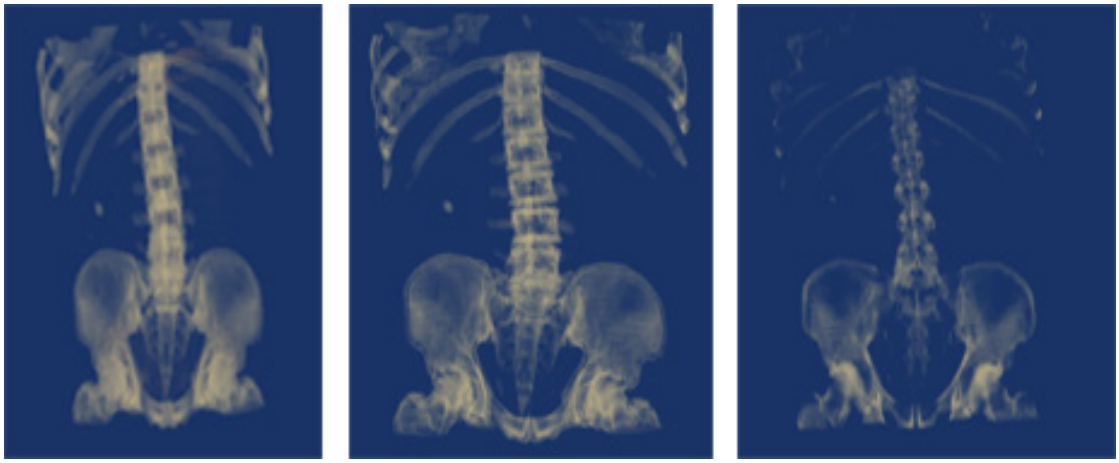


Figure 4.12: Colon tissues rendering.



(a) Different organs of abdomen dataset.



(b) Vertebral column and pelvic bones.

Figure 4.13: Different tissues of abdomen dataset.

4.3.2 Experiment 2

In experiment 2 we set the median value as a preference value for the classification process, and the algorithm converged after 100 iterations.

For an image with a pre-defined four intensity values, the result of clustering comes as shown in Figure 4.14, and the rendering result comes as shown in Figure 4.15.

Figure 4.14: Clustering results of a pre-defined four intensity values image.

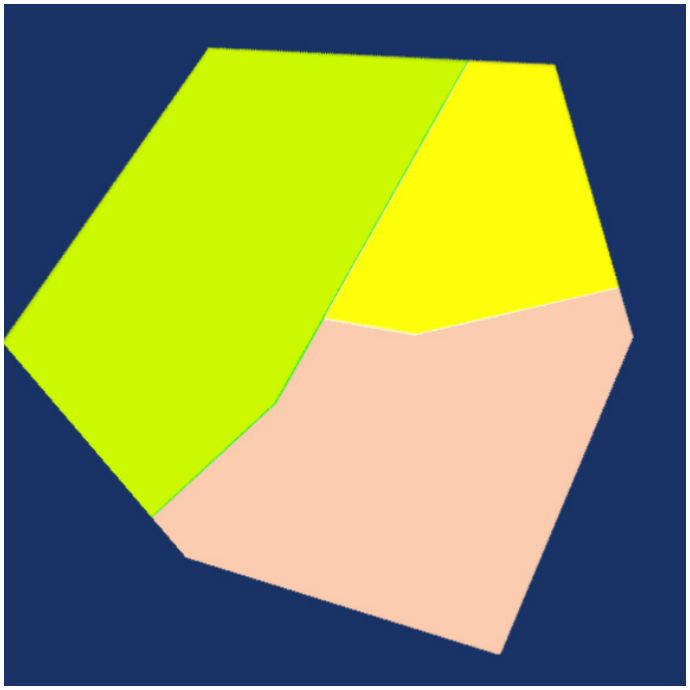


Figure 4.15: Rendering results of a pre-defined four intensity values image.

For the real data, the rendering results come as follow:

1- Foot dataset

The rendering result for the foot dataset comes as shown in Figure 4.16.

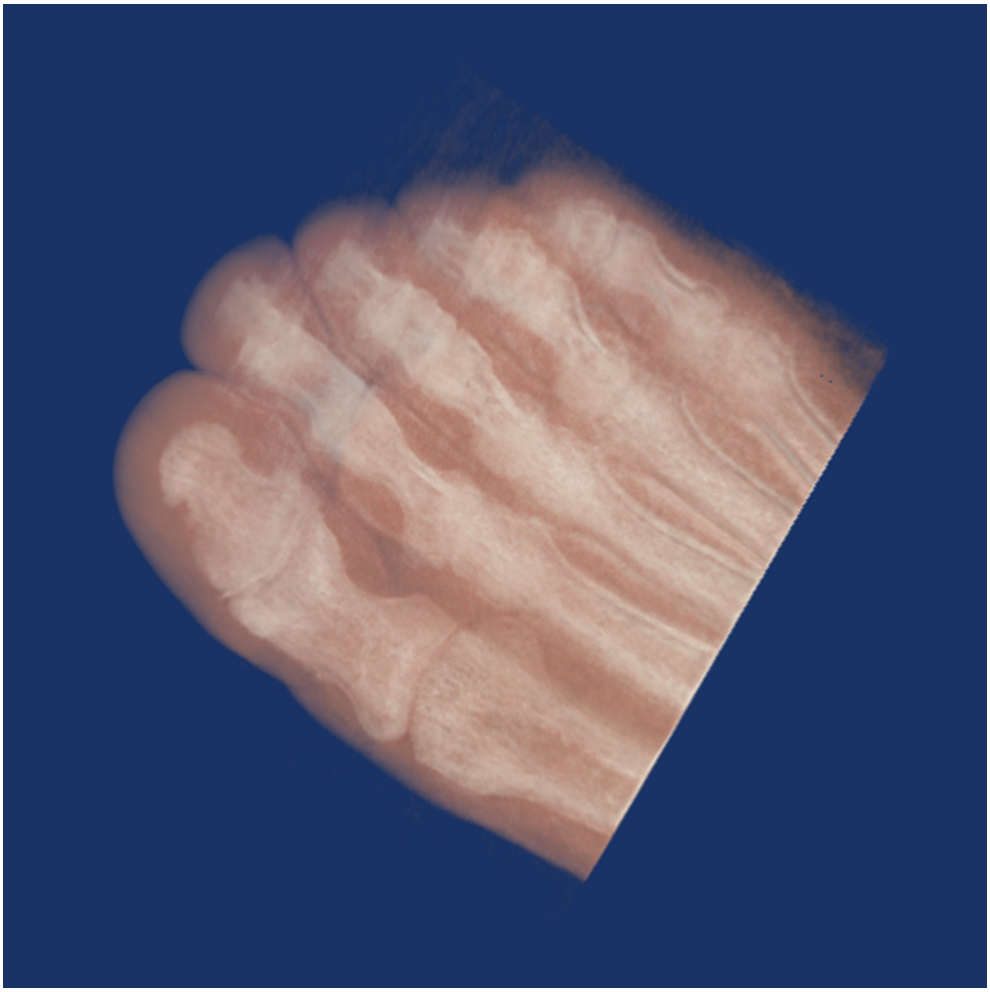
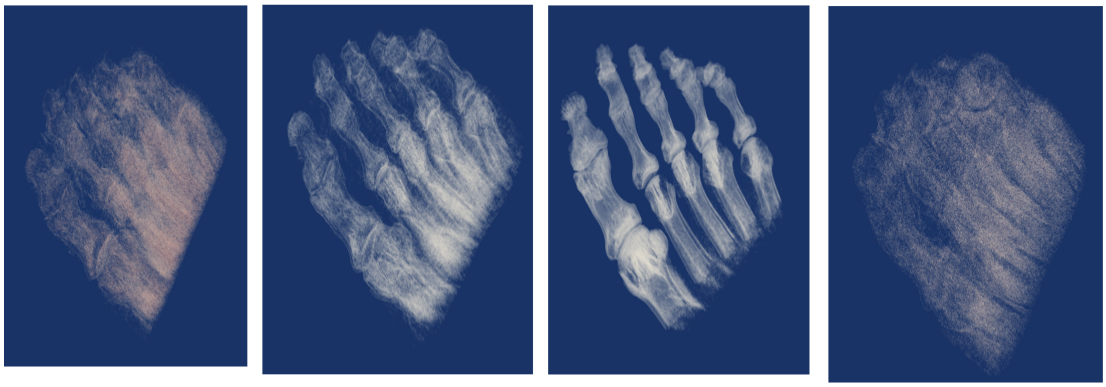
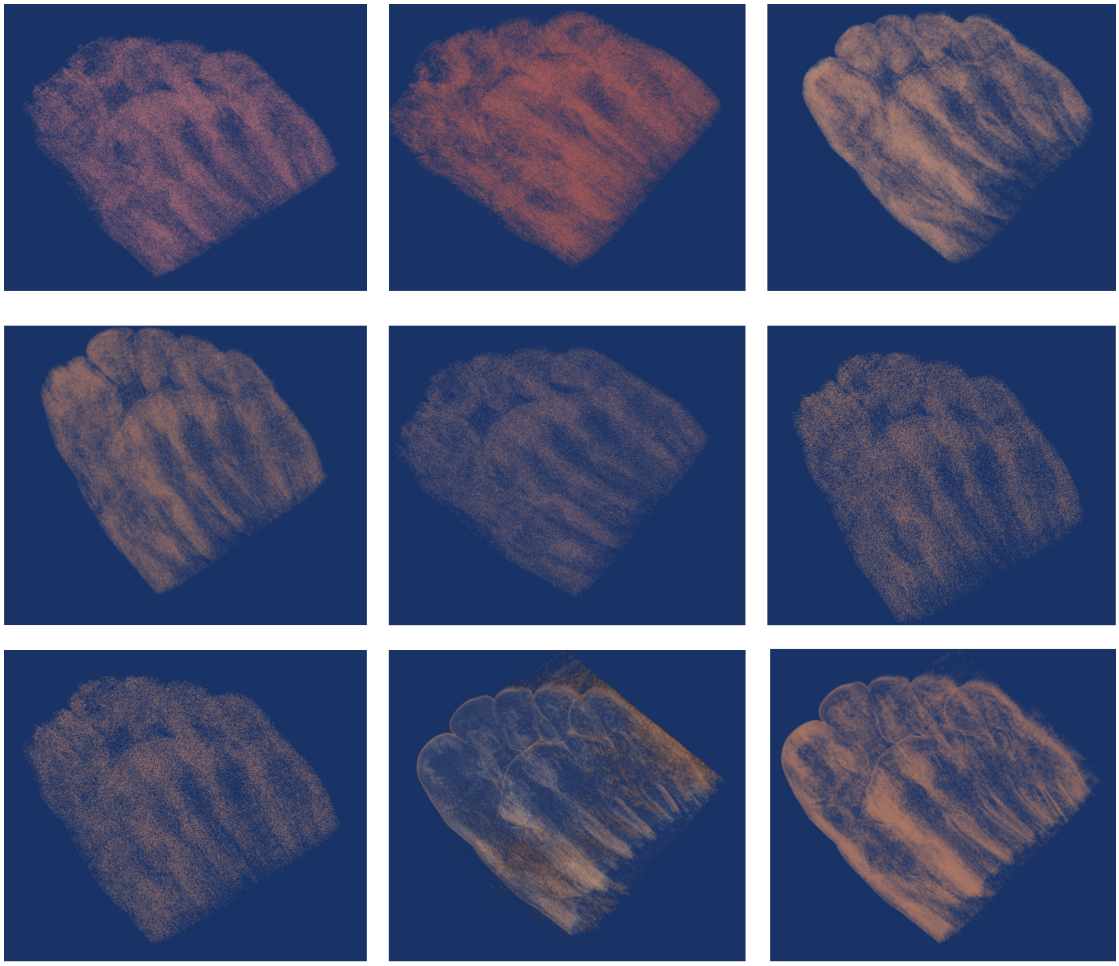


Figure 4.16: Rendering result for the foot dataset.

And the various clusters are distributed as shown in Figure 4.17.



(a)



(b)

Figure 4.17: Various clusters renderings for the foot dataset using the median value.

2- Brian MRI dataset

The rendering result of the brain MRI dataset comes as shown in Figure 4.18.

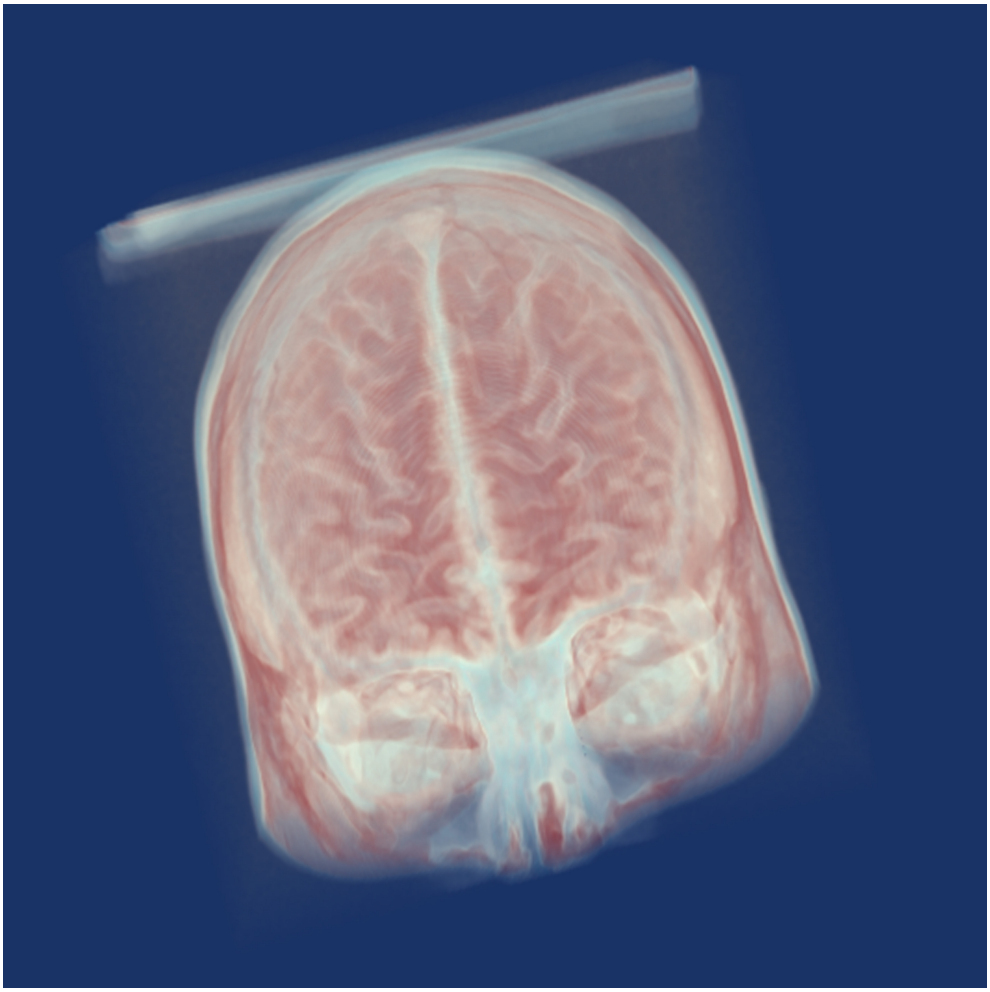


Figure 4.18: Rendering result for the brain MRI dataset .

And the various tissues distributed through the clusters as shown in Figure 4.19.

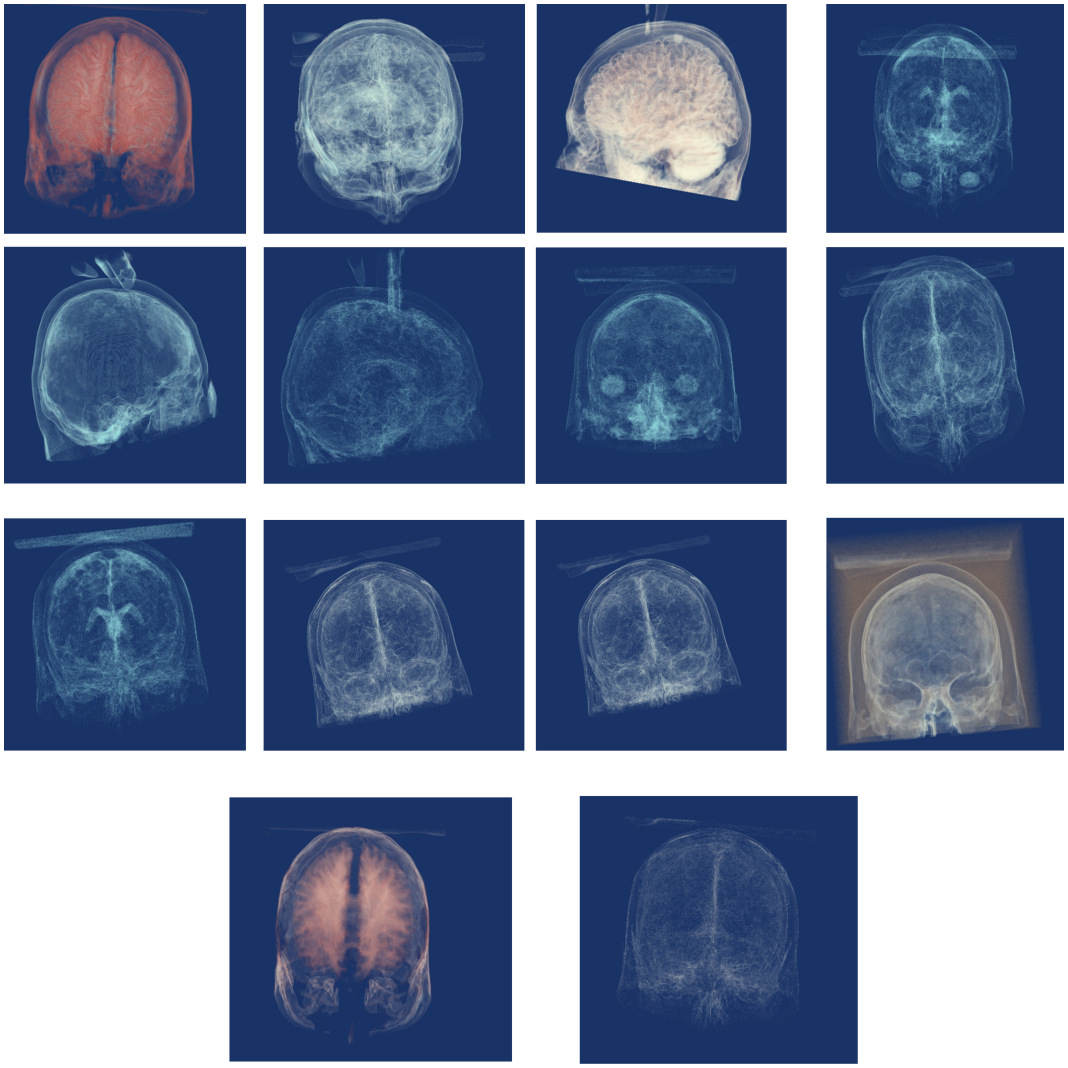


Figure 4.19: Various clusters rendering for the foot data set using the median value.

3- Abdomen dataset

The rendering result of the abdomen dataset using the median values shown in Figure 4.20.

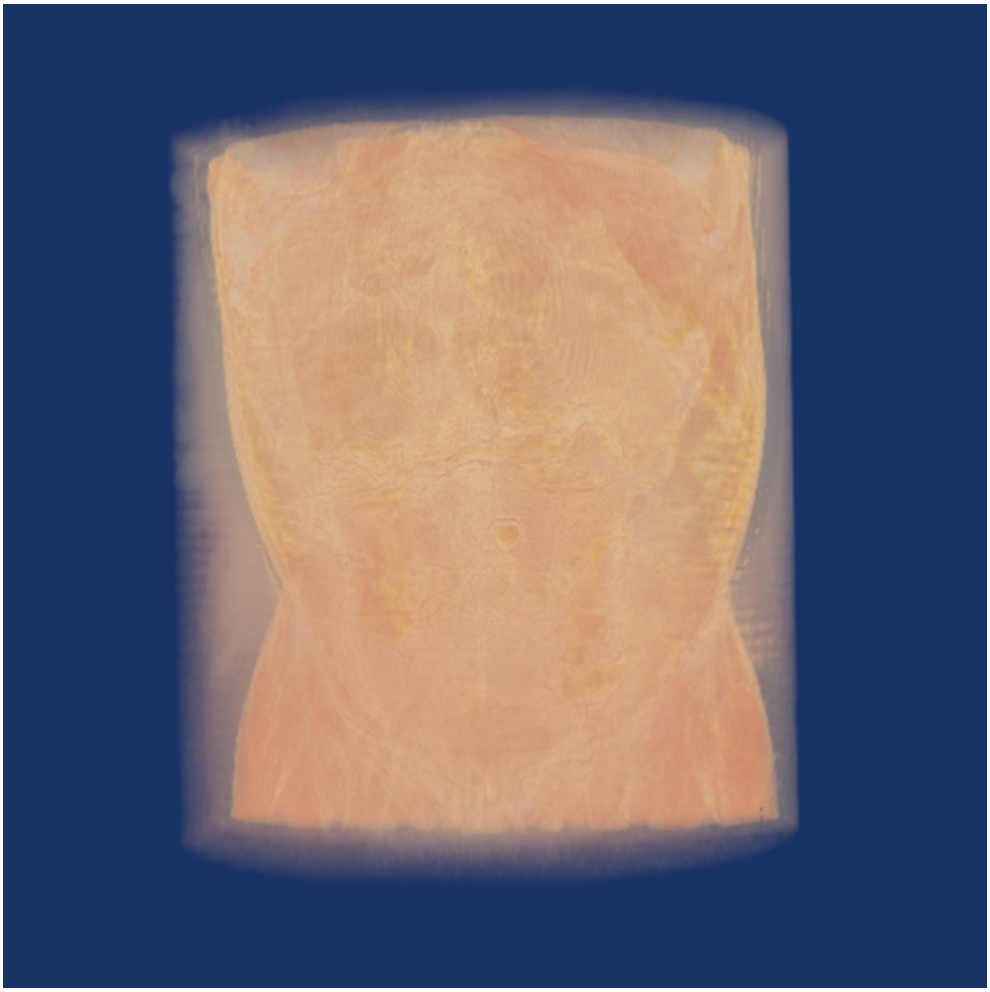


Figure 4.20: Rendering result for the abdomen dataset.

and the various tissues distributed through clusters as shown in Figure 4.21.

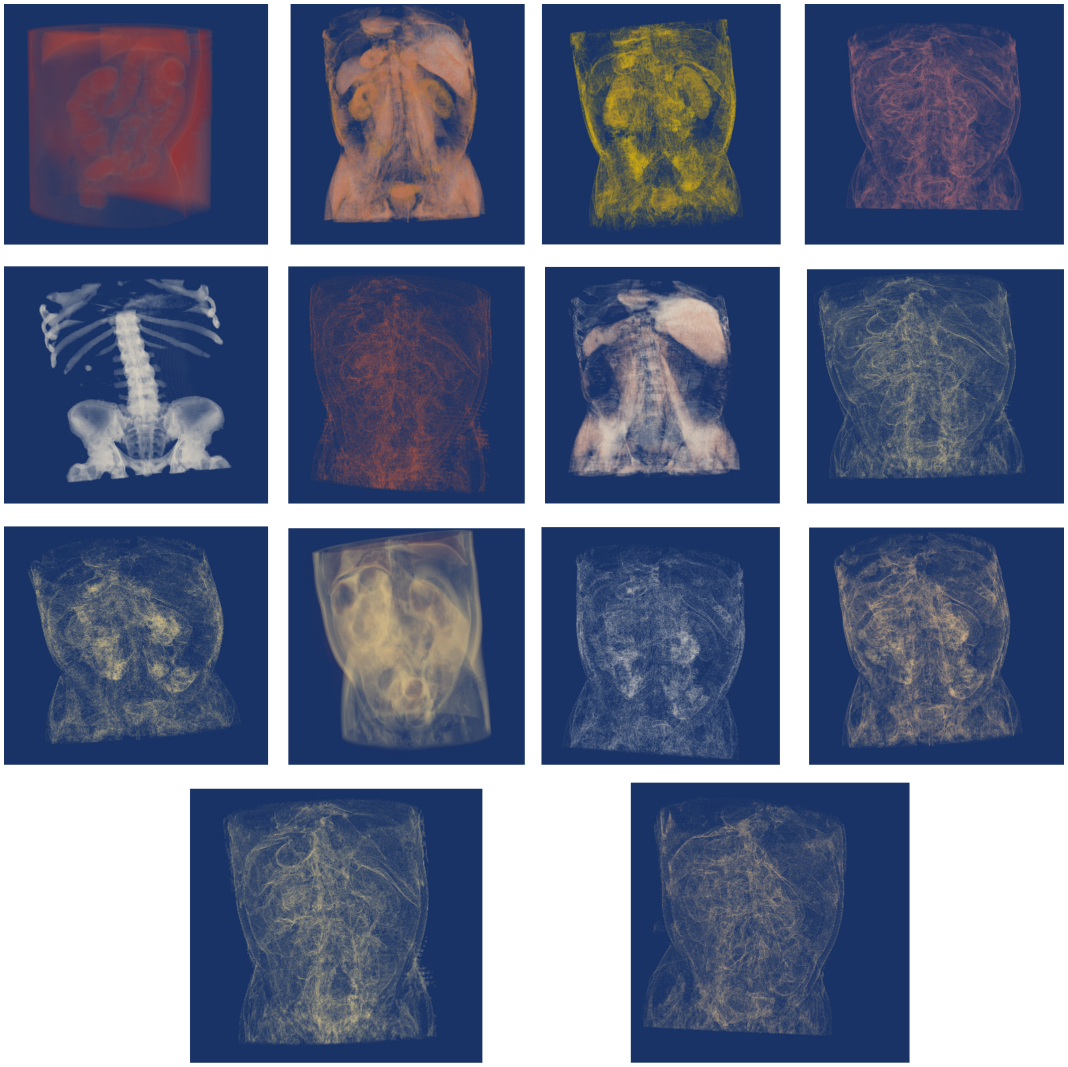
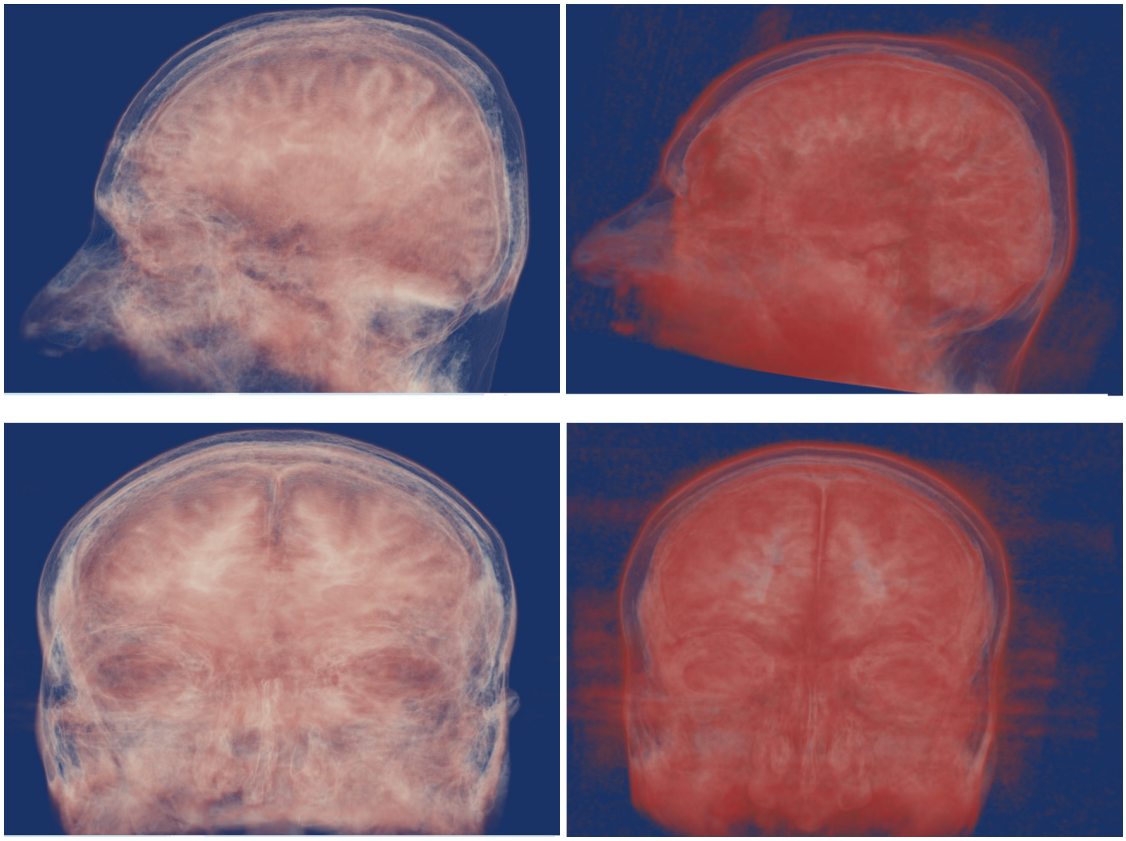


Figure 4.21: Various clusters renderings for the abdomen data set using the median value.

4.4 Method Evaluations

This section represents an evaluation of the system in the following aspects:

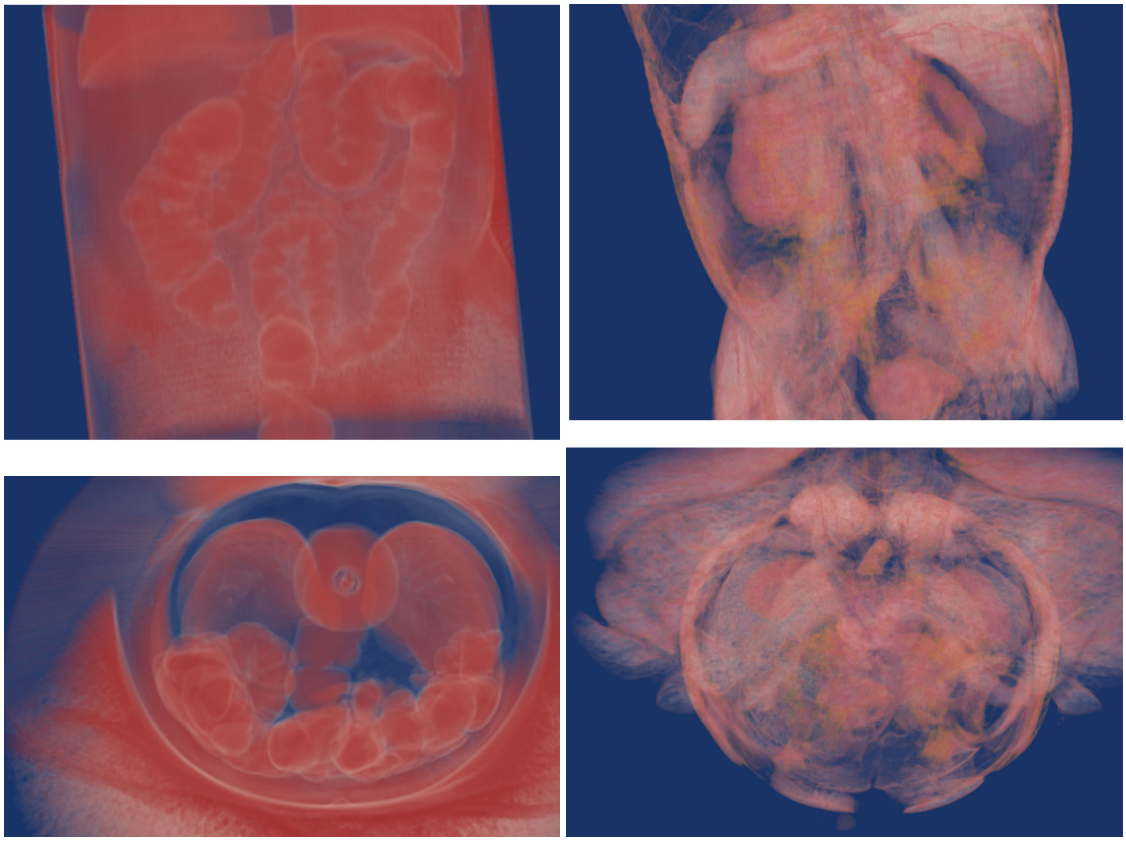
1- Effectiveness



(a) (b)

Figure 4.22: Rendering result of two different clusters of head MRI dataset.

Figure 4.22 shows rendering of the second (a) and third (b) class of the head MRI dataset. The Figure shows how the method precisely classifies the inner and the outer layer of the brain in different clusters which mean the ability of the method to detect and distinguish the various isosurfaces using the proposed similarity matrix creation method.



(a) (b)

Figure 4.23: Rendering result of two different clusters of abdomen dataset.

Also, Figure 4.23 shows the first (a) and third (b) clusters of the abdomen dataset. We can see how the method accurately classifies the colon tissues in a single class and the various organs in another class, this is because the tissues of the colon are different from the tissues of the other organs.

2- Performance

In order to evaluate the performance of the system, we measure the computation time taken for each process of the system for many datasets using a desktop PC has Intel Core(TM) i5-5200U CPU and 8.00 G RAM, and x64-based processor.

In Table 4.1 the *minimum* value been used as preference in 3.2.2, and in Table 4.2 the *Median* value has been used as a preference.

From Table 4.1 and Table 4.2 we can see that the time used for each process is nearly similar for all datasets, this is due to many reasons; the first reason is the standardizing of the image size to 256*256*256, the second reason is the nature of the algorithm in dividing the volume datasets, which makes the number of the tested cells equal for all images, and the final reason is that the classification algorithm converges after the same number of the iterations for all images.

Table 4.1: Performance of the pipeline , measured in terms of the computation time of each step, in the classification the minimum value has been used as a preference.

Computation Table			
Dataset	Similarity Matrix (min)	Classification (sec)	Rendering(sec)
four values dataset	12:55	42:52	2:59
Foot dataset	13:10	49:38	5:28
Brain MRI dataset	13:00	50:05	7:47
Abdomen dataset	13:16	50:57	5:31

Table 4.2: Performance of the pipeline , measured in terms of the computation time of each step, in the classification the median value has been used as a preference.

Computation Table			
Dataset	Similarity Matrix (min)	Classification (sec)	Rendering(sec)
four values dataset	12:55	21:70	2:92
Foot dataset	13:10	22:71	5:94
Brain MRI dataset	13:0	21:98	6:61
Abdomen dataset	13:16	22:39	4:57

Summary

In this chapter we discussed the experiments that we carried out through the research and their results, and we presented an evaluation of the system and how it achieved the goals of the research.

5

Conclusions and Future works

The purpose of this thesis was to present an efficient system to generate the transfer function for direct volume rendering. Throughout the preparation of this thesis we faced many problems and challenges. And after finishing the project, we came up with many conclusions. In this chapter we discuss the challenges and the conclusions of the project. Also, we present the reader with future challenges that deserve research and work to improve them.

5.1 Limitations and Challenges

The challenges that faced us during the project can be summarized in two main points.

The first one is concerned with the classification algorithm, where the affinity propa-

gation algorithm has the disadvantage of generating many exemplars to the same cluster, that yields generating a high number of clusters. To deal with this limitation, we propose to order the generated clusters according to the number of voxels that belong to each cluster and render the highest 15 clusters. This number is user defined and be altered to suit the targeted application of the ultimate rendering.

Another challenge in the classification algorithm is the effect of the preference value, where the use of each preference value generates different results from the other, and the resulting clusters using the minimum value are different from the clusters generated by using the median value. Unfortunately, we could not judge the difference in quality of the two choices and left this for future user-oriented studies.

Finally, we had to deal with the fact that we lack benchmarks for evaluating the resulting volume rendering. This makes the dependence on human perception the base to test the results.

5.2 Conclusions

In frame of this thesis, we have presented an efficient method for automatic generation of a transfer function for direct volume rendering. The proposed method formed a combination of two approaches, where we made use of the similarity matrix created based on the marching cube based algorithm. And we fed this similarity matrix to the affinity propagation algorithm.

In this method, we exploited the positive aspects of both methods. where marching cubs based algorithm has the advantage of extracting different tissues and differentiating them but has the constraint of having to define the number of clusters in advance. On

the other hand, the affinity propagation algorithm has the advantage of extracting an optimal number of clusters without the need to pre-define them.

As a whole, our method was able to efficiently distinguish the various structures of the volume dataset within a reasonable time automatically without the intervention of the user. So, in this method we could generate an efficient automatic design of the transfer function for the direct volume rendering

5.3 Future Works

In future works we wish to extend our project and enhance the affinity propagation algorithm to overcome its limitations of generating a high number of clusters.

Also, we are looking forward to study the effect of the preference value of the affinity propagation algorithm in more depth, in order to better decide which preference value should be used to give a better clustering of the volume dataset, probably proposing new preference options.

On the other hand, we wish to investigate other implementation methods that might lead to the acceleration of the isosurface similarity matrix generation method. A GPU-based implementation is a potential option that we wish to experiment.

References

[1] S. Arens and G. Domik. "ASurvey of Transfer Functions Suitable for Volume Rendering". In: *IEEE/EG International Symposium on Volume Graphics* (2010).

[2] R. Bramon et al. "Information Theory-Based Automatic Multimodal Transfer Function Desighn". In: *IEEE Journal of biomedical and health informatics* (July 2013).

[3] L. Cai et al. "Automatic transfer function design for medical visualization using visibility distributions and projective color mapping". In: *Computerized medical imaging and graphics* (Sept. 2013).

[4] C. Chen. "Information visualization". In: *WIREs Computational Stats* (2010).

[5] COVISE Online Documentation. In: (). URL: https://fs.hlrs.de/projects/covise/doc/html/usersguide/volumerendering/volumerendering.html.

[6] B. J. Frey and D. Dueck. "Clustering by Passing Messages Between Data Points". In: *Science* (2012).

[7] L. Guang and M. Robert. "Volumetric Image Registration of Multi-modality Images of CT, MRI and PET". In: *Research Gate* (2010).

[8] Ascript posted by university of new hampshire. In: (). URL: http://www.celebisoftware.com/Tutorials/volume_rendering/ch1_1.htm.

[9] C. D. Hansen and C. R. Johnson. *The Visualization Handbook*. Elsevier Butterworth–Heinemann, 2005.

[10] W. He et al. “Transfer Functions in Volume Rendering of Lake Water Quality Considering Frequency Distribution”. In: *A preprint at a Research Square journal* (Dec. 2021).

[11] T. Isaac et al. “Low-Cost Parallel Algorithms for 2:1 Octree Balance”. In: *IEEE 26th International Parallel and Distributed Processing Symposium* (2012).

[12] G. Kindlmann. “Transfer Functions in Direct Volume Rendering: Design, Interface, Interaction”. In: *Course notes of ACM SIGGRAPH* (2002).

[13] G. Kindlmann and J. W.Durkin. “Semi-automatic Generation of Transfer Functions for Direct Volume Rendering”. In: *IEEE Symposium on Volume Visualization* (Oct. 1998).

[14] O. Klaas and M.Shephard. “Automatic Generation of Octree-based Three-Dimensional Discretizations for Partition of Unity Methods”. In: *Computational Mechanics* (2000).

[15] YISHA LAN et al. “A Clustering Based Transfer Function for Volume Rendering Using Gray-Gradient Mode Histogram”. In: *IEEE Access* (June 2019).

[16] G. Li et al. “Accuracy of 3D volumetric image registration based on CT, MR and PET/CT phantom experiments Article”. In: *JOURNAL OF APPLIED CLINICAL MEDICAL PHYSICS* (2008).

[17] P. Ljung et al. “State of the Art in Transfer Functions for Direct Volume Rendering”. In: *Computer Graphics Forum* (2016).

[18] B. Ma and A. Entezari. “Volumetric Feature-Based Classification and Visibility Analysis for Transfer Function Design”. In: *2017* ().

[19] R. Mazza. *Introduction to Information Visualization*. 3, 2009.

[20] R. Mesquita and W. Celes. “Robust and effective method for automatic generation of one—dimentional transfer function”. In: *SIBGRAPI Conference on graphics, patterns and images* (2019).

[21] H. R. Nagel. “Scientific Visualization versus Information Visualization”. In: *ResearchGate* (2006).

[22] G. Neilson and B. Hamann. “Techniques for Interactive Visualization of Volumetric Data”. In: *IEEE* (1999).

[23] P.Lücke. “Volume Rendering Techniques for Medical Imaging”. In: (2005).

[24] N. Paenoi and S. Sitjongsataporn. “Automatic Transfer Function Improvement based on Genetic Algorithm”. In: *7th International Conference on Engineering, Applied Sciences and Technology (ICEAST)* (2021).

[25] B. Preim and D. Bartz. *Visualization in Medicine Theory, Algoritms, and Applications*. Morgan Kaufmann Publishers, 2007.

[26] Marc Ruiz et al. “Automatic Transfer Funcrion Based on Informational Divergance”. In: *IEEE Transactions on Visualization and Computer Graphics* (Dec. 2011).

[27] Z. Salah. *Segmentation and Illustrative Visualization of Medical Data*. 2006.

[28] M.S. Abou El Seoud and A.Mady. “A Comprehensive Review on Volume Rendering Techniques”. In: *International Conference on Software and Information Engineering* (Apr. 2019).

[29] P. Šereda et al. “Automating Transfer Function Design for Volume Rendering Using Hierarchical Clustering of Material Boundaries”. In: *32nd SIBGRAPI Conference on Graphics, Patterns and Images* (2019).

[30] O. Sharma, T. Arora, and A. Khattar. "Graph-Based Transfer Function for Volume Rendering". In: *Computer Graphics Forum* (2019).

[31] M. Sr´amek. "20 Years of Volume Rendering". In: *Association for Computing Machinery* (2006).

[32] A. Telea. *Data Visualization Principles And Practice*. A K Peters Ltd, 2007.

[33] P. Thavikulwat. "Affinity Propagation: A Clustering Algorithm for Computer-Assisted Business Simulations and Experiential Exercises". In: *Developments in Business Simulation and Experiential Learning* (2008).

[34] dental Website. In: (). URL: https://familydentalcenters.com/archives/1983.

[35] Matt's$_w$*ebcornerWebsite*. In: (). URL: https://graphics.stanford.edu/~mdfisher/MarchingCubes.html.

[36] T. Zhang et al. "A Clustering-Based Automatic Transfer Function Design for Volume Visualization". In: *Hindawi Publishing Corporation Mathematical Problems in Engineering* (2016).