Meshing Your Body - Finer Sampling in MATLAB

A dissertation submitted to The University of Manchester
for the degree of Master of Science
In the Faculty of Engineering and Physical Sciences

2015

By
Yixian Fang
School of Electrical and Electronic Engineering

Contents

Total words for main body: 9026.

List of Tables

List of Figures

Abstract

Nowadays, there are many concrete applications of the numerical simulation of the electromagnetic wave propagation based on the Finite Difference Time Domain method. One of these applications is the computation of the electromagnetic waves in and on the human body for the development of health care devices.

There is a large amount of interest in biophysical simulation. To run numerical simulations, digital human phantoms are required. The spatial resolution of the digital human phantom has to be flexible depending on the simulation requirement. In order to obtain required spatial resolution, resampling of the digital human phantom is necessary. The normal resampling of the digital human phantom might exaggerate the effect of the staircase of the original one or blur the edges.

This report aims to introduce the project which focuses on accessing the raw data of the images and developing an algorithm to perform finer resampling without bringing new information. Special attention is paid to reduce the staircase by smoothing.

Declaration

No portion of the work referred to in this dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

Firstly, I would like to thank my supervisor Dr. Fumie Costen for her guidance, encouragement and support during this project.

I’m grateful to Buraq Abdulkareem and Kenan Tekbas for their help, advices and patience.

Finally, I would like to thank my family for their endless support over the years. Without them, I may never have reached this stage.

Chapter 1

Introduction

As the biological science and modern medicine develop fast, a wide range of new technologies and methods are used in these fields. The Finite Difference Time Domain method is one of them, which is the most common method of simulating the propagation of waves within space. And it is well suited to model complex and irregularly shaped objects such as the human body [1]. Numerical simulation of the electromagnetic wave propagation based on this method can be used to compute the electromagnetic waves in and on the human body. Using numerical simulation can protect human beings from harmful experiments. On the other hand, numerical simulation is a complex procedure and requires radio environment setting [2].

In order to be able to complete the simulation, digital human phantoms are required for setting the propagation media. With the development of biophysical simulation, digital human phantoms are used frequently and widely. There are different kinds of human body phantoms gathered through different methods, such as Magnetic Resonance Imaging and Computed Tomography [3].

The spatial resolution of the digital human phantom has to be flexible depending on the simulation requirement. When a very precise computation is required, the simulation needs to be operated based on the very fine digital human phantom such as 0.3 mm spatial resolution to achieve high accuracy [4]. If the current spatial resolution of the digital human phantom is not accurate as what is needed, resampling of the digital human phantom is necessary to achieve higher spatial resolution.

In this project, the currently available resolution of the digital human phantom is 1mm. In order to change the spatial resolution to 0.33mm, the digital human phantom should be enlarged by 3 times. However, the normal resampling methods might exaggerate the effect of the staircase of the original 1 mm resolution digital

human phantom.

1.1 Aims and objectives

This project is to achieve finer sampling in MATLAB. The data from an image can be represented through a matrix. And MATLAB is a kind of software which is particularly efficient in processing matrix operations [5].The objectives of this project are listed below:

- Understand the principles of resampling and smoothing.
- Develop an algorithm for resampling and smoothing for 2D images.
- Implement the algorithm and verification of the code with the test 2D case.
- Apply the smoothing algorithm to a slice of brain data.
- Adapt the algorithm to smooth 3D voxel data.
- Implement the algorithm and verification of the code with the test 3D case.
- Apply the smoothing algorithm to the whole brain data.
- Optimize the algorithm from the viewpoint of accuracy and CPU time.

This project aims to up-scale and smooth the digital human phantom in both two dimensions and three dimensions for producing high spatial resolutions. The main aim is to develop a smoothing algorithm that can reduce the sharp corners while bringing no new value into the image data, since every pixel value in the image represents a specific human tissue and new values will make no sense. Therefore, common smoothing methods can not fit this project well because they might bring some new values to the image and cause meaningless output.

1.2 Dissertation overview

To understand the problem and the objectives of this project, some background knowledge is required. The PGM format and the representation of human body tissues are discussed in chapter 2, with the simple introduction of digital human phantoms.

Chapter 3 introduces some current research with simple examples, including previous standard methods for rescaling and smoothing, and explains the reason why they are not suitable for this project.

Chapter 4 and chapter 5 are the most important parts of this thesis. Chapter 4 discusses the adaptation of the chosen modal filter method to get better smoothing results and fit the requirements in both 2D and 3D. The memory problem with MATLAB in processing 3D images is also mentioned in this chapter. Chapter 5 concerns the application in real human brain data and shows the resampling results of 2D and 3D data. Then there is a simple analysis of the results.

Optimization is discussed in chapter 6 from the viewpoint of accuracy and CPU time. The parallelism processing method is concerned in this chapter.

Finally, conclusions of this project are summarized in chapter 7, including potential improvements and future work.

Chapter 2

Background Knowledge

2.1 Digital human phantoms

Digital human phantoms are models of the human body used in computational analysis [6]. They are currently being increasingly implemented in the teaching of anatomy [7]. With the use of the digital human phantoms, many kinds of simulations and experiments can be operated without doing harm to living objects.

There are a variety of different kinds of models for digital human phantoms. Figure 2.1 shows a model which is constructed from Magnetic Resonance Imaging scans taken of live volunteers, with voxel sizes of $2 \times 2 \times 2$ mm [8].

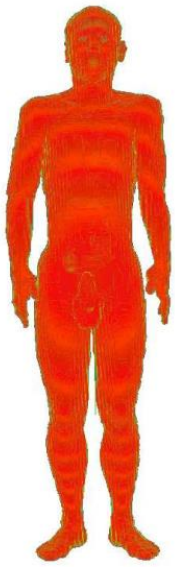


Figure 2.1: A model constructed from Magnetic Resonance Imaging scans taken of live volunteers [9].

When a human body is scanned using Magnetic Resonance Imaging method, each scan shows the cross section of the human body orthogonal to the direction of the backbone. The data used in this project is scanned human brain data. The Magnetic Resonance Imaging scanned image is segmented and the segmentation can be used

to identify a tissue, which means each pixel has a tissue number in this kind image. Different numbers represent different human tissues. Therefore, the scanned image can be replaced with a stream of integers [10]. This stream of integers is stored in a file, which can be converted into PGM format by adding some appropriate line headers.

2.2 PGM format

The 2D and 3D data used in this project is in PGM format. The portable gray-map format (PGM) is an image file format which is easily exchanged between platforms. A PGM data file consists of numbers of gray between black and white. It has 256 kinds of colors in gray scale, which are from 0 to 255. The maximum value of the data is presented as white, while the minimum value is presented as black [11]. A data interpretation is shown in Table 2.1. The result of this data file is shown in Figure 2.3. This result is obtained with MATLAB.

```
P2
24 8
8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8
```

Figure 2.2: A file with PGM format.

Table 2.1: The data interpretation of PGM format

P2	This means the data is pgm format.
24 8	The x-axis has 24 pixels and y-axis has 8 pixels.
8	The effective maximum value in the data is 8.
1… 2… …8	These are the actual data at each pixel.

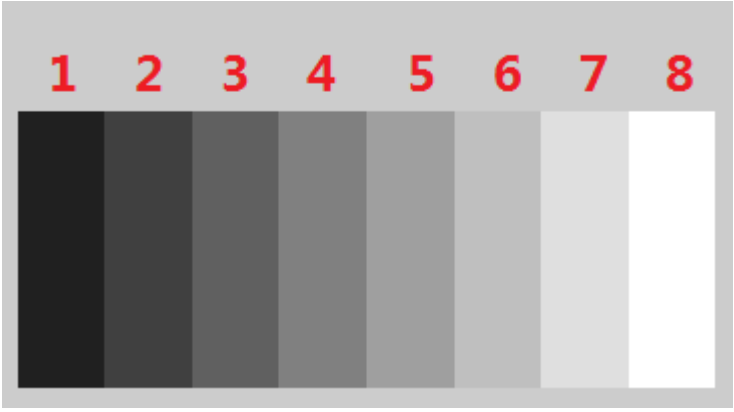


Figure 2.3: The result of the example data file in MATLAB.

When the effective value is set to a value higher than the maximum value of the whole data, the entire data is presented as a low value data [12]. If the value is set to be 15, the appearance is like Figure 2.4 (a). When the effective value is set to a value lower than the maximum value of the whole data, then the data values above this effective value are presented as white. Figure 2.4(b) shows the result when the value is set to be 5.

(a)

(b)

Figure 2.4: The results of two examples with inappropriate effective values.

Therefore, the effective value can't be set to a lower value and there is no need to set it to a higher value. For the 2D and 3D data used in this project, the values in the

images are representing different human brain tissues. All the images used in this project are in PGM format, and their effective maximum value is 255.

2.3 2D and 3D data

2D graph is a planar graph that can be embedded in the plane. Each point of the 2D graph has its x-axis position and y-axis position. Whereas pixels in 2D graphic have the properties of position, brightness or color, 3D pixels add a depth property that indicates where the point lies on Z-axis [13].

The scanned digital human phantom used in this project is generated by scanning a human body every 1mm using magnetic resonance imaging method. The size of the whole 3D data is 265×490×1682 voxels, each layer is a slice of 2D data whose size is 265×490 voxels. 3D data is constructed with many layers of 2D data. The 2D image is like the example shown in Figure 2.5. Figure 2.6 is a 3D image built with 10 layers of 2D images, and Figure 2.7 shows the whole 3D human brain data.

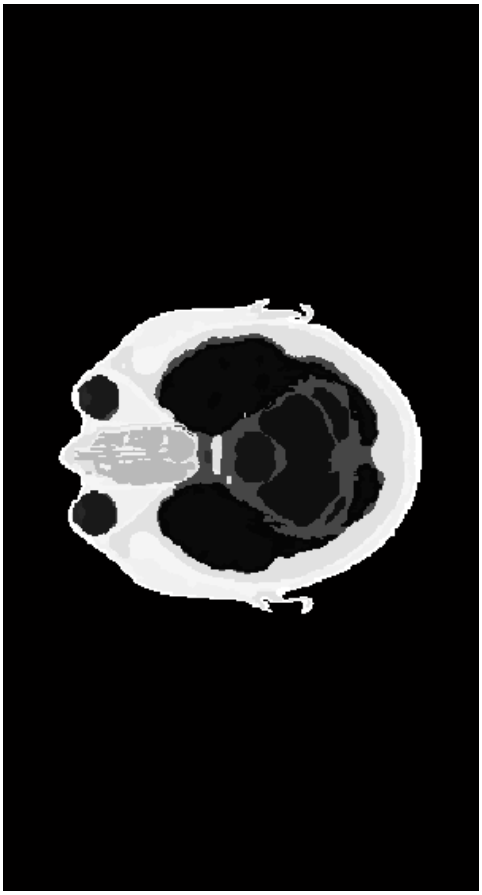


Figure 2.5: A slice of 2D data used in this project whose size is 265×490 voxels.

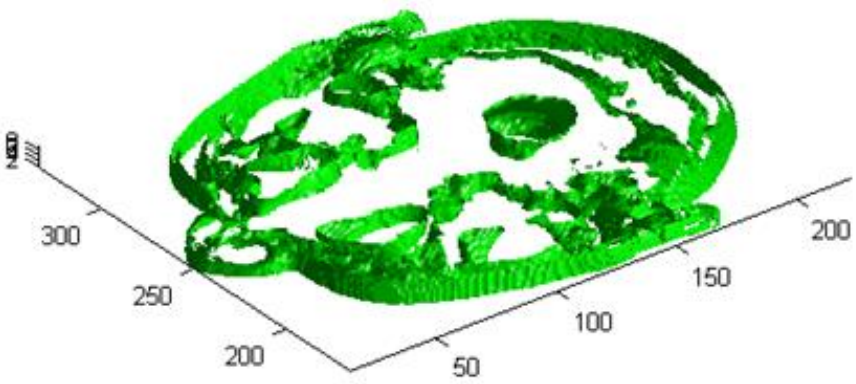


Figure 2.6: 3D image built with 10 layers of 2D images.

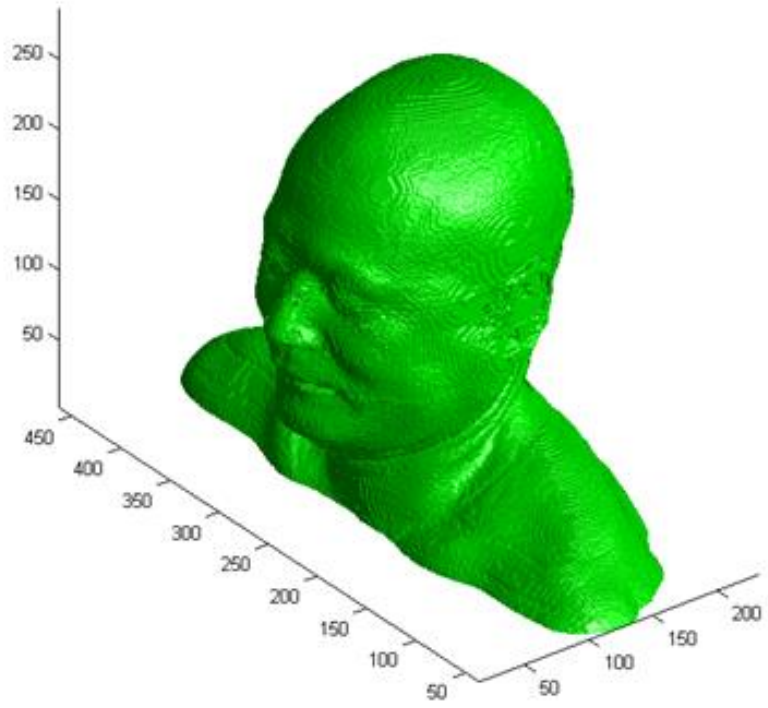


Figure 2.7: The whole 3D human brain data.

Since the whole 3D data needs a large memory space to process, only a brain part is processed in this project.

2.4 Up-scaling and smoothing

A variety of fast and powerful algorithms are developed to operate on image data. There are various filters which are used to reduce noise on images, and a lot of scaling techniques are necessary to change the image sizes while retaining as much information as possible or bringing as little noise as possible [3]. Many scaling techniques are developed to decrease image sizes. However, this project focuses on increasing the size of the image and smoothing it while providing no new information in the up-scaled image.

For the data used in this project, the size of the pixel is 1mm×1mm. In some applications, the spatial resolution is required to be 0.1~0.3 mm. Since high spatial resolution is important to detect some details in an image, such as the edges of structures and margins of tumors [14]. In order to change the spatial resolution, the digital human phantom has to be up-scaled and smoothed.

2.4.1. Up-scaling

Figure 2.8 shows a simple example of up scaling an image. The original image has (2×2) pixels. After it is increased by 3 times, the up-scaled image has (6×6) pixels. That means every pixel in the original image is replaced by 9 pixels which have the same value as the original pixel [15].

0	1
1	0

0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
1	1	1	0	0	0
1	1	1	0	0	0
1	1	1	0	0	0

Figure 2.8: The original image and the image which is up scaled by 3 times [3].

When an image is up-scaled using a normal method, such as the nearest neighbor interpolation method, there can be a lot of sharp steps and corners at the edges such as shown in Figure 2.9. The original image is enlarged 6 times.

Figure 2.9: A part of image enlarged 6 times with nearest interpolation method.

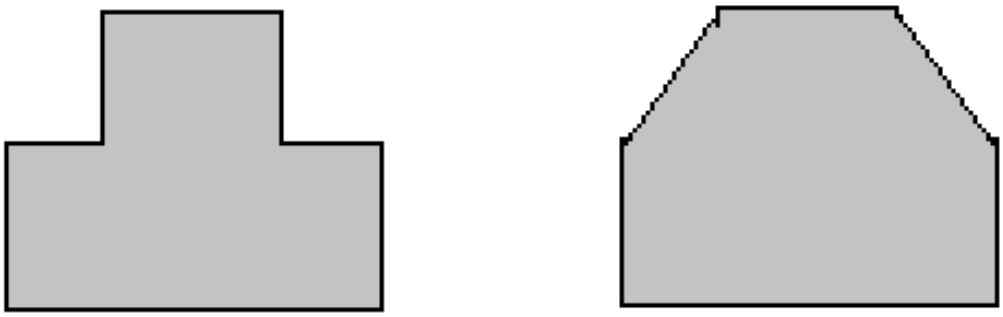
Since these sharp corners are exaggerated during up-scaling procedure, they are unnecessary information, hence produce a very jagged image such as the example in Figure 2.9 and affect the image quality. In order to reduce these sharp steps, smoothing is required.

2.4.2. Smoothing

The problems caused by up-scaling can be reduced through smoothing techniques. Smoothing is a very important image processing method in which some data points are modified to reduce individual points and smooth the corners and sharp steps on the image. The aim of smoothing is to obtain slowly changing values and replace

those sharp corners with smooth curves [16].

Imagine extracting one single sharp corner and smoothing it, as the example shown by Figure 2.10. The original one has a sharply changing edge line. After smoothing, the original one is replaced by an edge line which changes slowly.



(a) Original (b) After smoothing

Figure 2.10: A simple example of smoothing a single sharp corner [3].

In order to achieve the expected smoothing effect in Figure 2.10, the values of some pixels in the original image are required to be replaced by more appropriate values [3]. The changes of raw data are shown in Figure 2.11.

0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1

0	0	0	1	1	1	0	0	0
0	0	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1

(a) Original values (b) Values after smoothing

Figure 2.11: Changes of the pixel values to achieve smoothing result [3].

As previously discussed, each pixel in the digital human phantom has a tissue number and different numbers represent different tissues. All the values in the original image have their own meanings. Thus, smoothing method chosen for this project should not produce any new values, because new values are meaningless and represent for nothing. In the above case for example, assuming ‘0’ represents fat tissue and ‘1’ represents bone. Since the original data only has ‘0’ and ‘1’, the data

after smoothing should also only have ‘0’ and ‘1’. If some other numbers, such as ‘0.5’ or ‘2’, were brought into the data during smoothing, the new numbers would represent some tissues that were not exist in the original data and would cause problems.

Chapter 3

Previous Research and Standard Smoothing Methods

Generally, digital image processing means a discrete image is processed in some way to produce a new discrete image. Digital images are constituted of a rectangular grid of evenly spaced pixels. Resampling is used to create a new version of the image with a different width and height in pixels. It also changes the number of pixels. When images are up-sampled, the number of pixels increases [17].

There are many different up-scaling schemes. The simplest one is nearest neighbor interpolation. Each pixel in the new image is given the value of the nearest pixel in the original one. But as mentioned previously, the nearest neighbor interpolation produces a blocky result with many sharp corners. More advanced methods compute new pixels according to more of the surrounding pixels to produce a better result.

MATLAB provides resize function to change the size of an image, with options to decide how the resampling is performed. Different methods offer different kinds of balance of computational cost against visual appeal [9].

3.1. Bilinear and bicubic interpolation

Bilinear and bicubic interpolation methods are two standard smoothing methods used during up-scaling images. Both of them are used for interpolating data points on a 2D regular grid.

Bilinear interpolation is an extension of linear interpolation. It takes (2×2) pixels into account. The first step is to perform interpolation in one direction, and then in the other direction [18]. Figure 3.1 is a simple example of bilinear interpolation. In this case, point R is the location to be interpolated, while P1, P2, P3 and P4 are four nearest data points from raw data. The attributes of the weighted average of the four nearest values is applied to point R [19].

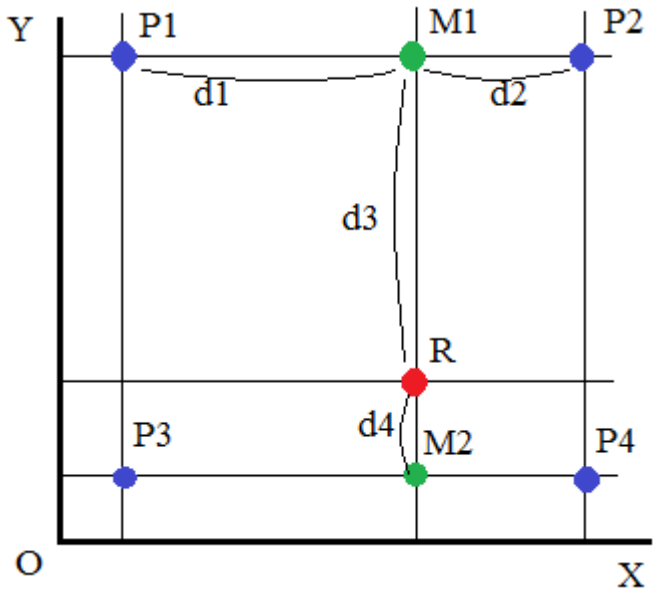


Figure 3.1: A simple example of bilinear interpolation.

The first step is to perform the horizontal interpolations M1 and M2. The distance between P1 and M1 is defined as d1, while the distance between P2 and M1 is defined as d2. The values of M1 and M2 can be calculated from function (1) and (2), respectively.

$$f(M1) = \frac{d1}{d1+d2}f(P1)+\frac{d2}{d1+d2}f(P2) \quad (1)$$

$$f(M2) = \frac{d1}{d1+d2}f(P3)+\frac{d2}{d1+d2}f(P4) \quad (2)$$

$$f(R) = \frac{d3}{d3+d4}f(M1)+\frac{d4}{d3+d4}f(M2) \quad (3)$$

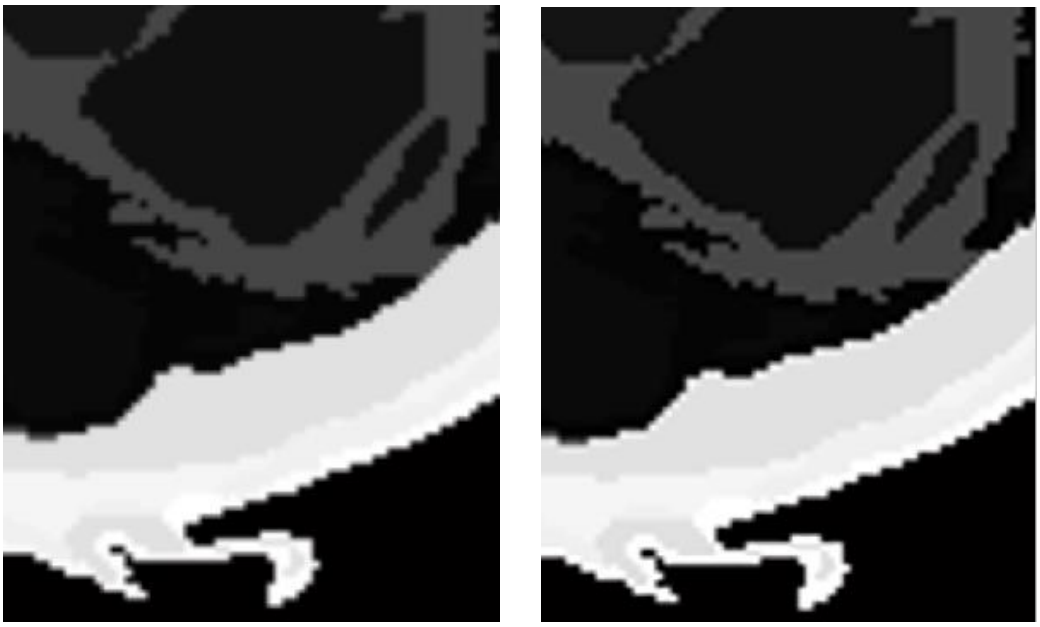
After two horizontal interpolations are made, the result of point R is achieved by vertical interpolation, which can be calculated with function (3).

Bicubic interpolation takes (4×4) pixels into account. It considers 16 surrounding pixels, which is 4 times than the bilinear interpolation. Thus, comparing with the bilinear interpolation, bicubic interpolation produces smoother results.

Figure 3.2 shows the effects of bilinear and bicubic interpolation. The original one is enlarged by 6 times and smoothed with these two methods respectively. Although the results are not as blocky as the result of nearest interpolation method, and there are less sharp edges and corners, these methods are still inappropriate for this project.

Because both bilinear and bicubic interpolation bring new pixel values while blurring sharp corners.

For the case of Figure 3.2, the original image only has 15 pixel values. While the bilinear smoothed image contains 253 values and the bicubic smoothed one has 256 values. The new values in the resampled images are meaningless and cause blurred results.



(a) Bilinear (b) Bicubic

Figure 3.2: The enlarged image is smoothed with bilinear and bicubic method respectively.

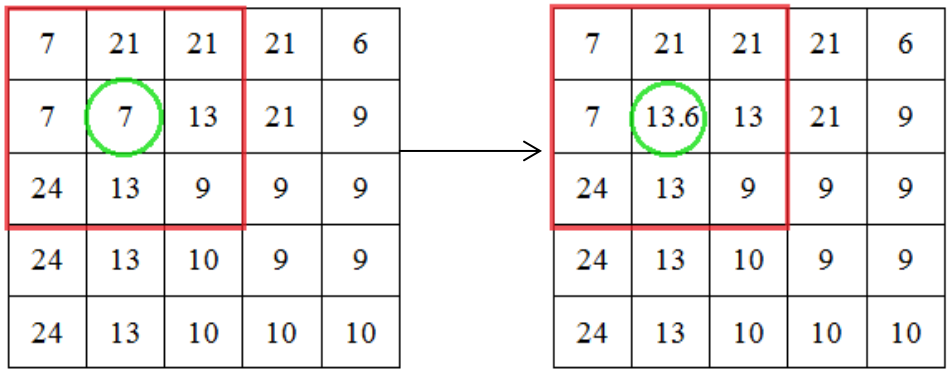
The original image only has 15 pixel values, which means only 15 kinds of human body tissues exist in the image. The results resampled with bilinear and bicubic methods produce 238 and 241 new values representing some tissues don't exist in the original image and cause blurred and meaningless results. Therefore, the bilinear and bicubic method can't be chosen for this project.

3.2. Convolution filter based

Convolution filter based methods are common methods used for smoothing. The convolution filter method operates on a value and its surrounding elements to obtain a new value and uses the new value to replace the original one [3]. There are a lot of

different convolution filters and among these convolution filters, the mean filter and the median filter are used most frequently.

The mean filter is very simple to operate. It calculates the average result of the values in a certain region and replaces the center pixel of this region with the calculated result [15]. Figure 3.3 shows an example of the operation with a 3×3 mean filter. The region in the red square is the 3×3 mean filter model. The value of the center pixel in this region, which is circled by the green line, is changed from 7 to 13.6 after smoothing with the mean filter. The value 13.6 is obtained by averaging the 9 values in the original 3×3 region.



(a) Original (b) After filtering

Figure 3.3: The example of applying a 3×3 mean filter to change a pixel value.

According to this example, it's obvious that the value 13.6 is a new value which doesn't exist in the raw data. The new value represents nothing and has no meaning. Using mean filter has a high possibility of producing new values. This is the problem of applying the mean filter or any other weighted averaging filters [15]. Therefore, the mean filter can't be chosen for this project.

The median filter is non-linear digital filtering technique. The main idea of the median filter is to replace each pixel value with the median value of its surrounding values. When the median filter is applied to the same raw data as the original one shown in Figure 3.3, the first step is to arrange the 9 values in order. The order of these 9 values in the red region should be '7 7 7 9 13 13 21 21 24'. Then the median value 13 is chosen to replace the original value. The result is shown as Figure 3.4.

The center pixel of the model is changed from 7 to 13. It's obvious that median filter uses a value from the raw data to replace the original one, which means it doesn't bring new values into the image.

7	21	21	21	6
7	7	13	21	9
24	13	9	9	9
24	13	10	9	9
24	13	10	10	10

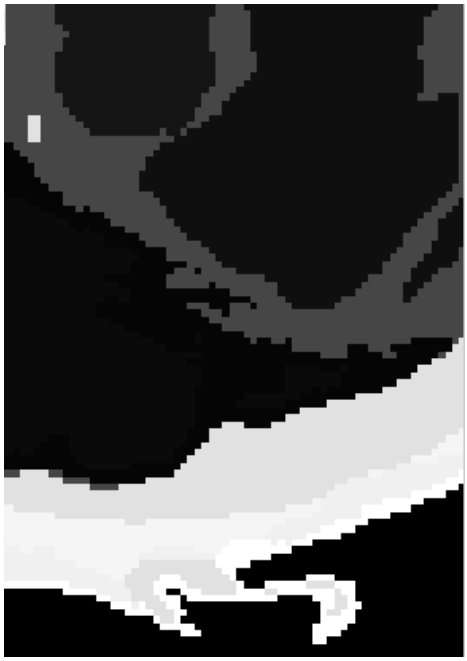
(a) Original

7	21	21	21	6
7	13	13	21	9
24	13	9	9	9
24	13	10	9	9
24	13	10	10	10

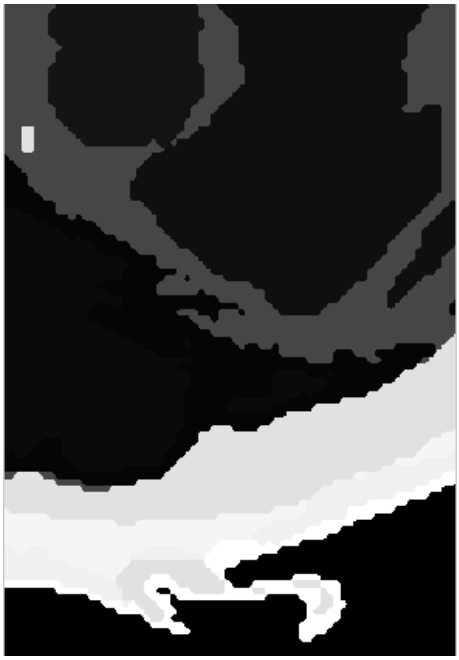
(b) After filtering

Figure 3.4: The example of applying a 3×3 median filter to change a pixel value.

Figure 3.5 shows the smoothing effect of applying a 3×3 median filter based method to digital human phantom data. According to Figure 3.5, the median filter based method produces reasonable results, but the smoothing effect is not good enough. The edges still have sharp corners and staircases on them.



(a) Original



(b) After filtering

Figure 3.5: The smoothing effect of applying a 3×3 median filter based method to digital human phantom data.

The median filter is normally used for removing impulsive type noise from a signal [20]. For digital image processing, the impulse noise is also known as salt and pepper noise, which are minimum or maximum values in a digital image [21]. However, in most cases, the sharp corners on the edges are not caused by extreme values. Thus, the median filter is not appropriate for this project.

3.3. Look-up table based

Look-up table method has a principle which is different from the convolution filter based methods. The convolution filter based methods scale the image first and then consider the contents of the local region for a large amount of new pixels [9]. The convolution filter based methods operate up-scaling and smoothing separately. The look-up table method uses the arrangement of similar pixels around a certain pixel in the original image to determine the layout of this region in the up-scaled image, which means it smooths the image while up-scaling it. The look-up table method operates up-scaling and smoothing at the same time.

There are different schemes for operating look-up table method. The hqnx family of scalers created by Maxim Stepin [22] is one of them. According to this scheme, the look-up table based filter compares the 8 surrounding pixels with the center pixel to define each as being either near or distant based on how similar the colors are. The distances of the surrounding pixel values to the center pixel are irrelevant in building the phantom. A value that is one pixel away is as different as a value 20 pixels away, which means the connection of the values is not dependent on their distance. The connection of the values only depends on the colors. The color of a pixel depends on the value of it. Therefore, the only pixels considered as similar are those with exactly the same value. Since each pixel has 8 surrounding pixels, and each of them may have the same value with the center pixel or not, there are 256 (2^8) potential combinations of similar pixel arrangement. When an image is up-scaled and smoothed with the look-up table method, the first step is to define the shape of the output region according to the original pixel layout. Then the original pixel data

is used to combine with this shape to produce the final output region. Therefore, the layout of the final pixels is dependent on the original data and the scheme chosen for the look up table [9].

Figure 3.6 shows an example. The original data is used to select an output shape firstly. This shape is then combined with the original data to produce the layout of pixels in the output region [9].

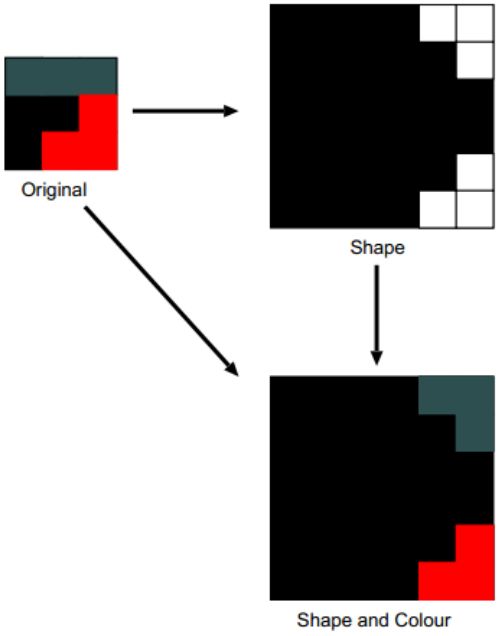
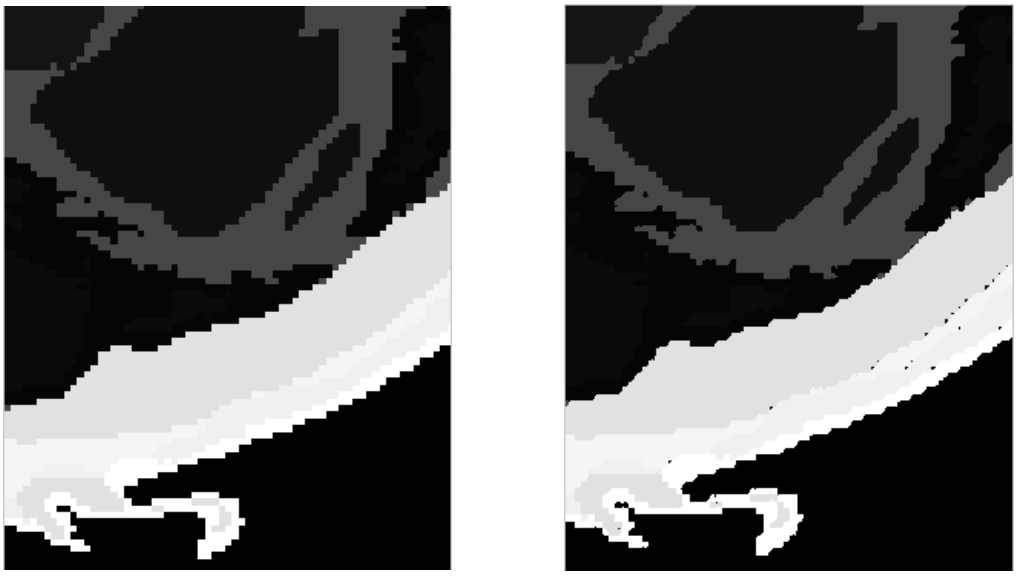


Figure 3.6: An example of the look-up table process [9]



(a) Original (b) After filtering

Figure 3.7: The smoothing effect of applying look-up table method to digital human phantom data.

Figure 3.7 shows the smoothing effect of applying the look-up table method to digital human phantom data. The result is good. There are less sharp corners while no new information is brought in, but there is some obvious pepper noise on the resampled image.

Chapter 4

Modal Filter Method

As mentioned previously in chapter 3, there are some common smoothing methods, but generalized smoothing risks the possibility of removing necessary sharp edges, providing new information or pepper noise in the resampled image. However, there are some methods that can be adapted to provide more satisfactory results.

The modal filter method [3] uses the neighborhood to restrict the output to a value which is from the model. Therefore, no new information is created. This method is appropriate for this project.

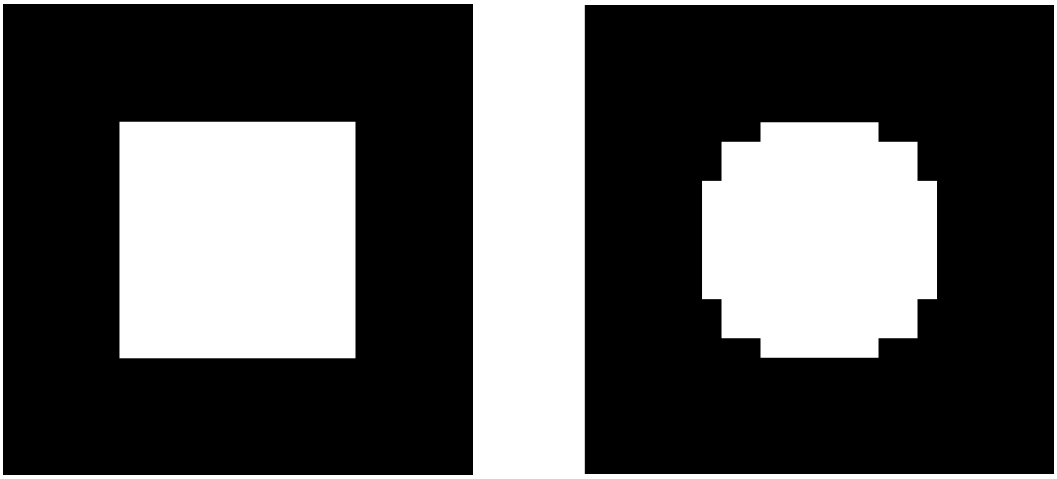
4.1 Modal filter method in 2 dimensions

The key point is to restrict the output to a value that is known to exist in the model and is appropriate for the region which is required to be smoothed. To achieve the expected output, a neighborhood mode is applied. The main idea of this neighborhood mode is to count the existing pixel values in the model region and find the most common one to replace the pixel of interest [3].

When a 5×5 modal filter is used to smooth a section of an image, every time there are 25 pixel values to be counted. Since different values represent different tissues, the most common value is the tissue which appears most frequently in this 5×5 region [3]. This most common tissue is considered to be the expected output and it is used to replace the pixel of interest.

This method is like the convolution filter based method which scales the image first and then considers the contents of the local region for a large amount of new pixels [9]. When the image is enlarged, there are many jagged edges. To make them smooth is to create circles on a grid of squares. And turning a square into a circle means the pixels at its corners should be replaced by more appropriate pixel values

so that these corners can be removed. Figure 4.1 shows an example of removing corners from a square to make it like a circle. The original square with white color has 12×12 pixels whose values are '1', and the black background are pixels whose values are '0'. After the original image is smoothed by a 7×7 modal filter, 5 pixels from each of the 4 corners are changed from '1' to '0'. After smoothed with modal filter, the white square looks more like a circle. The sharp corners are turned into some smaller corners which are smoother than the original corners.



(a) Original image (b) Image smoothed with modal filter

Figure 4.1: An example of removing corners from a square to make it like a circle.

When a pixel is enlarged by 6 times, it is turned into a square of 36 pixels. The largest circle that can fit within this square should have $\pi \times 3^2 = 28.27$ pixels [9]. The smoothed result of a square should be symmetrical, which means each corner should remove equal amount of material. If each corner removes 1 pixel, there are 32 pixels left. Removing 1 pixel converts the corner into two smaller corners, while removing 3 or 5 pixels converts the corner into three smaller corners, as shown in Figure 4.1. Since there are 4 corners of the square at first, there would be 8 smaller corners after removing 1 pixel from each of 4 corners. If 3 or 5 pixels are removed, there would be 12 smaller corners. In this way, the sharp corner is smoothed.

If a corner is surrounded by a single media, removing single pixel from each corner can be achieved with a 3×3 modal filter. Removing 3 pixels requires a 5×5 modal filter. For the first image in Figure 4.2, the pixel in green circle has a value of '1'.

When this pixel is smoothed with a 5×5 modal filter, the value of this pixel is changed from ‘1’ to ‘0’, since there are 16 pixels with a value of ‘0’ and 9 pixels with a value of ‘1’ in the 5×5 region, which means ‘0’ is the most common value in this region and should be chosen to replace the original pixel value in the green circle. When the 5×5 modal filter slides to the next pixel, the 5×5 modal filter region is like the red region of image (b) in Figure 4.2. It has 13 pixels with a value of ‘0’ and 12 pixels with a value of ‘1’. The value of the center pixel should also be replaced by ‘0’. However, when the 5×5 modal filter slides to the third pixel, the 5×5 modal filter region is like the red region of image (c) in Figure 4.2. There are 10 pixels with a value of ‘0’ and 15 pixels with a value of ‘1’. Therefore, ‘1’ is the most common value, which means the center pixel value shouldn’t be changed.

After the whole data is smoothed with the 5×5 modal filter, the square is converted into the one shown in Figure 4.2 (d). Each of the 4 corners removes 3 pixels.

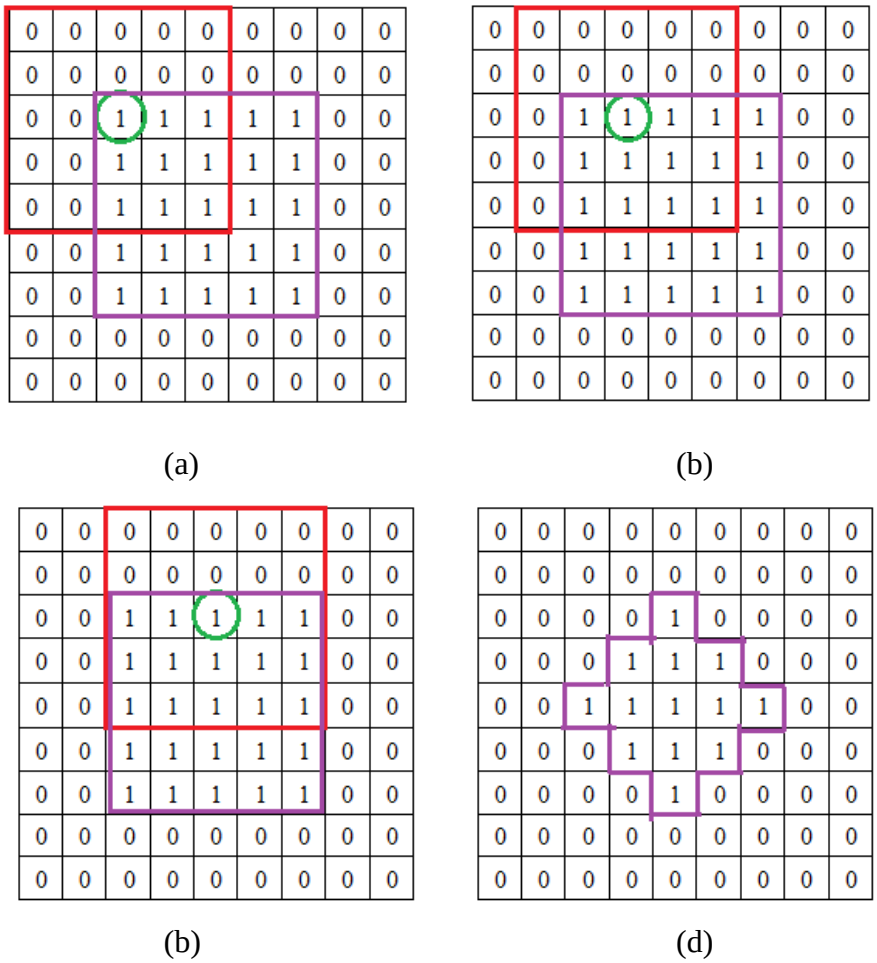
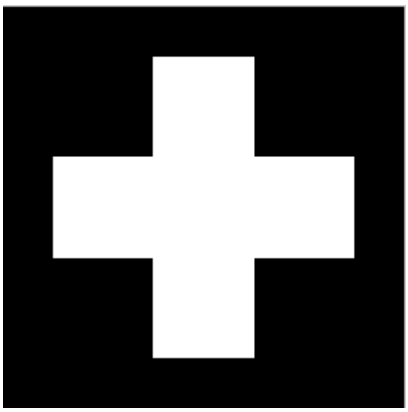


Figure 4.2: The procedure of removing 3 pixels at corners with a 5×5 modal filter.

Figure 4.3 shows the original 2D image. This image is a 2-dimensional image case used for testing. It is a 8×8 image with only 2 pixel values. After enlarged by 6 times, the image has 48× 48 pixels. Figure 4.4 shows the image which is enlarged by 6 times with nearest interpolation method and the enlarged images which are smoothed with a 3×3 modal filter, a 5×5 modal filter and a 7×7 modal filter respectively. From the results, it is obvious that the effect of the modal filter is associated with its size. The modal filter with bigger size gets better result. However, the data at the edges of the image can't be smoothed because lack of enough surroundings. When bigger size is chosen, more edge data can't be smoothed. Thus, considering the advantage and disadvantage of bigger size modal filter, the 7×7 modal filter is chosen.

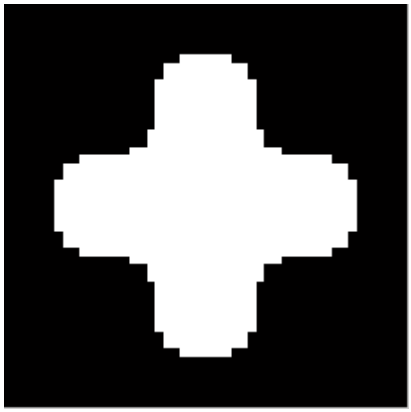
Figure 4.3: The original 2D image



(a) Nearest interpolation

(b) 3×3 modal filter

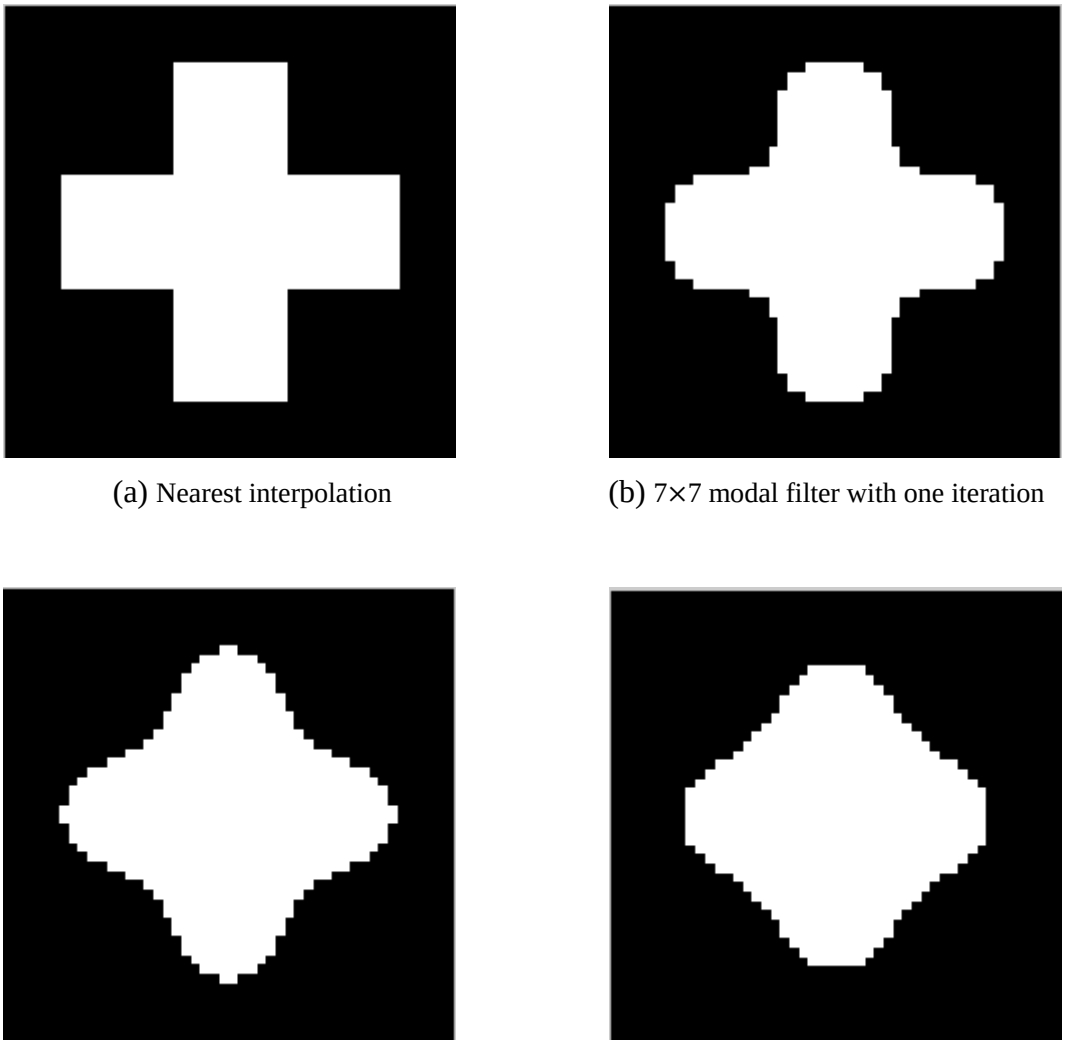
(c) 5×5 modal filter



(d) 7×7 modal filter

Figure 4.4: The effects of different size of modal filter

The first image in Figure 4.5 shows the image which is enlarged by 6 times with the nearest interpolation method. The second image is enlarged by 6 times and smoothed with a 7×7 modal filter for once. The third and the forth images are smoothed with a 7×7 modal filter for three iterations and five iterations respectively. From the results, it can be seen that the repetition of modal filter can improve the effect evidently. The more iterations are applied, the better result is achieved, but iterations cost time. As shown in table 4.1, more repetitions take longer time. Although the time needed for smoothing this simple case is short, when applying to real digital human body images, it needs much longer time. Considering the speed of the program, three times repetition is chosen for this project.



(a) Nearest interpolation

(b) 7×7 modal filter with one iteration

(c) 7×7 modal filter with three iterations

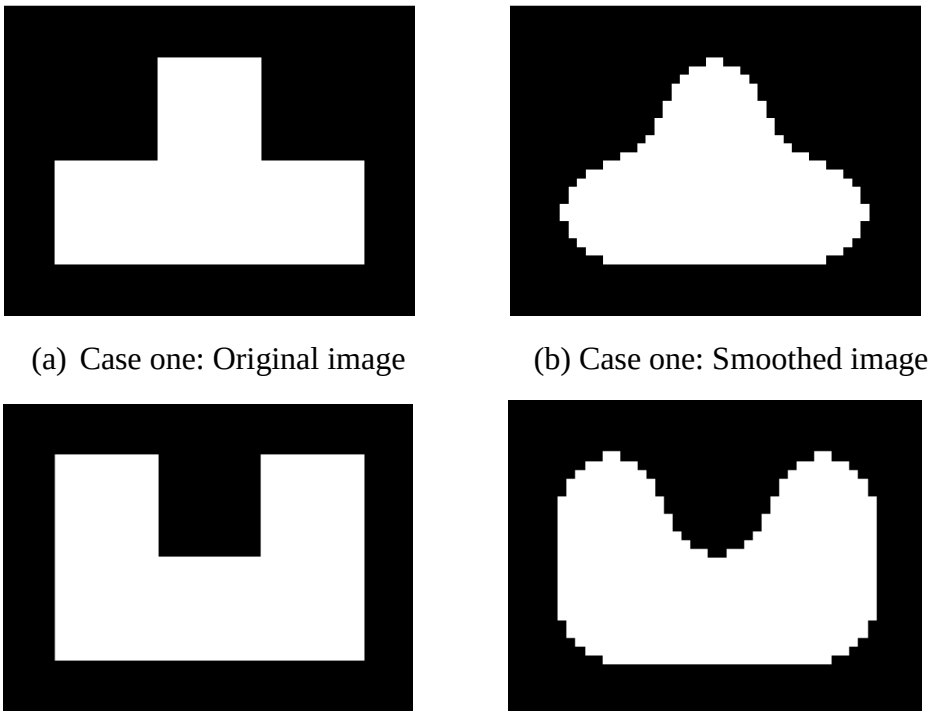
(d) 7×7 modal filter with five iterations

Figure 4.5: The effects of 7×7 modal filter with different amount of iterations

Table 4.1: Running time for different amount of iterations when the 7×7 modal filter is applied to simple 2D cases.

Amount of iterations	One	Three	Five
Time for running the program(seconds)	0.360 s	0.606 s	0.825 s

According to the previous discussion, the 7×7 modal filter and repeating for three times are chosen for these example cases. Figure 4.6 shows the other two simple cases of using nearest interpolation and the chosen method respectively. Both of the two images have 48×36 pixels.



(a) Case one: Original image (b) Case one: Smoothed image

(c) Case two: Original image (d) Case two: Smoothed image

Figure 4.6: Two cases of original images and the images smoothed by 7×7 modal filter with three iterations.

4.2 Modal filter method in 3 dimensions

When the modal filter method is applied to smooth simple cases in 2 dimensions, a plane modal filter is used. But when the modal filter method is used to smooth 3 dimensional cases, the modal filter need to be spatial. Because the modification of each pixel value in a 3 dimensional image depends not only on the layer which it

belongs to, but also on the nearby layers.

For 2 dimensional cases, a 7×7 modal filter is chosen. For 3 dimensional cases, the modal filter should be 7×7×7. For each pixel in 3 dimensional cases, except for the pixels on the edges, the modal filter covers 343 surrounding pixels of it. Therefore, every time there are 343 pixel values to be counted. Since different values represent different tissues, the most common value is the tissue which appears most frequently in this 7×7×7 region. This most common tissue is considered to be the expected output and it is used to replace the pixel of interest.

The key point to smooth a 2 dimensional image is to create circles on a grid of squares. Since for a 2 dimensions image, when it is enlarged by 6 times, each pixel is turned into a 6×6 square with 36 pixels in it. But if a 3 dimensional image is enlarged by 6 times, each pixel in it is turned into a 6×6×6 cube which has 216 pixels in it. Therefore, the key point to make a 3 dimensional image smooth is to create spheres from cubes, which means the pixels at corners of cubes should be replaced by more appropriate pixel values so that those corners can be removed.

The 3 dimensional case in Figure 4.7 has 48× 36×48 pixels. It is built up with 48 layers of 2 dimensional images.

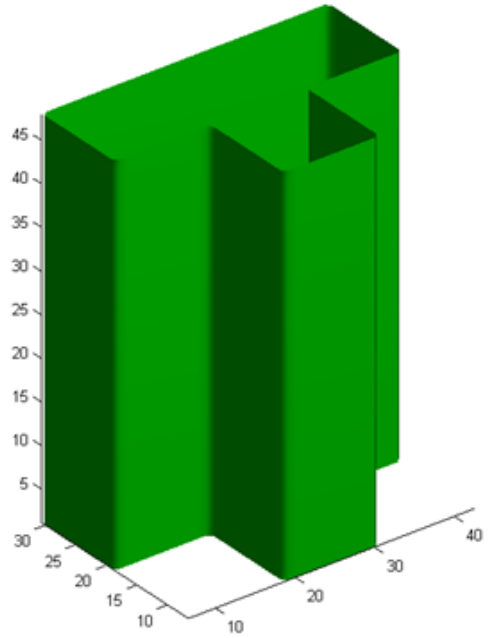


Figure 4.7: The 3 dimensional case which has 48×36×48 pixels.

In order to smooth the 3 dimensions image in Figure 4.7, the values of those pixels at edges and corners should be changed. If the 3 dimensions image is smoothed with a 7×7×7 modal filter directly and is repeated for 3 times. The smoothed result is like the one in Figure 4.8.

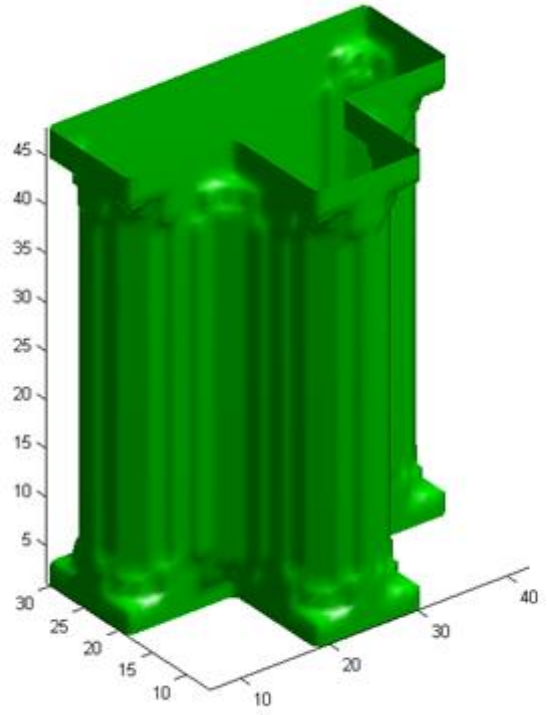


Figure 4.8: The smoothed result of the 3D image above with 7×7×7 modal filter.

According to Figure 4.8, the top part and the bottom part of this image are not smoothed. And the nearby layers are also affected. This problem is caused because the pixels at the top and bottom parts don't have enough surroundings. In this project, since the 7×7×7 modal filter is used, the first three layers at the top and the last three layers at the bottom don't have enough surroundings.

To solve the problem, the first three layers at the top and the last three layers at the bottom can be smoothed using a 2 dimensional modal filter whose size is 7×7. After they have been smoothed, the 3 dimensional modal filter is then used to smooth the 3 dimensional image. The result is shown in Figure 4.9.

This solution can achieve good result with this simple 3 dimensional case because

all the layers contain the same data, which means that there is no difference using 2 dimensional modal filter for each layer or using 3 dimensional modal filter for the whole data. Since all the layers are the same 2 dimensional image, the pixel from one layer doesn't depend on the data from other layers to find the most common value to replace it. But when this method is used in real human body data, the effect is not as good as this one because all the layers in a real 3-dimensions human body image are different from each other. Smoothing of each layer depends on the data from its nearby layers. However, this method still can improve the smoothing result somehow, though it can't solve this problem completely.

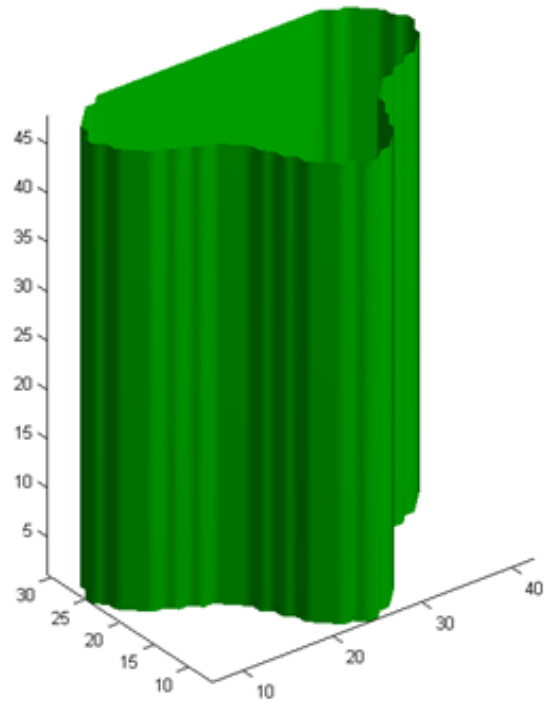


Figure 4.9: The smoothed result of the 3D image with improved modal filter method.

Table 4.2: Contrast of the pixel values in original 3D example image and smoothed 3D example image.

Pixel value	Original image	Smoothed image
0	55296	55418
255	27648	27526

The most important objective of this project is smoothing while bringing no new information to the image. In MATLAB, function ‘imhist(uint8(…))’ can be used to check the pixel values and the quantity of each pixel value in the image. Table 4.2 shows the pixel values in both original image and smoothed image. The original image only has 0 and 255, while the smoothed one also only has 0 and 255. The difference is the quantity of 0 and 255. Though there are 82944 pixels in total for both images, they have different amount of zeros. This difference is achieved through smoothing.

Chapter 5

Application of Modal Filter Method with Human Body Data and Results Analysis

According to chapter 4, the modal filter method achieves good smoothing effect with the simple example cases in both 2 dimensions and 3 dimensions. In all example cases, the jagged edges and sharp corners are reduced obviously. And no new information is brought into the smoothed images.

The modal filter method is appropriate for this project according to the smoothed results of the example cases. However, the example cases only have 2 pixel values, while the real human body images have much more pixel values. Smoothing real human body data might be a little different from smoothing the example cases.

5.1. Application and results analysis in 2D human body image

The 2 dimensions human body data used in this project is a slice of human brain image whose size is 490×265. It contains 15 different pixel values, representing 15 different kinds of human tissues.

When a part of this brain image is enlarged by 6 times using nearest interpolation, there are a lot of jagged edges as shown in figure 2.9 previously. This part of image has been resampled using other methods at the previous chapters. The bilinear and bicubic smoothing results are shown in figure 3.1, while the resampled result using look up table method is shown in figure 3.5. As mentioned previously, the bilinear and bicubic results are blurred with meaningless new information, while the result using look up table method has some pepper noise on it.

Figure 5.1 shows the image smoothed with modal filter method. This image is produced when the nearest interpolation result in figure 2.9 is smoothed by a 7×7 modal filter for 3 times. According to this result, the smoothing effect of modal filter

method is impressive. There is no blurring or jagged edge, less sharp corners and no pepper noise.

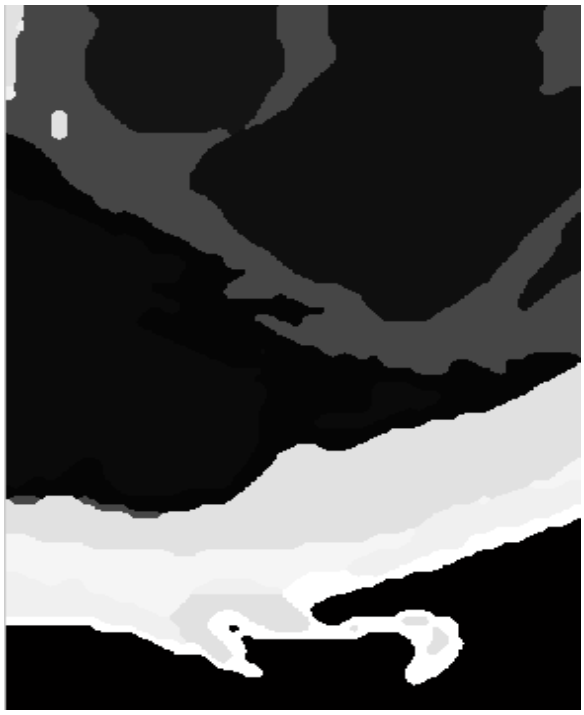


Figure 5.1: The smoothing result of a piece of 2D human brain image with modal filter method for three iterations

The slice of human brain data has 15 different human tissues. Table 5.1 shows the amount of each tissue in the original image, the nearest interpolation image and the image smoothed with modal filter method. There is nothing new brought into the data while smoothing, since the image smoothed also has 15 kinds of tissues. The only difference is the quantity of each tissue.

Table 5.1: Contrast of the pixel values in original 2D human brain image and smoothed 2D human brain image.

Tissue	Original image	Nearest interpolation	Modal filter method
0	104262	3753432	3753565
5	2180	78480	78964
10	4056	146016	145847
15	2550	91800	91712
20	538	19368	19390

25	709	25524	25678
30	84	3024	3035
40	58	2088	2013
45	26	936	882
70	2859	102924	102902
185	1372	49392	50518
225	4353	156708	157969
240	3167	114012	111229
245	2362	85032	84656
255	1274	45864	46240

The modal filter method is very flexible to use. According to different requirements, the size of the modal filter and the time of repeating can be changed. Figure 5.2 and Figure 5.3 are smoothed with a 7×7 modal filter for once and five iterations respectively, and figure 5.1 shows the image smoothed with a 7×7 modal filter for three iterations.

Figure 5.2: The smoothing result of a piece of 2D human brain image with the modal filter method for once.

Figure 5.3: The smoothing result of a piece of 2D human brain image with the modal filter method for five iterations.

From the above three figures, it's obvious that the more times the modal filter is repeated, the better smoothing effect is achieved, and the more details are faded out. From the table 5.2, it also takes a longer time to run the program if the modal filter is repeated for more times.

Comparing these factors and the smoothing effects, the 7×7 modal filter is chosen for this project and it is repeated for three times.

Table 5.2: Running time for different amount of iterations when the 7×7 modal filter is applied to a slice of 2D human brain image.

Amount of iterations	One	Three	Five
Running time(seconds)	11.014	33.805	58.741

5.2. Application and results analysis in 3D human body data

The 3 dimensions human body data used in this project contains 300 layers of 2-dimensions human brain image whose size is 490×265. The size of 3 dimensional data is 490×265×300. It contains 92 different pixel values, representing 92 different kinds of human tissues. The entire 3-dimensions human brain image is as shown in Figure 2.7.

The PC used in this project has 8.00 GB RAM and the operating system is 64-bit windows. Use "memory" command to check the available memory in MATLAB and the following information is given.

```
>> clear
>> memory
Maximum possible array:     13340 MB (1.399e+10 bytes) *
Memory available for all arrays:     13340 MB (1.399e+10 bytes) *
Memory used by MATLAB:      1040 MB (1.091e+09 bytes)
Physical Memory (RAM):      8066 MB (8.458e+09 bytes)

*  Limited by System Memory (physical + swap file) available.
```

Figure 5.4: The memory information of MATLAB with the PC used in this project

If the 3 dimensions human body data is enlarged by 6 times, a matrix with size of 2940×1590×1800 is required. The default numeric type for MATLAB is double, which means each figure in the matrix occupies 8 bytes.

$$2940\times1590\times1800\times8 = 6.7314\times10^{10}$$

This matrix contains 6.7314×10^{10} bytes. It needs much more memory than the available memory. Even if the 3-dimensions human body data is enlarged only by 3 times. The size of the matrix needed should be $1470\times795\times900$, containing 8.4143×10^{9} bytes. This is less than the available memory. However, the program needs two matrixes to run and each of them has 8.4143×10^{9} bytes. Therefore, the memory needed for all arrays is 1.6829×10^{10} bytes. Running the program with these arrays causes "out of memory" errors in MATLAB.

In order to avoid this problem and run the program successfully, the human body data is divided into three parts as shown in Figure 5.5, Figure 5.6 and Figure 5.7. Each part contains 100 layers. Thus the size of each part is 490×265×100. The three parts are enlarged by three times and smoothed with modal filter method respectively.

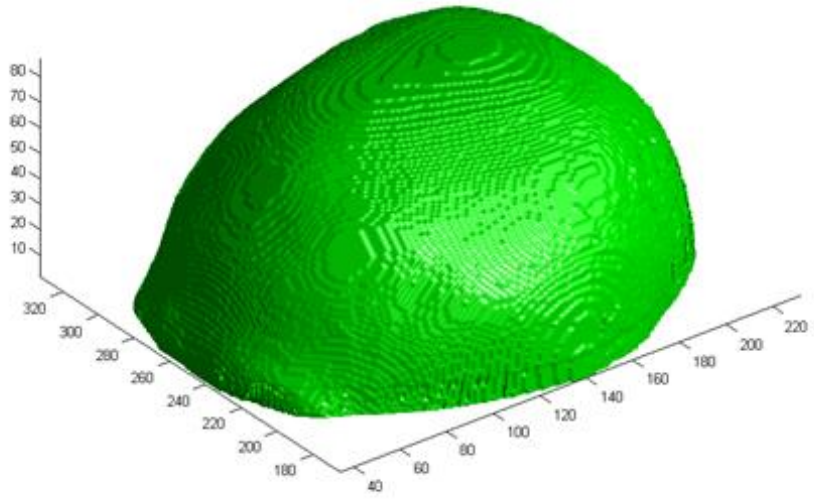


Figure 5.5: First part of the 3D human brain image.

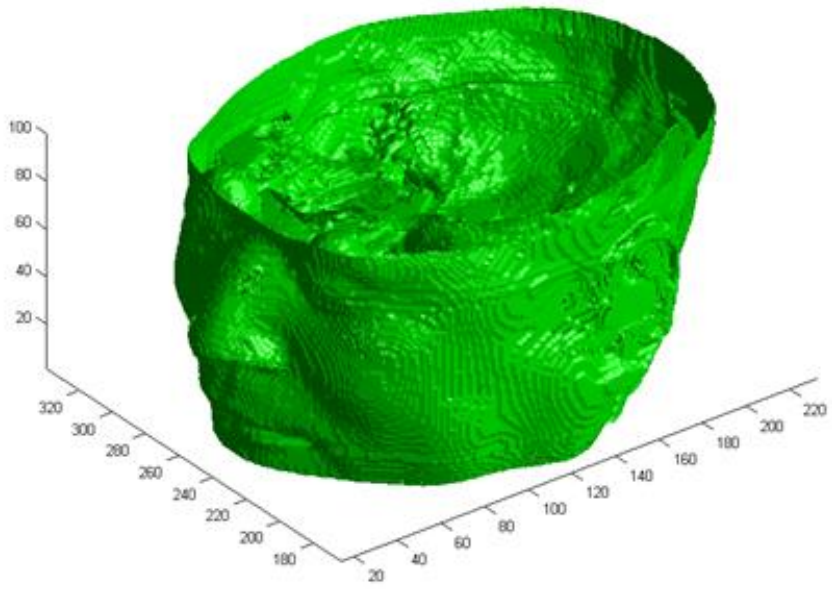


Figure 5.6: Second part of the 3D human brain image.

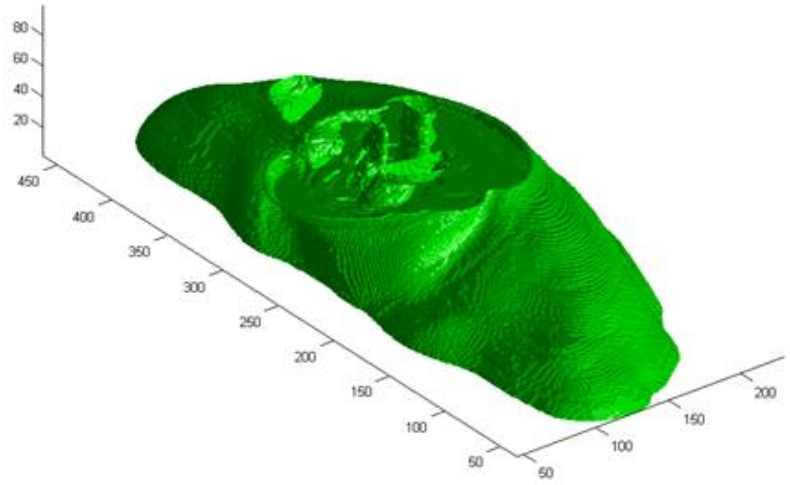
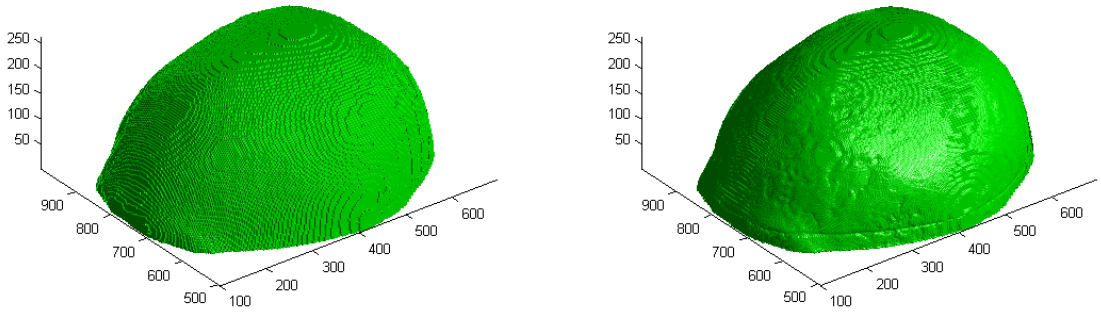


Figure 5.7: Third part of the 3D human brain image.

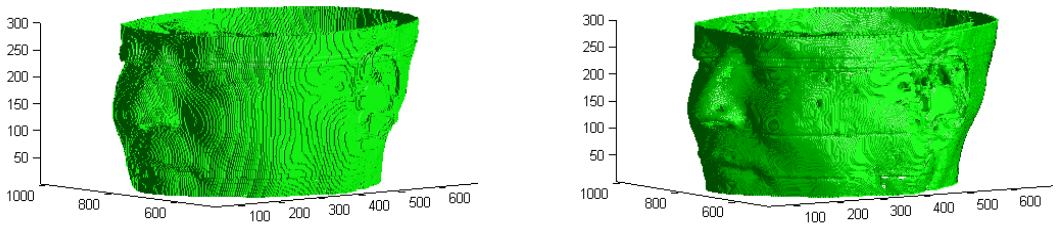
Figure 5.8, Figure 5.9 and Figure 5.10 show the enlarged first part, second part and third part respectively. In all these three figures, the first image is enlarged by three times with nearest interpolation method, while the second image is smoothed by modal filter method.

There are many obvious knurls on the surface of the enlarged 3 dimensions images. To remove these knurls, the key point is to create spheres from cubes. After the enlarged images are smoothed with a 7×7×7 modal filter and repeated for three times, the surfaces of the 3-dimensions images are smooth. There are less jagged edges on the surfaces. This is because pixels at corners of cubes are replaced by the more common pixel values from their surroundings. In this way, there are less sharp corners or sharply changed edges.



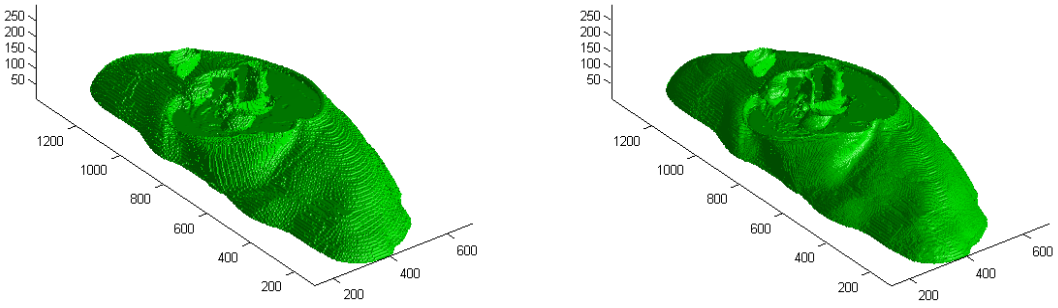
(a) Enlarged result (b) Smoothed result

Figure 5.8: The enlarged first part and the smoothed result of it.



(a) Enlarged result (b) Smoothed result

Figure 5.9: The enlarged second part and the smoothed result of it.



(b) Enlarged result (b) Smoothed result

Figure 5.10: The enlarged third part and the smoothed result of it.

Comparing the results of nearest interpolation method and the results of modal filter method, the modal filter method produces good smoothing effects. But the above three figures mainly show the smoothing effects of surfaces. The smoothing effects of the inside compositions are also very important.

The vertical section chosen in this project is divided from the original 3 dimensions human brain data. The size of the vertical section is 490×135×150. After it is enlarged by three times, the size is turned into 1470×405×450. Figure 5.11, Figure 5.12, Figure 5.13 show the original 3 dimensional vertical human brain section, the enlarged image with nearest interpolation method, the smoothed image with modal filter method respectively. Figure 5.14 and Figure 5.15 show the contrast of the inside smoothing effect.

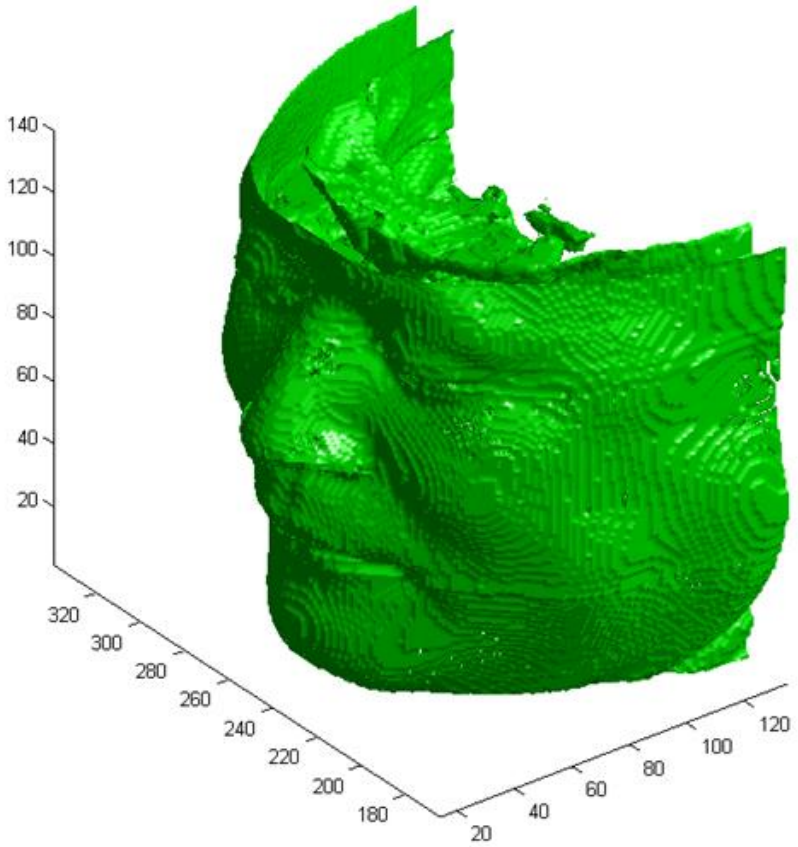


Figure 5.11: The original 3 dimensional vertical human brain section.

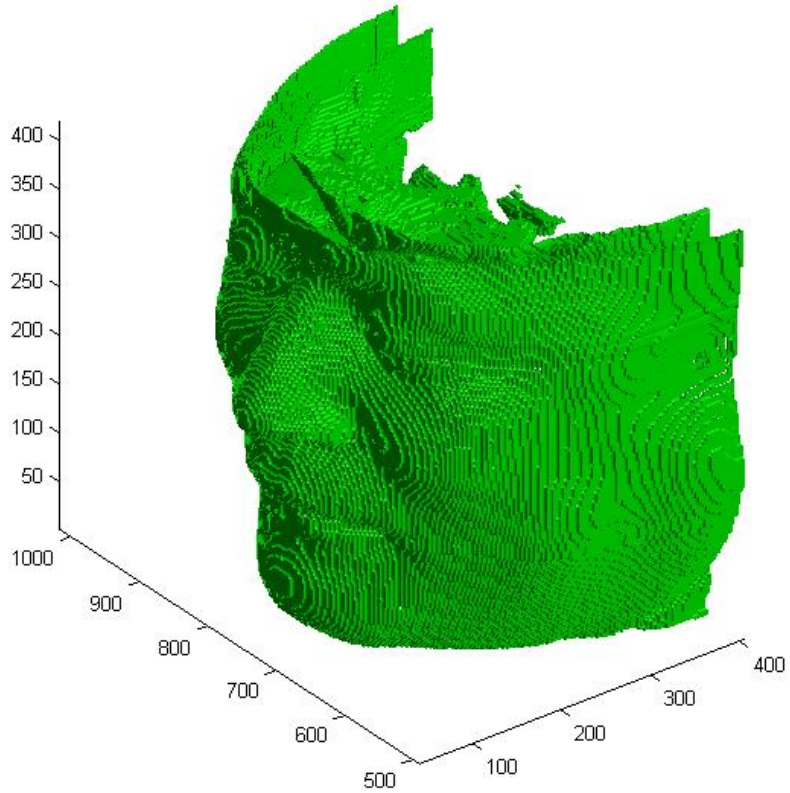


Figure 5.12: The enlarged 3 dimensional vertical section with nearest interpolation method.

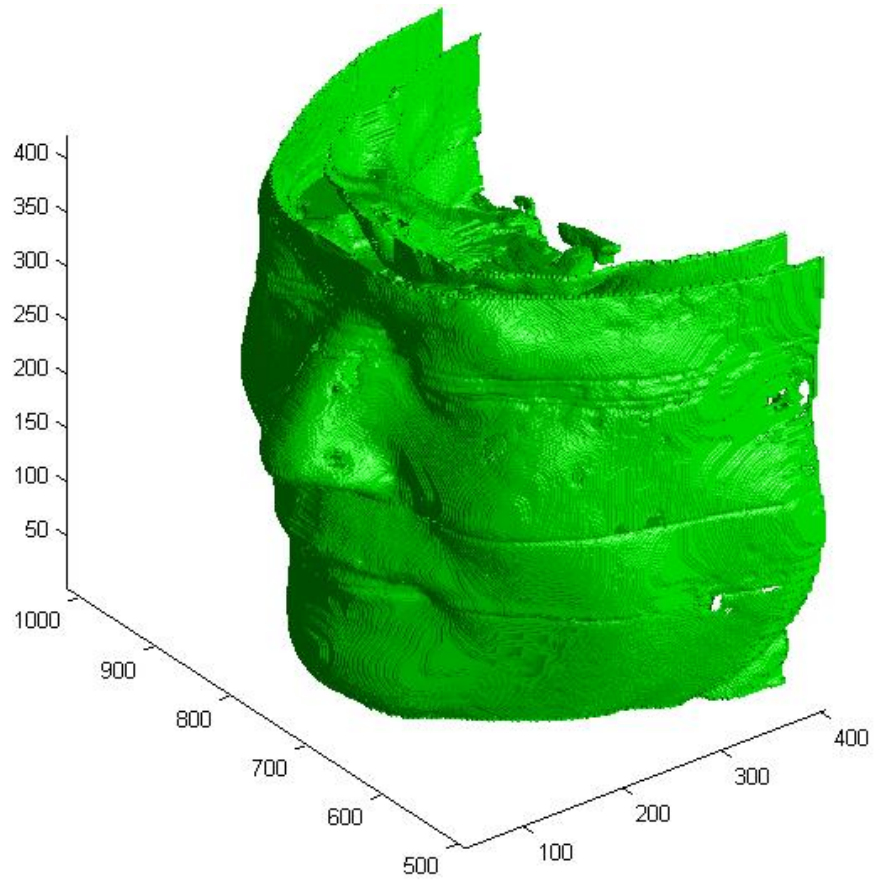


Figure 5.13: The smoothed 3 dimensional vertical section with modal filter method.

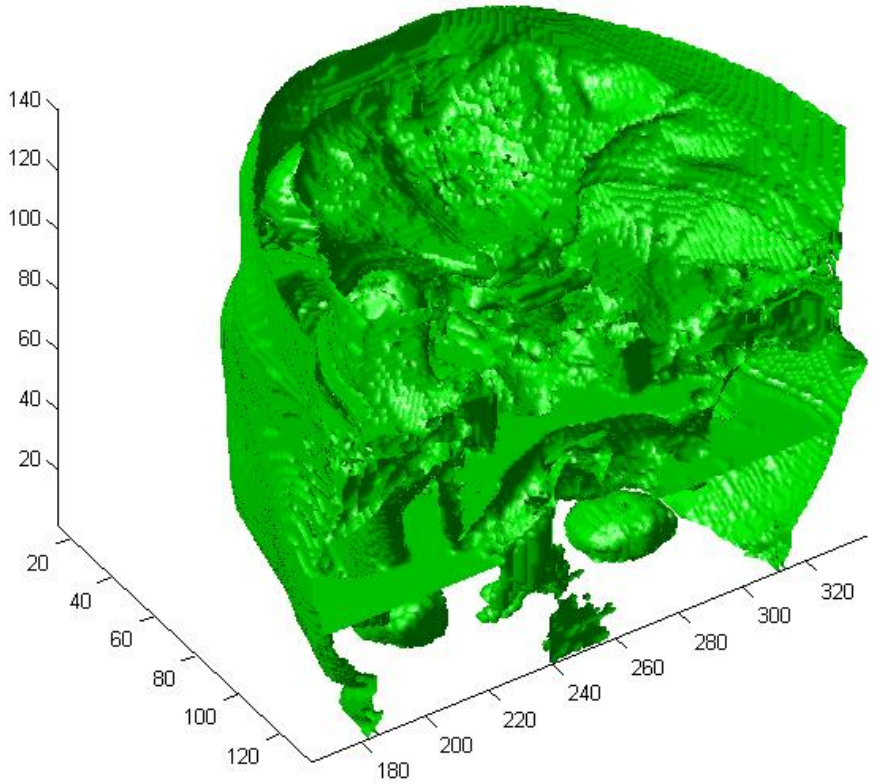


Figure 5.14: The inside composition of the original vertical section.

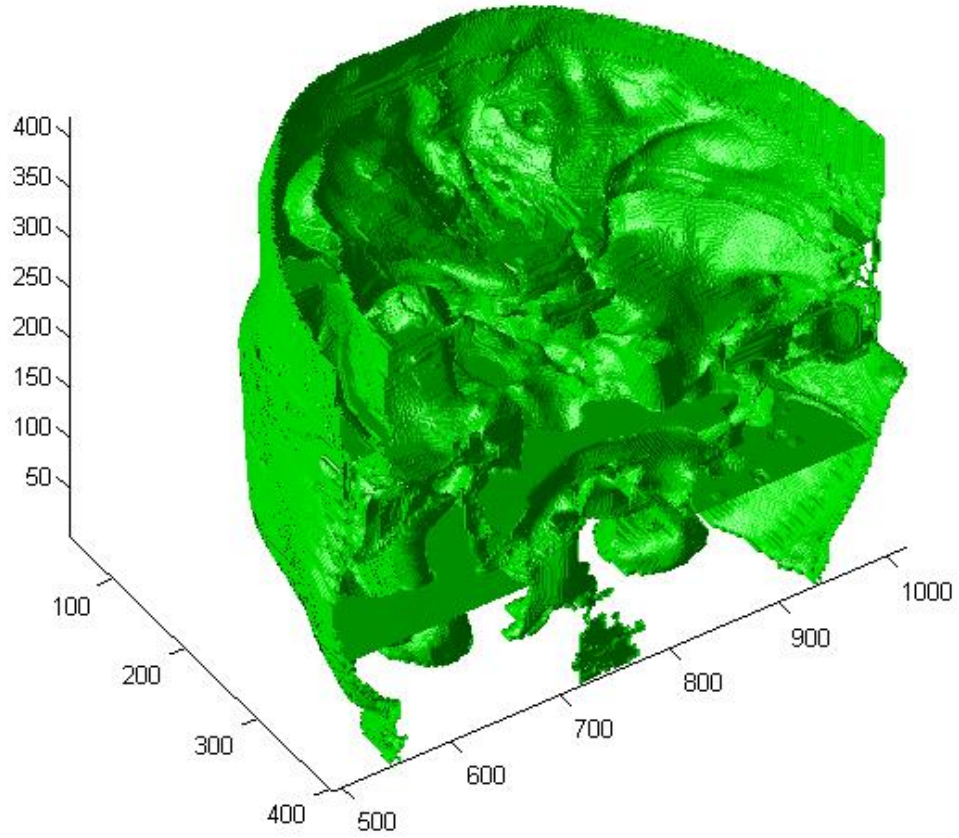


Figure 5.15: The inside composition of the up-scaled and smoothed one.

Comparing the original image with the resampled image, the resampled image is much smoother and clearer. The finer resampling is achieved through modal filter method.

Chapter 6

Speed up with Parallelization in MATLAB

Though the modal filter method achieves the finer sampling in MATLAB and produces good results, it takes a very long time to smooth the 3 dimensions human brain data.

For the three divided parts of human brain data discussed in chapter 5, smoothing the enlarged three parts takes 14869 seconds, 19668 seconds and 16254 seconds, respectively. And this 3 dimensions data is just for a human brain. If the data of a whole human body is needed to be enlarged and smoothed, the program will take a much longer time to run. Therefore, a new approach is necessary to be developed to improve the processing speed and reduce the running time.

Parallel computing is a computation in which many calculations are carried out at the same time [23]. With parallel computing, a computation task is usually decomposed into several subtasks that can be processed independently. After these subtasks are processed, the results of subtasks are combined together to produce the complete result.

Parallel computing can be classified into two types mainly, task parallelism and data parallelism. Task parallelism is achieved with multiprocessors in parallel computing environments. The processors work on different tasks based on the same or different data. Data parallelism is a kind of parallelization achieved with multiprocessors performing the same task on different sections of distributed data.

MATLAB has its own parallel computing toolbox which can be used to solve computationally and data-intensive problems with multicore processors or computer clusters [24].

MATLAB has parallel pools in it. A parallel pool is a set of MATLAB workers in a computer cluster or desktop, bound together by a special type of job so that they can

be used interactively and can communicate with each other during the lifetime of the job [25]. MATLAB with different versions have different numbers of workers available. With one computer, version R2009a can create 8 workers at maximum, while the version R2014a can create 12 workers.

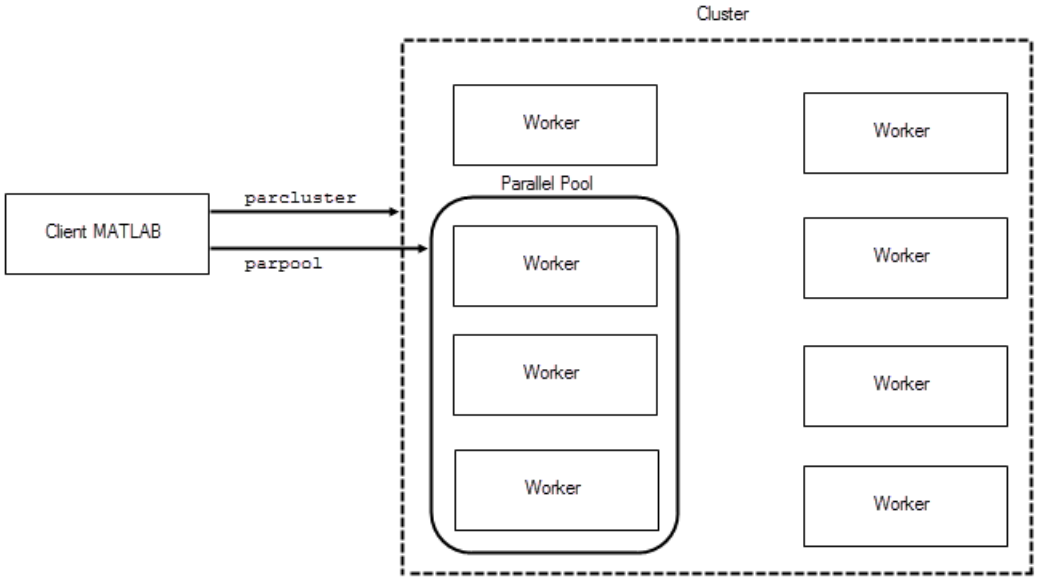


Figure 6.1: The structure of a parallel pool in MATLAB [25]

6.1. Task parallelism in MATLAB

In MATLAB, task parallelism can be achieved by parallel for-loops (parfor) on multiple processors with shared memory architecture [26]. The parallel for-loops allow several MATLAB workers to perform individual loop iterations at the same time.

Some programs contain loops with many iterations or long iterations. When there are a lot of iterations, the program might require a long time to run even if each of the iterations by itself doesn't take a long time.

Typically, the only difference between iterations is the input data. In these cases, running separate sweep iterations simultaneously can improve the speed of the program [24].

If using the parallel for-loops to run a loop of 20 iterations on 4 MATLAB workers

simultaneously, each worker executes 5 iterations of the loop. The time required for running this loop can be reduced to a quarter of the original one. Whether the loop has many iterations or long iterations, the speed can be improved by distributing iterations to a set of MATLAB workers [24].

However, there is a limitation with the task parallelism using parallel for-loops. When parallel for-loops are used, the iterations should be independent, which means no iterations are allowed to depend on any other iterations. But in this project, the iterations are not independent from each other. When the modal filter is used for three times, the input data for the second iteration is the result of the first iteration and the input data of the third iteration is from the second one. Therefore, the task parallelism with parallel for-loops can't be applied for this project.

6.2. Data parallelism in MATLAB

As mentioned in the above section, the parfor-loop can't be used when an iteration in the loop depends on the results of other iterations [24]. Data parallelism doesn't require the iterations to be independent because data parallelism can process the same task on different data.

The workers from the same parallel pool can communicate with each other, so the array can be distributed among workers. The single program multiple data (spmd) construct can be used to define a block of code running in parallel on all the workers. The workers executing a spmd statement operate simultaneously and are aware of each other [24]. They can communicate and transfer data with each other.

In this project, there are three main steps to up-scale and smooth an image with the modal filter method. The flow chart of up-scaling and smoothing with modal filter method is shown by Figure 6.2. Firstly, the 3 dimensions human brain data is enlarged by 3 times with nearest interpolation method. Secondly, the data of the whole image is traversed to locate the edges in the image. Thirdly, the corners are smoothed with modal filter method. For the pixel of a corner, its original surrounding pixels in the modal filter region are counted to determine the most

common pixel, which represents the most common human brain tissue. After the most common pixel is determined, it is used to replace the original pixel.

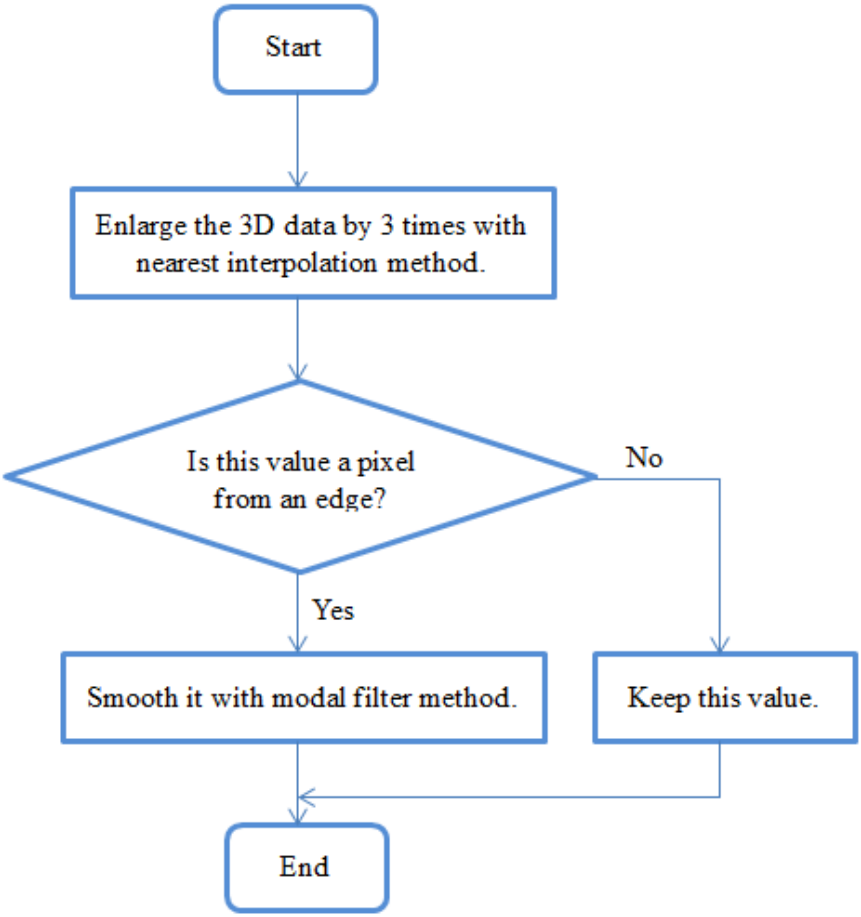


Figure 6.2: The flow chart of up-scaling and smoothing with modal filter method.

These three processing steps are repeated procedures for the whole image data. Therefore, the original input image data can be decomposed into several data blocks which can be processed in parallel. In other words, the integral original image can be decomposed into several different sections and all the sections are processed simultaneously. After they are up-scaled and smoothed, the rescaled sections are coalesced together to obtain the integral rescaled image.

Since the data parallelism distributes the data instead of iterations, it doesn't require the iterations to be independent from each other. Therefore, data parallelism is appropriate for this project.

The 3 dimensional human brain data can be distributed into several different sections easily. After the data is distributed, the set of data is processed by a

computer cluster with distributed shared memory architecture [27] simultaneously. As mentioned in chapter 5, the whole human brain data is distributed into three parts and each of them is processed by a computer from the computer cluster. The results of these sections can be transferred to one computer after they are achieved. All results are combined together to build up the final result. Although the data distribution, transmission and data composition take extra time, the total running time should still be reduced. However, due to time constraints, the parallelization part is never been achieved during this project.

Chapter 7

Conclusions and Future Work

7.1 Conclusion

The 2-dimensions digital human body image used in this project consists of 1 mm^3 voxels. The size of the whole 3 dimensions human brain data is 490×265×300. This human brain data is intended for use in an exploratory study.

The main aim of this project is to achieve higher spatial resolution and finer sampling in MATLAB without bringing any new information. Direct scaling produces results with jagged edges and sharp corners that are not expected to appear in human brain. In order to remove these unnecessary formations and reduce the impacts of sharp corners, smoothing is required. Standard and common smoothing methods, such as bilinear method, bicubic method and convolution filter based methods, can't produce ideal results, since common smoothing methods might bring new information that are not supposed to exist. The look up table method doesn't produce new information in the smoothed image. But there is some pepper noise on the smoothed image.

In this project, the modal filter method is developed to smooth the up-scaled human brain image in both 2D and 3D. The 2-dimensions image is enlarged by 6 times with nearest interpolation method first, then it is smoothed by a 7×7 modal filter with 3 iterations. 3 dimensional human brain data is enlarged only by 3 times and it is divided into 3 parts due to the limited memory capacity of the available resources. The enlarged 3-dimensions image is then smoothed by a 7×7×7 modal filter with 3 iterations. The smoothing effect of the modal filter method is pretty good. The blocky and jagged edges are replaced with much smoother edges. The sharp corners are reduced significantly. Modal filter method produces good results without bringing anything new.

When the 2D image is up-scaled by 6 times, the spatial resolution is improved from 1 mm to 0.167 mm. The 3D image is only up-scaled by 3 times due to the memory limitation. Therefore, the spatial resolution of the up-scaled 3D image is 0.33 mm. After the up-scaled image is smoothed, the image looks smoother and clearer. With these two steps, finer sampling in MATLAB is achieved.

7.2 Future work

The modal filter method is flexible depending on the requirement of the project, since the size of the modal filter and the amount of iterations can be changed easily. Increasing the size of the modal filter or the amount of iterations can produce a better smoothing effect. However, larger size of modal filter or more iterations means longer running time. Thus, considering all these factors, a compromise should be made between the smoothing effect and the running time according to the requirement.

Although the modal filter method is appropriate for this project and is developed to get finer sampling results in MATLAB, it has an obvious drawback. When it is extended into 3 dimensions, the program takes a very long time to run. Chapter 6 discusses about using the parallelization to reduce the running time. Due to time constraints, parallelization is never been implemented in this project. Through the data decomposition and parallel processing, the running time should be reduced. The purpose of the future work is to up-speed the program.

Appendix A

Feasibility Study Report

Table of Content

Abstract

A variety of waves are used in a wide range of technologies. The Finite Difference Time Domain (FDTD) method is the most common method of modelling the propagation of waves within space. There are many concrete applications of the numerical simulation of the electromagnetic wave propagation based on the FDTD method. One of these applications is the computation of the electromagnetic waves in and on the human body for the development of health care devices.

A digital human phantom is required to run a numerical simulation. The spatial resolution of the digital human phantom has to be flexible depending on the simulation requirement. In order to obtain required spatial resolution, resampling of the digital human phantom is necessary. However, the normal resampling of the digital human phantom can exaggerate the effect of the staircase of the original one.

This report aims to introduce the project which focuses on accessing the raw data of the images and developing an algorithm to perform finer resampling. Special attention is paid to reduce the staircase by smoothing.

1. Introduction

As the biological science and modern medicine develop quickly, many new technologies and methods are used in these fields. The Finite Difference Time Domain (FDTD) method is one of them. The FDTD method is a very popular technique for wave modelling. And it is well suited to model complex and irregularly shaped objects such as the human body [1]. Numerical simulation of the electromagnetic wave propagation based on this method can be used to compute the electromagnetic waves in and on the human body. The use of numerical simulation protects human beings from harmful experiments. However, numerical simulation is complex and requires radio environment setting. In order to be able to the simulation, a digital human phantom has to be used for setting the propagation media.

Digital human phantoms are used frequently and widely. There are different kinds of human body phantoms gathered through different methods, such as Magnetic Resonance Imaging (MRI) and Computed Tomography (CT).

Currently the spatial resolution of the digital human phantom is fixed to 1 mm for one model and 2 mm for two other models used in this project. The spatial resolution should be flexible depending on the simulation requirement. When a very precise computation is required, the simulation needs to be operated based on the very fine digital human phantom such as 0.1 mm~0.25 mm spatial resolution to achieve high accuracy [2].

In order to change the spatial resolution, the digital human phantom should be resampled. The normal resampling might exaggerate the effect of the staircase of the original 1 mm resolution digital human phantom.

When an image is up-scaled using a normal method, such as the nearest neighbor interpolation method, there can be a lot of sharp steps and corners on the edges such as shown in figure 1.1. The original image is enlarged 6 times.

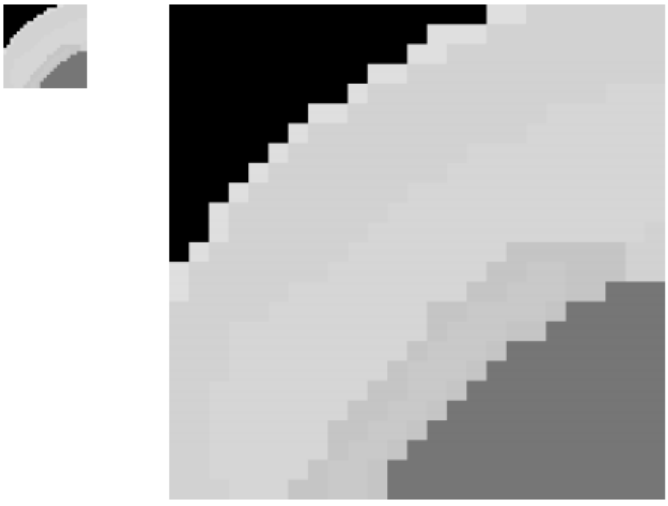


Figure 1.1: A part of image enlarged 6 times [3]

Since these sharp corners are exaggerated during up-scaling procedure, they are unnecessary information and affect the image quality. In order to reduce these sharp steps, smoothing is required.

This project aims to up-scale and smooth the digital human phantom in both two dimensions and three dimensions for producing high spatial resolutions. The main object is to develop a smoothing algorithm that can reduce the sharp corners while bringing no new value into the image data, since every pixel value in the image represents a specific human tissue and new values will make no sense. Therefore, common smoothing methods can not fit this project well because they might bring some new values to the image and cause nonsensical output. The objectives of this project are listed below:

a) Understand the principles of resampling and smoothing.
b) Develop an algorithm for resampling and smoothing.
c) Write codes to realize the algorithm and apply it to a slice of brain or chest data.
d) Optimize the algorithm from the viewpoint of accuracy and CPU time.

This report mainly introduces the aims of the project, the review of this subject and the analysis of the methodology to achieve the objectives. It has several sections, including the introduction, which sets out the aims and objectives of the project in detail. Then the background knowledge and associated research are given in the second part. The third part describes an appropriate methodology to achieve the desired results. The following part concerns the project planning with a chart listing detail activities and time planning. Project risk is also offered in this part. Finally, there is a conclusion at the end of this report.

2. Background Knowledge and Current Research

2.1. PGM Format

A pgm data file consists of numbers of gray between black and white. The maximum value of the data is presented as white, while the minimum value is presented as black [4]. A data interpretation is shown in table 2.1. The result of this data file is shown in figure 2.1. This result is obtained with a software “xv”.

Table 2.1: The data interpretation of PGM format

P2	This means the data is pgm format.
4 2	The x-axis has 4 pixels and y-axis has 2 pixels.
8	The effective maximum value in the data is 8.
1 2 . . . 8	These are the actual data at each pixel.

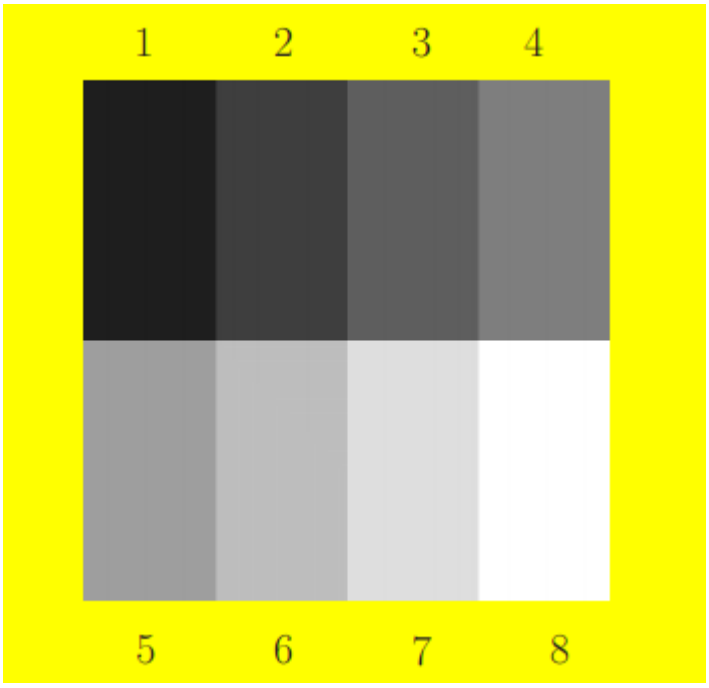
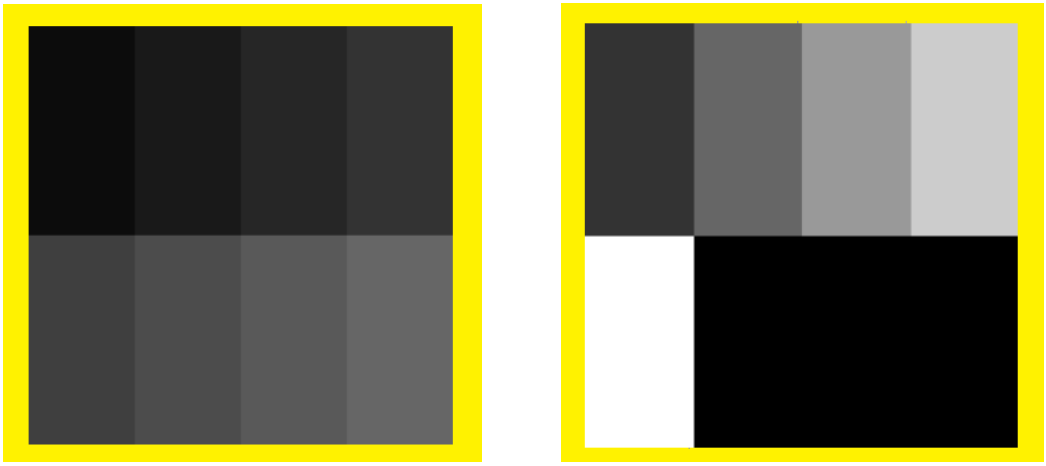


Figure 2.1: The result of the example data file in xv [5]

When the effective value is set to a value higher than the maximum value of the whole data, the entire data is presented as a low value data. For example, if the value is set to be 20, the appearance is like figure 2.2 (a). When the effective value is set to a value lower than the maximum value of the whole data, then the data values above this effective value are regarded as zero. For example, if the value is set to be 5, the appearance is as shown in figure 2.2 (b) [5].



(a) (b)

Figure2.2: The results of two examples with inappropriate effective values [5]

Therefore, the effective value can't be set to a lower value and there is no need to set it to a higher value.

2.2. Digital Human Phantoms

Digital human phantoms are models of the human body used in computational analysis [6]. They are currently being increasingly implemented in the teaching of anatomy [7]. With the use of the digital human phantoms, many kinds of

simulations and experiments can be operated without doing harm to living objects. There are a variety of different kinds of models. Figure 2.3 is a TARO model, which is constructed from Magnetic Resonance Imaging (MRI) scans taken of live volunteers, with voxel sizes of $2 \times 2 \times 2$ mm [8].

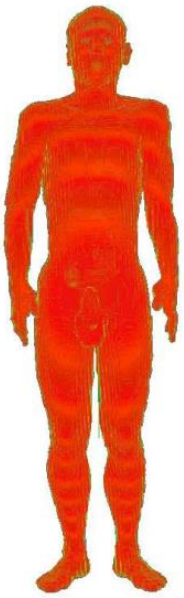


Figure 2.3: A rendering of the TARO voxel phantom produced with the ParaView visualization software [3]

When a human body is scanned using Magnetic Resonance Imaging (MRI) method, each scan shows the cross section of the human body orthogonal to the direction of the backbone. The MRI scanned image is segmented, and the segmentation can be used to identify a tissue. In this image, each pixel has a tissue number. Different numbers represent different tissues. For example, 3 is the number used to represent cornea, 38 is the representation of blood and 42 is for fat. Therefore, the scanned image can be replaced with a stream of integers. This stream of integers is stored in a file named with the height of the MRI scan phantom from the ground [9].

These files can be converted into pgm format by adding some appropriate line headers. Figure 2.4 is an example of a .pgm image converted from a voxel phantom.

Figure 2.4: Z-normal plane 220 from the voxel phantom TARO converted to a pgm image [3]

2.3. Up-scaling and Smoothing

In this project, the scanned digital human phantom is generated by scanning a human body every 1mm or 2mm. The size of the pixel is 1mm × 1mm or 2 mm × 2mm. However, in some applications, the spatial resolution is required to be 0.1~0.3 mm. High spatial resolution are important to detect some details in an image, such as the edges of structures and margins of tumors [10]. In order to change the spatial resolution, the digital human phantom has to be up-scaled and smoothed.

2.3.1. Up-scaling

A variety of fast and powerful algorithms are developed to operate on image data. For example, various filters are used to reduce noise on images, a lot of scaling techniques are necessary to change the image sizes while retaining as much information as possible or bringing as little noise as possible. Many scaling techniques are developed focusing on decreasing image sizes. However, this project targets at increasing the size of the image while providing no new information in the up-scaled image.

The simple principle of up scaling an image is like the simple example shown in figure 2.5. The original image has (2×2) pixels. Since it is increased by 3 times, the rescaled image has (6×6) pixels. That means every pixel in the original image is replaced by 9 pixels which have the same value as the original pixel [11].

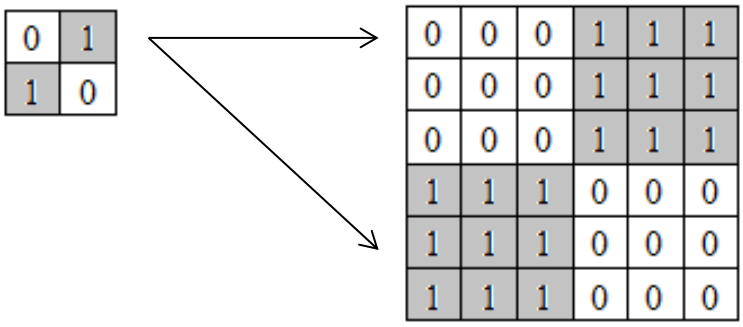


Figure 2.5: The original image and the image which is up scaled by 3 times

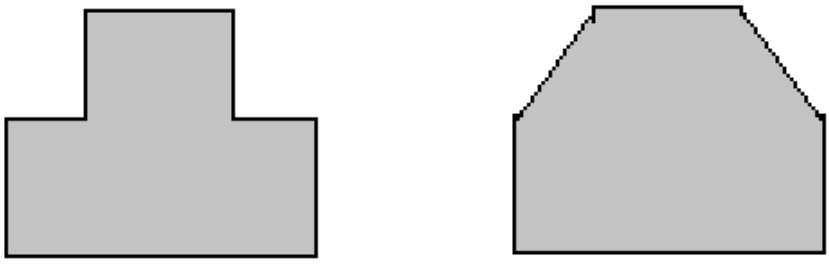
However, up-scaling can bring some sharp corners and exaggerate the effect of the staircase of the original image, hence produces a very jagged image such as the example in the previous figure 1.1. The original image is rescaled to increase the resolution of a small section of a TARO z-plane from 2 mm to 0.33 mm [3].

2.3.2. Smoothing

The problems caused by up-scaling can be reduced through smoothing techniques. Smoothing is a very important image processing method in which some data points are modified to reduce individual points and smooth the corners and sharp steps on the image. The aim of smoothing is to obtain slowly changing values and replace those sharp corners with smooth curves [12].

Imagine extracting one single sharp corner and smoothing it, as the example shown

in figure 2.6. The original one has an edge line which changes sharply. After smoothing, the slowly changing edge replaces the original one.



(a).Original (b).After smoothing

Figure 2.6: Simple example of smoothing a single sharp corner

In order to achieve the expected smoothing effect in figure 2.6, the values of some pixels in the original image are required to be replaced by more appropriate values. The changes of pixel values are shown in figure 2.7.

0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1

0	0	0	1	1	1	0	0	0
0	0	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1

(a).Original values (b).Values after smoothing

Figure 2.7: Changes of the pixel values to achieve smoothing result

As previously discussed, in the MRI scanned image, each pixel has a tissue number and different numbers represent different tissues. Thus, all the values in the image have their own meanings and new values are meaningless. That is the reason why normal smoothing methods are inappropriate in this project.

2.4. Current Smoothing Methods

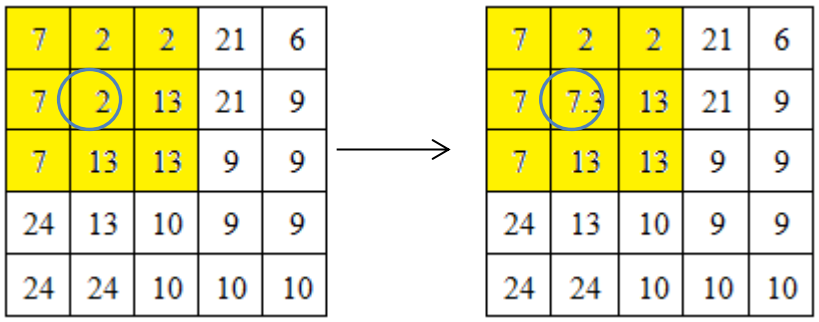
There are some common smoothing methods, but generalized smoothing risks the possibility of removing necessary sharp edges or providing new information in the rescaled image. However, there are some common methods that can be adapted to provide more satisfactory results [3].

2.4.1. Convolution Filter Based

Convolution filter operates on a value and its surrounding elements to obtain a new value and uses the new value to replace the original one. There are a lot of different convolution filters and the mean filter is the simplest one. It calculates the average result of the values in a certain region and replaces the center pixel of this region with the calculated result [11].

Figure 2.8 shows an example of the operation with a 3×3 mean filter. The yellow region is the 3×3 mean filter model. The value of the center pixel in this region is

changed from 2 to 7.3 after filtering. The value 7.3 is obtained by averaging the 9 values in the original yellow region.



Original After filtering

Figure 2.8: The example of applying a 3×3 mean filter to change a pixel value

From the example, it's obvious that the value created by using the mean filter is a new value which doesn't exist in the original image. Since the aim of the project is to up-scale and smooth the image while producing no new information. The new value doesn't represent any tissue and has no meaning. Thus, there is a high possibility of producing nonsensical output by using mean filter. This is the problem of applying the mean filter or any other weighted averaging filter [11]. Although the median filter doesn't produce new values, it is also likely to produce nonsensical output [13]. Therefore, common convolution filter based methods are inappropriate for this project.

2.4.2. Look-up Table Based

Look-up table method is a kind of smoothing method employed successfully in the field of computer games. The principle of this method is different from the convolution filter based method. The convolution filter based method scales the image first and then considers the contents of the local region for a large amount of new pixels [3].

The look-up table method uses the arrangement of similar pixels around a certain pixel in the original image to determine the layout of this region in the up-scaled image. Different schemes for the look-up table can lead to different kinds of layout of the final pixels [3]. In the hqnx family of scalers created by Maxim Stepin, the look-up table based filter compare the 8 surrounding pixels with the center pixel to define each as being either near or distant based on how similar the colors are [14].

However, the nearness of the surrounding pixel values to the center pixel is irrelevant in building the phantom. A value that is one away is as different as a value 20 away, which means the connection of the values is not dependent on their nearness. Therefore, the only pixels considered as similar are those with exactly the same value. There are 8 pixels surrounding the center, each of them may have the same value with the center pixel or not. This makes 256 (2^8) potential combinations of similar pixel arrangement. When an image is up-scaled and smoothed with the look-up table method, the shape of the output region is defined according to the original pixel layout. Then the original pixel data is used to combine with this shape

to produce the final output region. The layout of the final pixels is dependent on the original data and the scheme chosen for the look up table. Using different schemes can produce different output [3].

Figure 2.9 shows an example. The original data is used to select an output shape firstly. This shape is then combined with the original data to produce the layout of pixels in the output region. In this example, pixels that are not assigned to the original value can be instead assigned the value of the more common pixel value in this region [3].

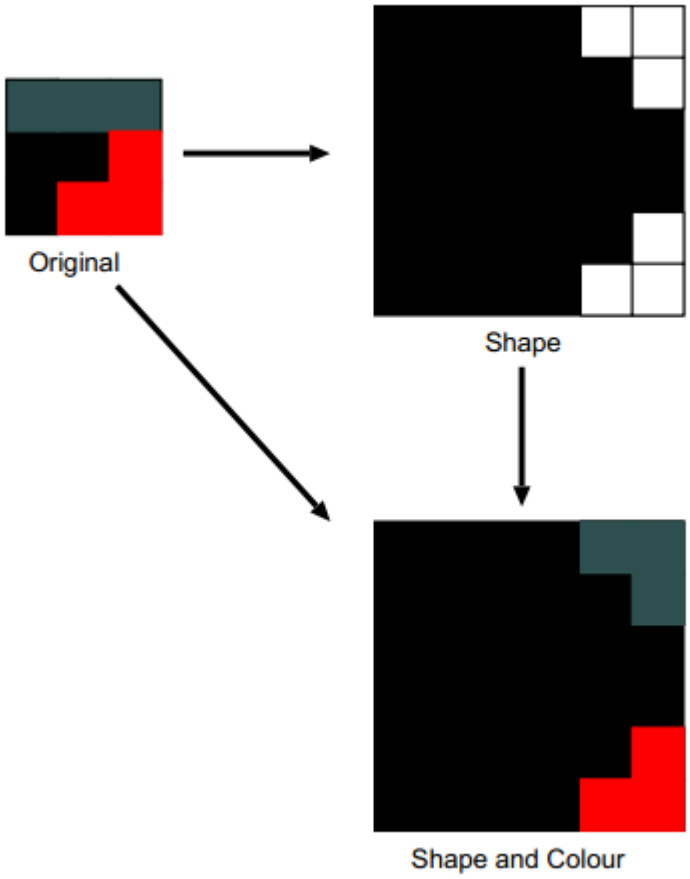


Figure 2.9: An example of the look-up table process [3]

3. Methodology

This project aims to obtain a finer resampling with MATLAB. The data from an image can be represented through a matrix. And MATLAB is a kind of software which is particularly efficient in processing matrix operations [15].

In order to achieve a finer resampling, an appropriate algorithm should be developed. Once the algorithm is determined and developed, the code based on this algorithm will be written and tested in MATLAB. The code will be applied to a small image to see the effect of this algorithm. If the result is good enough, this algorithm will be used to up-scale and smooth the 2D images first and then used for 3D images. On the other hand, if the result is not good enough, the algorithm will be optimized and tested again and again till the expected result is obtained.

This part introduces two approaches simply. One is modal filter which is adapted from the convolution filter method and the other is the data decomposition and parallel processing approach for the look-up table method.

3.1. Modal Filter Method

As mentioned previously, convolution filter based methods are inappropriate for this project because common convolution filters are likely to bring new values into the rescaled image and produce nonsensical output. However, the common method can be adapted to provide more satisfactory results.

The key point is to restrict the output to a value that is known to exist in the model and is appropriate for the region which is required to be smoothed. To achieve the expected output, a kind of neighborhood mode is applied. The main idea of this neighborhood mode is to count the existing pixel values in the model region and find the most common one to replace the pixel of interest.

This method is like the convolution filter based method, which scales the image first and then considers the contents of the local region for a large amount of new pixels [3].

For example, if a 5×5 modal filter is used to smooth a section of an image, every time there are 25 pixel values to be counted. Since different values represent different tissues, the most common value is the tissue which appears most frequently in this 5×5 region. This most common tissue is considered to be the expected output and it is used to replace the pixel of interest.

Although the modal filter method produces reasonable results, the preservation of small structure size results in lesser reduction of sharp corners [3]. This is a problem to be solved when applying and optimizing this algorithm.

3.2. Data Parallelism Processing Approach

The previous approach for look-up table method is based on Michael Knight's work. In that work, the digital human phantom is scaled by 6 times and smoothed with the look-up table method [16]. The result was good, but the program took a very long time to run. Therefore, a new approach is necessary to be developed to improve the processing speed and reduce the running time. Data parallelism distributes the data across different parallel computing nodes. It is achieved by performing the same task on different pieces of distributed data [17].

In the previous work, there are three main steps to up-scale and smooth an image. Firstly, the data of the whole image is traversed to locate the corners in the image. Secondly, operate on the corners. For the pixel of a corner, check its original 8 surrounding pixels to determine the pattern and combination from 256 potential combinations. Thirdly, after the combination is determined, according to the scheme chosen for the look-up table method, there is a corresponding layout for the output region.

These three processing steps are repeated procedures for the whole image data. Therefore, the original input image data can be decomposed into several data blocks which can be processed in parallel. In other words, the integral original image can be decomposed into several different sections and all the sections are processed synchronously. After they are up-scaled and smoothed, the rescaled sections are coalesced together to obtain the integral rescaled image. Through the data decomposition and parallel processing, the running time can be reduced.

4. Project Plan and Risks

4.1. Project Planning

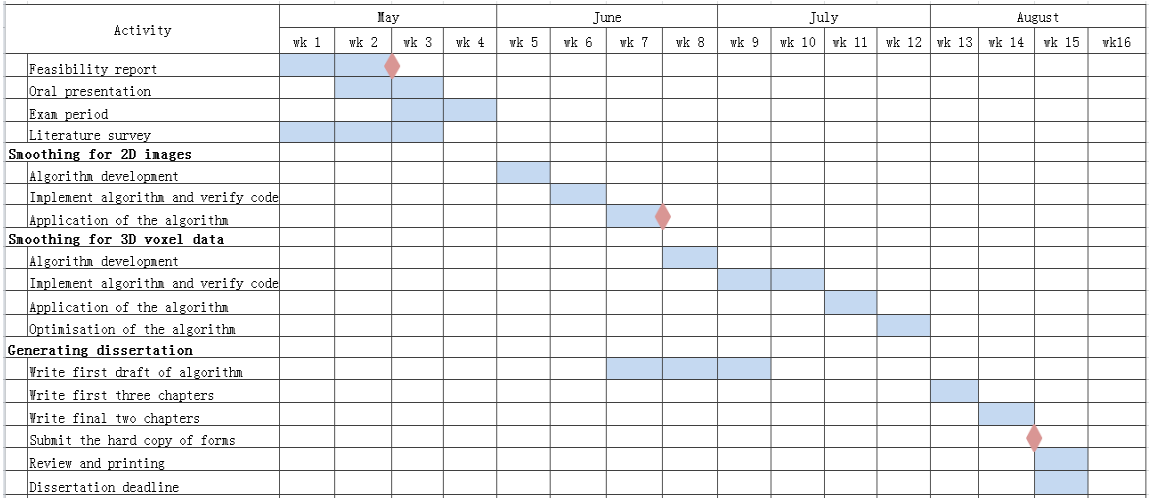


Figure 4.1: Gantt chart for this project

The algorithm is developed for 2D images at first. After 2D images are scaled and smoothed successfully, algorithm for 3D images is developed. The project planning is shown in the chart in figure 4.1. The detail planning from June 3rd is shown in table 4.1.

Table 4.1: Detail project planning from June 3rd

Date	Planning
03/June~09/June	Algorithm development for the smoothing for 2D images.
10/June~16/June	Implementation of the algorithm and verification of the code with the test case.
17/June~23/June	Application of the smoothing algorithm to a slice of brain or chest data and check whether or not all the smoothing is performed satisfactorily.
24/June~30/June	Algorithm development for the smoothing for 3D voxel data.
01/July~14/July	Implementation of the algorithm and verification of the code with the test case.
15/July~20/July	Application of the smoothing algorithm to the whole brain or the whole chest data and check whether or not all the smoothing is performed satisfactorily.
21/July~27/July	Optimization of the algorithm from the viewpoint of the accuracy, CPU time, memory requirement.

After the plans in table 4.1 are completed, the dissertation is started in August and is supposed to be finished before August 20th.

4.2. Project Risks

There are some potential issues that might prevent the successful completion of this project. The first issue is the time constraints. Due to time limitation, the algorithm development for 3D voxel data might not be fully completed. The detail time planning for this project in table 4.1 is the mitigation strategy for this issue. From table 4.1, there are several days that are unoccupied at the end of July. If the project couldn't be finished as the time planning, these days can be used as extension.

The second issue is about the algorithm mentioned previously in the methodology part. There is a possibility that the algorithm can't achieve the expected output result. For the modal filter, the effect of reducing sharp corners may not be good enough. For the data parallelism processing approach, the pixels at the edge of each decomposed section have a high possibility to get unsatisfactory smoothing output because those pixels are lack of necessary surrounding information. These problems will be considered when developing and optimizing the algorithm. If it turns out that the problems can be solved or the algorithm is inappropriate for this project, a new alternative algorithm will be chosen and developed.

The third issue is about the hardware and devices. During writing the code or the dissertation, there is a possibility that the devices, such as the computer in use, might break down and lead to important files losing. To avoid this situation, it is important to back up the code, the dissertation and other files.

5. Conclusion

This report introduces the aims and objectives of this project. It discusses the preliminary background studies and the detail planning of the project. The approach to the project is also described simply. Considering the aims, the existing work, the approaches and the time planning, this project is feasible to be carried out. Through there might be some potential issues or problems, there are mitigation strategies to reduce the influence of the issues.

References

[1] H.K. Rouf, F. Costen, and M. Fujii. Modelling em wave interactions with human body in frequency dependent crank Nicolson method. *J. of Electromagn*. Waves Appl., 25:2429-2441,2011.

[2] M. Potse, A.-R. LeBlanc, and A. Vinet. Why do we need supercomputers to understand the electrocardiographic t-wave? *Anatol. J. Cardiol., Suppl*. 1:123–4, 2007.

[3] M. Knight. *A digital human phantom for use in the FDTD computation of body surface potential during cardiac arrhythmia*. Master's thesis, The University of Manchester, School of Computer Science, 2008.

[4] W. Burger and M. J. Burge. Principles of Digital Image Processing, *Fundamental Techniques*. Springer, first edition, 2009.

[5] LectureEECEMLab. Chapter 3: Static spatial domain signal presentation for publication.http://personalpages.manchester.ac.uk/staff/fumie.costen/MScProject/lectureEECEMLab.pdf

[6] Xu, X.G. Eckerman, K.F. Handbook of anatomical models for radiation dosimetry. *Taylor & Francis*, 2010. ISBN 978-1-4200-5979-3.

[7] The National Library of Medicine. The national library of medicine's visual human project. http://www.nlm.nih.gov/research/visible/visible human.html. Accessed on 4 May 2015.

[8] Tomoaki Nagaoka, Soichi Watanabe, Kiyoko Sakurai, Etsuo Kunieda, Satoshi Watanabe, Masao Taki, and Yukio Yamanaka. Development of re-alistic high-resolution whole-body voxel models of Japanese adult males and females of average height and weight, and application of models to radio-frequency electromagnetic-field dosimetry. *Physics in Medicine and Biology*, 49:1–15, 2004.

[9] F. Costen and A. Balasko. Opportunities and challenges in porting a parallel code from a tightly-coupled systems to the distributed eu grid, enabling grids for e-science. *Engineering Science Reference*, 1:197–217, 2011.

[10] N. Joseph and T. Rose. Quality assurance and the helical (spiral) scanner. http://www.ceessentials.net/article33.html. Accessed on 6 May 2015.

[11] Rafael C.Gonzalez, Richard E.Woods, Steven L.Eddins. *Digital Image Processing Using MATLAB*. Third edition, 2010.

[12] Simonoff, Jeffrey S. *Smoothing Methods in Statistics*, the second edition,1998.

[13] Wang Xiaokai, Li Feng. Improved adaptive median filtering. *Computer Engineering and Applications*, 2010, 46 (3): 175-176.

[14] Maxim Stepin. hq3x magnification filter.

[15] Daniel So. Wireless Communication & Mobile Network. Lecture 5: Introduction to MATLAB. The University of Manchester, School of Electrical and Electronic Engineering.

[16] O. Sahin. *Digital human phantom for use in the FDTD computation.* Master's thesis, The University of Manchester, School of Electrical and Electronic Engineering. 2012.

[17] Hillis, W. Daniel and Steele, Guy L., *Data Parallel Algorithms Communications of the ACM*. December 1986.

Appendix B

Programs for Scaling and Smoothing

B.1 Median filter method for 2D image

This program operates on the raw data of a slice of 2D human brain image, which is currently located on E:\COMMS\lecture\project.

```
% Read information into array and create new arrays for up-scaled
% and smoothed image data. 5×5 median filter is used. Counter () has a length
% of 25 to save the 25 pixel values in a 5×5 median filter region.

clear
I = imread('E:\COMMS\lecture\project\test120.pgm');
[a,b] = size(I);
Large = zeros(6*a,6*b);
Result = zeros(6*a,6*b);
counter = zeros(25);

% Up-scale the original image with nearest interpolation method.

for I = 1:a*6
      for j = 1:b*6
            Large(i,j) = I(ceil(i/6),ceil(j/6));
      end
end

% Display the up-scaled image.

figure(2);subplot(1,2,1);imshow(uint8(Large));

% Find the pixels at edges and smoothed them with the median filter method.

for I = 3:a*6-3
     for j = 3:b*6-3

      % Operate only on regions which are away from the image boundaries and near
      % corners and edges.

          if(Large (i,j-2) = = Large (i,j) & & Large (i,j+2) = = Large (i,j)) ||
            (Large (i-2,j) = = Large (i,j) & & Large (i+2,j) = = Large (i,j))
```

```
            Result (i,j) = Large (i,j);
            else
                  k = 1;
                  for zonei = -2 : 2
                        for zonej = -2 : 2
                              counter(k) = Large (i+zonei,j+zonej);
                              k = k+1;
                        end
                  end

            % sort () is a Matlab function which can be used to put an array in order.
            % counter (13) is the median value chosen to replace the original pixel value.

                  counter = sort(counter);
                  Result(i,j) = counter(13);
            end
      end
end

% Display the smoothed image.

subplot(1,2,2);imshow(uint8(Result));
```

B.2 Modal filter method for 2D image

This program operates on the raw data of a slice of 2D human brain image, which is currently located on E:\COMMS\lecture\project.

```
% Read information into array and create new arrays for up-scaled
% and smoothed image data. 7×7 modal filter is used.

clear
I = imread('E:\COMMS\lecture\project\test120.pgm');
[a,b] = size (I);
Large = zeros (6*a,6*b);
Result = zeros (6*a,6*b);
Result_1 = zeros (6*a,6*b);
[num,tissue] = imhist (I);
temp = find (num~=0);
[x,y] = size (temp);

% counter () is an array with two columns and as many rows as there are tissues
% in the plane.

counter = zeros (x,2);
for k = 1 : x
      counter (k,1) = temp (k) - 1;
end

% Scale the image by 6 times in each direction and display it.

for I = 1:a*6
      for j = 1:b*6
            Large (i,j) = I (ceil(i/6),ceil(j/6));
      end
end
figure(2);subplot(1,2,1);imshow(uint8(Large));

% The part below is the smoothing loops. The modal filter used in this program is a
% 7×7 modal filter with 3 iterations.

for time = 1:3
      Result = Large;

      % Operate only on regions which are away from the image boundaries and near
      % corners and edges.

      for I = 4:a*6-4
```

```
            for j = 4:b*6-4
                counter(1:x,2) = 0;
                if (Large(i,j-3) = = Large(i,j) & & Large(i,j+3) = = Large(i,j)) ||
               (Large(i-3,j) = = Large(i,j) & & Large(i+3,j) = = Large(i,j))

        % If not near a corner, copy output from the scaled data.

                    Result(i,j) = Large(i,j);
                else

        % Count the tissues in the 7×7 modal filter region surrounding the pixel of
        % interest.

                    for zonei = -3:3
                        for zonej = -3:3
                            for k = 1:x
                                if counter(k,1) = = Large(i+zonei,j+zonej)
                                    counter(k,2) = counter(k,2)+1;
                                end
                            end
                        end
                    end

         % Find the most common tissue of this region and use it to replace original
         % pixel value.

                    [m,t] = find(counter(1:x,2) = = max(counter(1:x,2)));
                    Result(i,j) = counter(m(1),1);
                end
            end
        end

        % The input of each iteration is the result from its previous iteration.

        Large = Result;
end

% Display the result.

subplot(1,2,2);imshow(uint8(Result));
```

B.3 Modal filter method for 3D human brain data

This program operates on the raw data of an entire 3D human brain data, which is currently located on C:\ tmpf.

```
% Read 300 layers and save the all tissues in the array named ‘total’.
% 7×7×7 modal filter is used.

clear
images = [];
total = zeros (256,1);
for i = 1:299
      str = strcat ('C:\tmpf\v1_00', int2str(i) ,'.pgm') ;
      img = imread (str);
      images{300-i} = img;
      [num,tissue] = imhist (images{300-i});
      total = total + num;
end

% counter () is an array with two columns and as many rows as there are tissues
% in the 3D human brain data.

temp = find (total ~ = 0);
[x,h] = size (temp);
counter = zeros (x,2);
for k = 1:x
      counter (k,1) = temp (k)-1;
end

% Read the 3D human brain data into a cell array ‘D’.

for i = 1:299
      D = cat (3, D, image{i});
end

% Create two new arrays for up-scaled and smoothed arrays.

[a,b,c] = size (D);
L = zeros (3*a, 3*b, 3*c);
R = L;

% Scale the image by 3 times in each direction.

for i = 1:a*3
      for j = 1:b*3
```

```
            for k =1:3*c
                L (i, j, k) = D ( ceil(i/3), ceil(j/3), ceil(k/3) );
            end
        end
end

% The part below is the smoothing loops. The modal filter used in this program is a
% 7×7×7 modal filter with 3 iterations.

for time = 1:3
    R = L;

    % Operate only on regions which are away from the image boundaries and near
    % corners and edges.

    for i = 4:a*3 - 4
        for j = 4:b*3 - 4
            for h = 4:c*3 - 4
                counter (1:x, 2) = 0;
                if(L (i, j-3, h) = = L (i, j, h) & & L (i, j+3, h) = = L(i, j, h) ) ||
                  (L (i-3, j, h) = = L (i, j, h) & & L (i+3, j, h) = = L(i, j, h) )

                % If not near a corner, copy output from the scaled data

                R(i, j, h)=L(i, j, h);
                else

        % Count the tissues in the 7×7×7 modal filter region surrounding the pixel of
        % interest.

                    for zonei = - 3:3
                        for zonej = - 3:3
                            for zoneh = - 3:3
                                for k = 1:x
                                if counter (k,1) = = L(i+zonei,j+zonej,h+zoneh)
                                   counter (k,2) = counter (k,2)+1;
                                end
                                end
                            end
                        end
                    end

        % Find the most common tissue of this region and use it to replace original
        % pixel value.

                    [m,t] = find (counter (1:x, 2) = = max ( counter (1:x,2) ) );
```

```
                        R(i, j, h) = counter(m(1), 1);
                    end
                end
            end
        end

        % The input of each iteration is the result from its previous iteration.

        L = R;
end

% Display the original 3D human brain image and the enlarged and smoothed result.

[x,y,z,D] = reducevolume (D,[1 1 1]);
fv = isosurface(x,y,z,D,'verbose');
figure(1);subplot(1,2,1);
p = patch (fv,'facecolor','green','edgecolor','none');
colormap (gray);
view (3);
camlight;
lighting gouraud;
axis tight;
daspect ([1,1,1]);


[x,y,z,R] = reducevolume (R,[1 1 1]);
fv = isosurface(x,y,z,R,'verbose');
figure(1);subplot(1,2,2);
p1 = patch(fv,'facecolor','green','edgecolor','none');
colormap (gray);
view(3);
camlight;
lighting gouraud;
axis tight;
daspect ([1,1,1]);
```

References

[1] H.K. Rouf, F. Costen, and M. Fujii. Modelling em wave interactions with human body in frequency dependent crank Nicolson method. *J. of Electromagn. Waves Appl.*, 25:2429-2441,2011.

[2] O. Sahin. *Digital human phantom for use in the FDTD computation.* Master's thesis, The University of Manchester, School of Electrical and Electronic Engineering. 2012.

[3] Y Fang. *Meshing your body-finer sampling in MATLAB.* Feasibility study report, The University of Manchester. School of Electrical and Electronic Engineering. 2015.

[4] M. Potse, A.-R. LeBlanc, and A. Vinet. Why do we need supercomputers to understand the electrocardiographic t-wave? *Anatol. J. Cardiol., Suppl*. 1:123–4, 2007.

[5] Daniel So. Wireless Communication & Mobile Network. Lecture 5: Introduction to MATLAB. The University of Manchester, School of Electrical and Electronic Engineering.

[6] Xu, X.G. Eckerman, K.F. Handbook of anatomical models for radiation dosimetry. *Taylor & Francis*, 2010. ISBN 978-1-4200-5979-3.

[7] The National Library of Medicine. The national library of medicine's visual human project. http://www.nlm.nih.gov/research/visible/visible human.html. Accessed on 4 May 2015.

[8] Tomoaki Nagaoka, Soichi Watanabe, Kiyoko Sakurai, Etsuo Kunieda, Satoshi Watanabe, Masao Taki, and Yukio Yamanaka. Development of re-alistic high-resolution whole-body voxel models of Japanese adult males and females of average height and weight, and application of models to radio-frequency electromagnetic-field dosimetry. *Physics in Medicine and Biology*, 49:1–15, 2004.

[9] M. Knight. *A digital human phantom for use in the FDTD computation of body surface potential during cardiac arrhythmia*. Master's thesis, The University of Manchester, School of Computer Science, 2008.

[10] F. Costen and A. Balasko. Opportunities and challenges in porting a parallel code from a tightly-coupled systems to the distributed eu grid, enabling grids for e-science. *Engineering Science Reference*, 1:197–217, 2011.

[11] W. Burger and M. J. Burge. Principles of Digital Image Processing, *Fundamental Techniques.* Springer, first edition, 2009.

[12] LectureEECEMLab. Chapter 3: Static spatial domain signal presentation for publication.http://personalpages.manchester.ac.uk/staff/fumie.costen/MScProject/lectureEECEMLab.pdf. Accessed on 8 August 2015.

[13] 3-D graphics. http://www.webopedia.com/TERM/3/3_D_graphics.html. Accessed on 13 August 2015.

[14] N. Joseph and T. Rose. Quality assurance and the helical (spiral) scanner. http://www.ceessentials.net/article33.html. Accessed on 6 May 2015.

[15] Rafael C.Gonzalez, Richard E.Woods, Steven L.Eddins. *Digital Image Processing Using MATLAB.* Third edition, 2010.

[16] Simonoff, Jeffrey S. *Smoothing Methods in Statistics*, the second edition,1998.
[17] Jonathan Sachs. Image Resampling. *Digital Light & Color*, 2001.
[18] Jonathan W. Chipman. *E-Study Guide for: Remote Sensing and Image Interpretation*. The Sixth Edition. 2014, eISBN 9781467276696.
[19] Definition of: bilinear interpolation. http://www.pcmag.com/encyclopedia/term/38607/bilinear-interpolation. Accessed on 13 August 2015.
[20] Fabian Khateb, Tomasz Kulej, Montree Kumngern. 0.5-V DTMOS media filter. AEU – International Journal of Electronics and Communications, 2015.
[21] R.C. Gonzalez, R.E. Woods. *Digital Image Processing*. Addison Wesley, New York, 1992.
[22] Maxim Stepin. hq3x magnification filter. http://www.hiend3d.com/hq3x.html. Accessed on 8 August 2015.
[23] Gottlieb, Allan; Almasi, George S. *Highly parallel computing*. Redwood City, Calif. 1989, ISBN 0-8053-0177-1.
[24] The MathWorks, Inc. The R2015a Parallel Computing Toolbox™ User's Guide http://uk.mathworks.com/help/pdf_doc/distcomp/distcomp.pdf. Accessed on 19 August 2015.
[25] R2015a Documentation: Parallel Pools. http://uk.mathworks.com/help/distcomp/parallel-pools.html. Accessed on 19 August 2015.
[26] Hesham, Rewini. *Advanced computer architecture and parallel processing.*2005, ISBN 978-0-471-46740-3.
[27] Patterson, David A. and John L. Hennessy. *Computer architecture : a quantitative approach*. The fourth edition, 2007.Morgan Kaufmann Publishers. 201. ISBN 0-12-370490-1.