• U C •
FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Ricardo Miguel Machado Ribeiro Guerra

Segmentação 3D de imagens médicas volumétricas

Dissertação apresentada à Universidade de Coimbra para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Biomédica

Orientadores:
Henrique Manuel Venâncio Mendes (SEEMSREAL)
Prof. Dr. Luís Alberto da Silva Cruz (Departamento de Engenharia Eletrotécnica e de Computadores Faculdade de Ciências e Tecnologia Universidade de Coimbra)

Coimbra, 2017

Este trabalho foi desenvolvido em colaboração com:

SEEMSREAL – Leiria, Portugal

CODI – Leiria, Portugal

Table of contents

Abstract

Medical imaging is the process of creating visual representations of the internal structure of the (human) body, including important structural elements such as bones and soft organs, as well as external surfaces like the epidermis. Medical images are used more and more in modern medicine, as they can be used in the diagnosis and treatment of a variety of diseases by aiding the clinical practice in several ways like the identification and measurement of tissues and organs volumes and linear dimensions. Key to these tasks is medical imaging segmentation, an important and complex problem that has been studied in recent years. The goal of the segmentation is to identify and separate the different structures present in the 2D or 3D images in order to facilitate the processes mentioned above, making them faster, more efficient and reliable. With the segmentation of the tissues, it is also possible to study the same tissues using a variety of computer programs including creating 3D models of the structures to help in its visualization.

The aim of this work is to study, optimize and implement state of the art 2D and 3D medical segmentation algorithms and develop a prototype that can segment bone structures present in CT image volumes. Two such algorithms were studied in this work, first applied to 2D image segmentation and then extended to 3D volumes. The chosen method was implemented as an application that includes a GUI to help the interaction of the user with the segmentation algorithm.

Resumo

Imagem médica é o processo de criar representações visuais da estrutura interna do corpo humano, incluindo elementos estruturais importantes, tais como ossos, órgãos moles, assim como superfícies como a epiderme. Imagens médicas são usadas cada vez mais na medicina moderna, pois podem ser usadas no diagnostico e tratamento de uma variedade de doenças ajudando na prática clinica em diversas formas como a identificação e medição de dimensões lineares e volumétricas de tecidos e órgãos. Um aspeto chave destas tarefas é a segmentação de imagens médicas, um importante e complexo problema que tem sido estudado nos últimos anos. O objetivo da segmentação é identificar e separar as diferentes estruturas presentes nas imagens 2D ou 3D de forma a facilitar os processos referidos em cima, tornando-os mais rápido, mais eficientes e mais confiáveis. Com a segmentação de tecidos, também é possível estudar os mesmos tecidos usando uma variedade de programas de computador, incluindo a criação de modelos 3D das estruturas para ajudar na visualização das mesmas.

O objetivo deste trabalho é o estudo, otimização e implementação de algoritmos de segmentação médica 2D e 3D estado de arte e desenvolver um protótipo que consiga segmentar as estruturas ósseas presentes em imagens CT volumétricas. Dois algoritmos deste género foram estudados neste trabalho, primeiro aplicados em segmentação de imagens 2D e depois estendidos para volumes 3D. O método escolhido foi implementado como uma aplicação que inclui uma interface gráfica do utilizador (GUI) para ajudar a interação do utilizador com o algoritmo de segmentação.

List of Figures

List of Tables

List of Abbreviations

FC	Fuzzy Connectedness
RFC	Relative Fuzzy Connectedness
IRFC	Iterative Relative Fuzzy Connectedness
FOE	Fuzzy Object Extraction
OSXBF	Osirix Bebrix Foie
OSXBV	Osirix Bebrix Veineux

Chapter 1. Introduction

1.1 Problem Contextualization and Motivation

Medical image segmentation is a complex task that has been studied for many years [1]. The complexity of this process starts with the amount of existing different medical imaging modalities, such as x-ray computer tomography (CT), magnetic resonance imaging (MRI), positron emission tomography (PET), single photon emission tomography(SPET). Each one of the images obtained with these techniques contain a lot of useful information [2], even more so with the improvement of technology. Not only the size and resolution increase, but also the dimensions, resulting in a huge amount of image information [3]. This increase in information is beneficial for medical purposes, however, each different imaging technique needs to be analysed with different segmentation techniques [3].

Segmentation is process that partitions an image in an area, or volume, into regions that have similar properties or share certain characteristics [4]. In many cases this means dividing the image in two classes or regions: the object and the background. The Figure 1.1 displays a simple example of image segmentation. On image (a), there is the real image and image (b), the result of the segmentation of the leaf. Medical image segmentation, has been studied for a few years due to its complexity [5]. This task, separating objects from one another, seems to be quite natural for us humans, however, it is not practical to manually separate all the structures in all the images taken for medical use, as it would take a large amount of time [3]. As such, there has been an effort to develop automatic algorithms to segment the images and volumes requiring little to no intervention from the part of the user. However, these algorithms are still far from producing good results because of the variation of the structures in the human body and also because complex nature of these structures [4] and characteristics of the images, such as their noisiness. As such, every situation is normally studied by itself, either a

particular type of imaging or a specific part of the human body, because a general purpose segmentation algorithm is something that does not seem achievable [3].

(a) (b)

Figure 1.1 Image segmentation. (a) Image (b) segmented leaf [6].

Medical image segmentation is of considerable importance, as it uses images obtained using non-invasive processes that provide a considerable amount of information about the human body, its structure and the presence of certain diseases and pathologies [7]. This information can be used in many medical fields to help diagnose and plan the necessary treatments, such as surgery and radiation based therapy [4].

This work will address the sub-problem of bone and tissue CT image segmentation to identify bone regions. A CT image is obtained using a CT imaging system and it utilizes an x-ray source and detectors, as it is shown in Figure 1.2. The table moves continuously as the source and detectors rotate. This information is the processed by a computer that creates a series of slices, that are two dimensional transversal images of the human body. In Figure 1.3 it is displayed one of those slices from the test volume, that will be used in this work. These slices when put together form a 3D image volume of the patient inner structure [8].

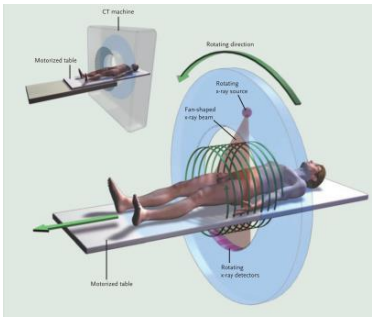


Figure 1.2 Schema of a CT scanner [8].

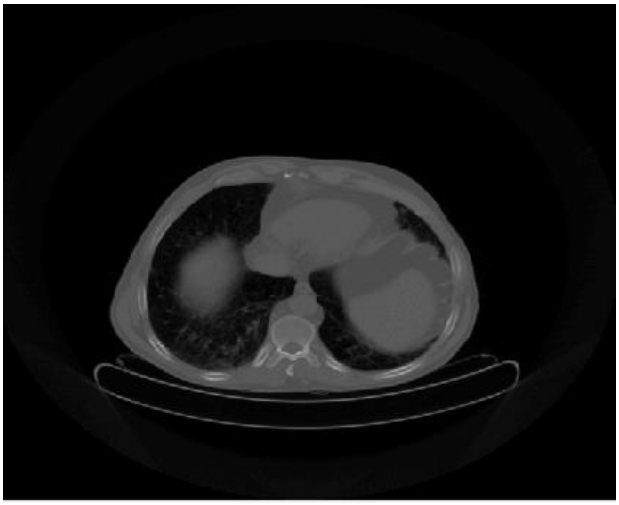


Figure 1.3 CT image slice.

CT images are not perfect, as they have a number of artefacts caused by patient motion, beam hardening, partial volume effect, among others [9]. Partial-volume effects occur when two or more different tissues contribute to a single pixel, or voxel, resulting in a blur of the intensity of the tissues in a single image element [10]. A pixel is the smallest element in a two dimensional image, where a voxel is the smallest element in a three dimensional image. Figure 1.4 shows a representation of the partial-volume effects. It is possible to observe that in the image (b) of that figure, it is difficult to determine accurately the boundaries of the two objects. When combined with the complex structure of the tissue, these artefacts can cause the boundaries between regions to mix and blur [9]. Another problem is the bad representation of walls in CT scans, resulting in objects that are not completely surrounded by their boundaries. This may appear to not be a problem at first, but it causes significant issues in some segmentation techniques [11].

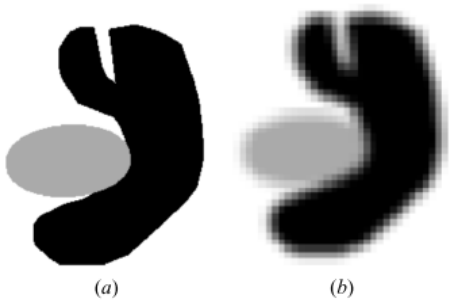


Figure 1.4 Representation of partial-volume effect. Image (a) ideal scenario (b) Real image [10] .

The nature of CT images makes the segmentation process more difficult, not only because of the artifacts mentioned above, but also because of the difficulty of separating the bones of interest from the other bones [1]. It is also needed to keep in mind the big intensity variations of the voxels of the bone in CT images, aggravates the difficulty of extracting the bone out of the surrounding tissue [9]. To deal with the difficulties mentioned above, many authors decided to choose a specific part of the human body to study and try to develop a process that can correctly segment that area. For example the authors of [1] and [5] studied head bones, and discovered that the skull bone has higher intensity and appears to be smooth, while spongy bone has lower intensity and is textured [5].

The analyses of the structures of the bone is very important for diagnostic and accurate treatment planning of osteological diseases [9]. Ideally, there would be algorithms could detect diseases, lesions and other pathologies on their own, all starting with the segmentation of the structures and identifying the problems automatically. However, designing such programs and algorithms, is not so simple, due to need of constant and reliable results every time the program is used, otherwise it can compromise the whole treatment of the patient [3]. As such, developing the algorithms that can give this sort of results requires thorough scientific and medical validation, as well as repeated tests in different environments to make sure they work properly [3].

The goal of this project is to present an algorithm and functioning prototype, that allows the user to select a bone, using a graphic user interface (GUI), displayed in a 3D x-ray computer tomography (CT) image volume. The algorithm will then attempt to segment that bone from the rest of the image, including the other bones present in the volume. In this work, for the test and development of the prototype, it will be used full body CT images, as such, using this algorithm in other forms of CT images may not present the segmented bone, as intended.

1.2 Document Overview

This thesis is structured in seven chapters plus one appendix.

Chapter 2: State-of-the-Art summarizes previous research and work performed on medical image segmentation techniques, that use CT images.

Chapter 3: Fuzzy Connectedness describes in detail the fuzzy connectedness algorithm used in this work, as well as proposed improvements. Both are described step-by-step. Also presented are the several segmentation results using the volumes described in Appendix A.

Chapter 4: Iterative Relative Fuzzy Connectedness describes in detail the iterative relative fuzzy connectedness algorithm used in this work, as well as proposed improvements. Both are described step-by-step. The chapter also presents segmentation results using the modified Iterative Relative Fuzzy Connectedness segmentation method on the volumes described in Appendix A.

Chapter 5: Scale-based Fuzzy Connectedness describes in detail the scale-based modification to the fuzzy algorithm as well as the results obtained using the same volumes as in the previous chapters.

Chapter 6: Prototype describes and presents in detail a prototype application developed for this work as well as an example of how the algorithm works.

Chapter 7: Conclusion evaluates the results of the work,

Appendix A: Materials Used describes the test dataset used in this work.

Chapter 2. State-of-the-Art

As it was mentioned in the previous chapter, medical image segmentation is a complex task that has been studied for many years. There has been a big effort to study possible strategies that can be used to be able to be retrieve and analyse the information contained in these images. This section briefly presents and explains some state-of-the-art segmentation techniques applied on CT images coupled with a brief summary on how they work.

These algorithms described utilize a variety of features from the images such as: appearance features, such as edges; intensity; texture; shape features of the objects contained in the images; combined shape. The first method presented is based on threshold, the following three are region based methods and the last section presents some fuzzy connectedness based strategies.

2.1 Segmentation of Bone in CT Images Using Thresholding Techniques

Thresholding techniques construct a binary image attribute each element in the image, may it be a pixel or a voxel, with value one or zero, creating a binary image. The element is attributed value one, if it is has an intensity value that is higher than the threshold, which is a constant. In case the element has an intensity lower than the threshold, it is given value zero. Thresholding methods normally use the same threshold on the whole image, with the intent of separating the objects of interest, as the main focus of the process, from the rest of the image. This results in a fast method, normally used in CT imaging for separating bone, as it has higher density than the rest of the tissue standing out even to the naked eye [12]. However, the segmentation is not perfect, because there are some pixels in the images that contain both bone and tissue elements in them, resulting in areas that are harder to separate correctly using a single value for the threshold [12].

J. Zhang *et al.* [13], developed a new strategy that possesses all the advantages of thresholding, being fast and efficient, but then post-processes the results to make sure a correctly classified final image is obtained. The method operation will be described for the case of two dimensions first as it is easier to explain. First, they utilized the thresholding technique described above in the image under study with a relative low threshold value that is sure to contain all of the bone in it. This results in a binary image, with pixels separated into two groups: bone and non-bone (background). However, the segmented image has some artefacts: parts that do not correspond to bone that are being represented as bone and holes appearing in the middle of the bones, as it is shown in image (a) figure Figure 2.1. This happens because of the low value of the threshold utilized. It was low enough to make sure all of the bone is segmented, even considering the pixels that have a mixture of bone and tissue, as mentioned before. The low threshold, can cause some denser parts of other tissues to be considered as bone, or part bone incorrectly. However, it is not low enough to cover the bone-marrow, which is softer and has a much lower value in the image than the bone.

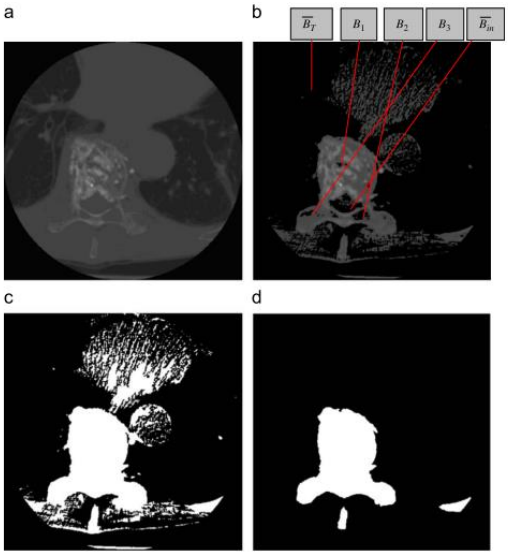


Figure 2.1 (a) original image. (b) Pixels with values lower than the threshold removed. (c) Pixels with value higher than the threshold given value 1 and the flooding algorithm has been run. (d) Final result [13].

The initial results appear to be a bad segmentation at first glance, as it can be seen image (b) from Figure 2.1, but it was necessary to make sure all the pixels that can contain bone are grouped together. The authors [13]**t**hen applied a flood-filling algorithm that starts from one point in the background. This algorithm searches for all the pixels, that are connected to the initial point, makes sure that they are part of the background and joins them in a growing region. In case it reaches solid bone it cannot go any further and will expand around it. Since the bone-marrow is surrounded by solid bone, it is never considered to be part of the background, resulting in bones that do not contain holes, as it is shown in image (c) from Figure 2.1. Considering that the bone is solid, there is no need to try to evaluate all of the elements in the region labelled as bone, to verify if they represent just bone or also represent other tissue. It is only necessary to check if the boundaries are bone or not, as these pixels are the ones that may represent a mixture of tissues. By only evaluating the pixels on the boundaries, it is also prevented the use of another flood-filling algorithm to close out the bone-marrow and it is prevented unnecessary calculations of some of the inner pixels.

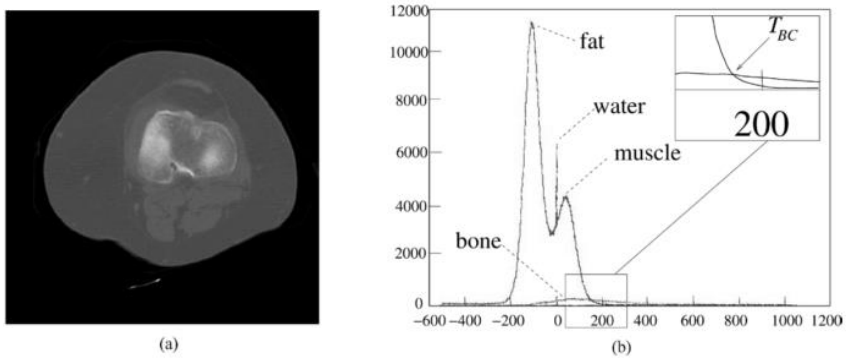


Figure 2.2 Histogram (a) Slice of a tibia (b) Histogram of HU for the over-lapping values of bone and non-bone[14].

The next step, presented by J. Zhang *et al.* [13]**,** is to discover all the pixels that belong to the boundaries of the bone. Then, for each of those elements, a bimodal histogram is constructed, using the intensities of the pixels, and it is assumed that the intensity those pixels have is a composition of two different Gaussian distributions, one for bone and one for non-bone [15]. Considering the

characteristics of those distributions, a Bayes decision rule is used to classify each element according to the majority of the tissue included in it. In case it is considered as non-bone it is tagged as such and the algorithm continues to the next element of the boundary. Once all these elements have been analysed, the algorithm discovers the boundaries of the bone again, repeating the processes of this paragraph. The processes are repeated until there are no non-bone pixels in the boundary, when it is considered that the segmentation process is finished and the image fully segmented.

This method can be extended to three dimensions, with some changes. Instead of considering an 8-pixel neighbourhood for the flood-filling and the boundary algorithms, it is considered a 10 voxel neighbourhood. The voxels that are in the slices directly above and below the one considered are farther away than the directly adjacent voxels in the same slice, as such it is only considered the closest voxel of those slices [13].

2.2 Random Walks for Image Segmentation

Leo Grad *et al*.[16] developed a segmentation method that labels the pixels of the image to be segmented by answering the question: "Given a random walker starting at this location, what is the probability that it first reaches each of the K seed points?". Before explaining how the strategy works, it is necessary to explain what a random walker and what seed points are. A seed point is an element of the image, a pixel, that is tagged with information, in this case that that pixel is part of an object. A random walker, in this case, is an agent that moves from pixel to pixel in a random direction on each step. By doing so, the walker will end up at one of the seed points, where it stops. There is a different probability that the walker reaches each of the seeds. Each one of those seed points is a pixel that is part of a different object in the image. As mentioned in the question, the algorithm requires a few of those seed points, provided by the user, before it can even begin.

The basis of this strategy is to label the unseeded pixels with the value of the seed where it is most probable to reach first. However, doing this manually is highly impractical as it requires a lot of different calculations for the labelling of a single pixel. As such, the process is very slow and requires a lot of computational power to segment the whole image.

Under normal conditions, all the possible directions for the next step are equiprobable, however, that is not the case in this method. The probability of each direction is modified according to the characteristics of the image and the properties of the pixels involved in the beginning and the end of the step. In order to save the information regarding the probabilities of the steps a weighted graph is created with as many nodes as the image has pixels. The edges have the probability of the step and the nodes will have the probability of a random walker starting at that position has to reach one of the seed points. This graph will be used to help give the answer to the question presented in the beginning of this section.

Leo Grad *et al.*[16] concluded that it is possible to determine the probabilities of each of the unlabelled elements to reach one of the seeds, by solving for x^s, the equation presented in formula 2.1. By solving this equation for each of the seeds, it is possible to label all the pixels of the image, according to the seed to which they have the highest probability of reaching first.

$$L_U x^s = -B^T m^s \tag{2.1}$$

In formula 2.1, x^s is a matrix that stores the individual probabilities that each unseeded element has of reaching the seed point, *s,* before any of the other seeds. Even though this formula only presents the calculations for one seed point, it is still required more than just one seed in the process, they will just be calculated separately.

The L_U and B^T elements are both obtained from the combinatorial Laplacian matrix L, as it is shown in formula 2.2. This Laplacian Matrix L, is constructed using the formula 2.3. Where i and j are elements of the image, w_{ij} is the weight of the edges in the directional graph mentioned above and that weight can be calculated using the formula 24. In the weight calculation formula, β is a free constant, g_i and g_j are the intensities of the elements i and j respectively. Returning to formula 2.3, d_i is the sum of all the weights in that column, as it is shown in formula 2.5.

$$L = \begin{bmatrix} L_M & B \\ B^T & L_U \end{bmatrix} \tag{2.2}$$

$$L_{ij} = \begin{cases} d_i, \ if \ i = j \\ -w_{ij}, \ if \ i \ and \ j \ are \ adjacent \\ 0, \ otherwise \end{cases} \tag{2.3}$$

$$w_{ij} = e^{\beta(g_i - g_j)} \tag{2.4}$$

$$d_i = \sum w_{ij,} \ for \ every \ pair \ of \ spels \ possible \ for \ the \ element \ i \tag{2.5}$$

In formula 2.2, L_U is the unseeded component and L_M is the seeded component, as such, the nodes that are seeded must be considered first when building the Laplacian matrix to correctly separate the two. m^s defines the labelled seed nodes. m^s has number of rows equal to the number of seeds, the row that corresponds to the seed considered has value one and the rest have value zero. The final segmentation is obtained when each node of the graph created above is labelled according to the seed, to which the element the node represents, has the higher probability of reaching first. It is shown in Figure 2.3 an example of a segmentation using the random walks algorithm. Because of the labels being attributed according to this probability, the method requires at least two seed to work, otherwise the solution would be all the nodes with the same label. Because all the probability would be always 1 as the walkers could not stop anywhere else. This also means that the sum of all the values calculated to each node have to be

1, because what it is calculated here are the probabilities of the walker reaching one place or the other, it cannot leave the map. This means, that it is not necessary to calculate the last probability system.

(a) (b) (c)

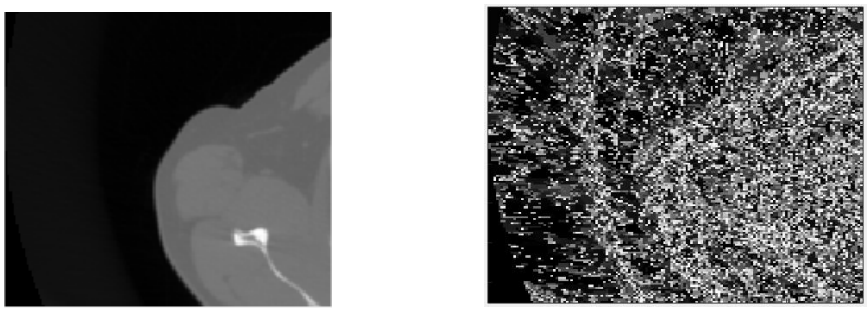
Figure 2.3 Segmentation using random walks. (a) Original image, with object and background seeds represented as green and blue respectively. (b) Probability map of the object seed. (c) final segmentation with the object shown as black and the background as white.

2.3 The Segmentation of 3D Images Using the Random Walking Technique.

J. Goclawski *et al*. [17] modified the method presented in section 2.3 to be more efficient when considered 3D volume images. The size of the graph used on 3D images is greatly increased, when compared to 2D images, as it needs to include all the voxels of all the slices in the volume. This causes the amount of steps that each walker can take to increase greatly with each new slice considered, because the walker moves randomly and can move backwards. As a result, the amount of computation is a lot larger for 3D than for 2D, because of the increase in the size of the graph leads to an increase of the elements in the combinatorial Laplacian matrix.

The authors present a solution to deal with these problems, that consists in grouping up similar voxels together to create super voxels, by doing so, the amount of nodes that exist on the graph is reduced. The super-voxels are created by randomly selecting one of the voxels in the image and label it as a super-voxel. This super-voxel will be saved as having the intensity value of this first labelled voxel. As soon as one is labelled in this manner, the next step of the algorithm is to

search in the neighbour voxels of that super-voxel to analyse if they have common proprieties. In this case, if the difference between the intensity value of the neighbour voxels and the intensity of the super-voxel is less that ΔI. In case the voxels meet these requirements, they are labelled as part of that super-voxel. These search and grouping stages are repeated until the super-voxel has reached a certain size or there are no more compatible neighbour voxels to be added to it. The process then restarts and labels a new super-voxel until all the image has been processed. Each super-voxel is labelled numerically starting from one. The super-voxels are created in the slices, 2D images, because the voxels are much closer to each other inter-slice than they are between slices. In Figure 2.4, image (a), it displayed part of a CT image and in image (b), there is the image separated by the super-voxels, the structures are not discernible, because the values displayed for the super-voxels are the labels.



(a) (b)

Figure 2.4 Super-voxel (a) Original image. (b) Image separated by super-voxels.

The next step is to use the super-voxels created to form an irregular image graph. To do so, each of the super-voxels was considered as one of the vertices of the graph, however the graph edges are not regular, in the sense that each vertices is connected to either 4 or 6 other vertices. Instead, each super-voxel can be located next to several other super-voxels. To build the graph edges, only the voxels that make up the boundaries of the super-voxels were considered. The edge is only built when one of the voxels of the boundaries is next to a voxel that makes up the boundary of another super-voxel. In Figure 2.5, it is represented a voxel and the 6 closest voxels to it, one in each of the faces of the voxel in grey.

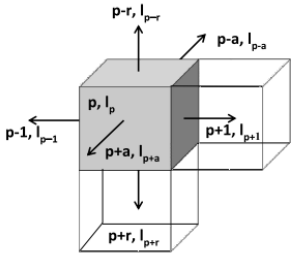


Figure 2.5 Voxel of index p and the 6 closest neighbour voxels.

The super-voxels are evaluated one at a time, however this creates repeated edges, because of the nature of the process. To prevent this, it was created a directional image graph, where the edge is only created from the super-voxel with the lower label to the higher labelled super-voxel. This means the stored connections between two super-voxels only have one direction, saving memory space computation time and preventing computational errors. The value of the graph edge is calculated using the formula 2.4 shown in the last section of this chapter. For the values of $g_i$and g_j, it is used the intensity value of the super-voxel. The random walks technique itself is not changed and the authors claim that the computation time is greatly reduced without loss in segmentation accuracy.

2.4 Fuzzy Vonnectedness

Udupa *et al.* [18] considers that images are naturally fuzzy and by proxy so are all the objects in it, as such, when attempting to retrieve this objects from the images the fuzzy nature should be taken into consideration. The authors consider that there are two important characteristics when describing an object in an image in a fuzzy setting: hanging togetherness and grade composition. Every image is composed by many basic elements, the authors call them spels whether they are two dimensional, pixels, or three dimensional, voxels. Hanging togetherness describes how the spels that hang together in a certain way form an object, may it be a bone or a muscle in a CT image. These objects have certain characteristics common to all the elements, but also have grade composition. Grade composition is based in the different variations of intensity in an object, for example, a bone is not only constituted by its dense exterior, but also by its softer interior.

The fuzzy connectedness method relies on the relations between connected pixels. As such, it was necessary to develop a function that allows the algorithm to recognize every connection possible between adjacent pixels, this property of the pixels is called Adjacency. In any image, its different basic elements have a fuzzy adjacency relation, which is bigger the closer the elements are, the immediate neighbour elements will have a higher value and the farther away ones will have a lower relation. However, adjacency alone is not enough to describe the connectedness of the points, as such, another concept is introduced by the authors called affinity.

Affinity is the fuzzy connectedness strength between two different spels in an image, the higher its value the more the two spels are connected. It takes into account the proximity between points, the adjacency, as well their intensity levels, the closer the value of their intensities the higher the affinity value will be. However, when connecting two points there are a lot of possible ways to connect them, there are many possible paths to take and each one will have different values according to the affinity of the points between them. To know the strength of the connections is to know the strength of the affinity of all the points in the path, which is determined by the weakest link. Of all the paths possible the one with the highest strength is always chosen.

The algorithm also needs a Seed, one or more elements, pixels or voxels of the image to begin. This initial point or points are considered as being part of the object that is intended to be segmented, from there, it analyses all the points around it and utilizes the connections between all the spels on the way to build an image, a map, of the affinity of each one of those elements to the seed point. The final product is an image that contains the strength of the fuzzy connectedness of each one of pixels in the original image to the seed point(s). From there a simple threshold can be utilized to extract the segmented object as the spels that belong to that object will have a higher affinity to the seed than the rest of the elements of the image.

2.4.1 Relative Fuzzy Connectedness

A different fuzzy connectedness concept is also presented, that is the Relative Fuzzy Connectedness[19]. In this process the fuzzy connectedness algorithm is ran more than once, one for the object that is intended to be segmented and another for the background. As the name suggests, this method is relative and takes into account both of this segmentations and the results obtained are relative to the individual ones obtained. Each spel of the image is either considered to be part of the background, or part of the object, according to the strength of connectedness to the seed chosen for each. If the connectedness is stronger to the seed of the object, then the spel is considered to be part of the object, on the other hand, if the connectedness is stronger to the background, that spel is considered to be part of the background.

2.4.2 Iterative Relative Fuzzy Connectedness

Following on the previous method, Udupa *et al.* [20] developed another concept called Iterative Relative Fuzzy Connectedness. The principle behind this method is to refine the original segmentation with an interactive process until to improve upon the segmentation, to achieve better results. In medical imaging, objects normally have a core that belongs to that object that is very easy to detect, however that same object also has subtle edges and fine parts that also belong to it. These parts are harder to segment, especially if the image has many similar objects that are close to one another. In this situation, those finer parts between the different objects become more diffuse and blurred. This effect is quite visible in the spine where the different vertebrae are very close to one another as well as the ribs that are held together to the vertebra, their limits are very close to one another, sometimes having a single element serving as a boundary to divide them. The spels in that region are hard to segment, as the affinity of the bone intended to be segmented and the affinity of others bones are both high.

The solution presented consists in an iterative component where an initial relative fuzzy connectedness segmentation is run, with seeds in the object intended to be segmented and the background. The region of the object to witch there is no doubt that it is indeed part of that object, this means the region that has higher affinity to the seed of the object, is considered as segmented and saved. Then, until there is no change in the saved region of the object, another segmentation using the relative fuzzy connectedness segmentation is done, with the saved area not being considered in that segmentation. In case there are spels that have a higher affinity to the seed of the object they are also considered part of the object, are also saved and not used in the next segmentation. This algorithm is later explained in more detail.

2.4.3 Scale-based Affinity Function

While the previous methods focused on the algorithm itself, there was a study in the affinity formula, that is more sophisticated than the original one that simply relied on intensity and adjacency between the different spels [21]. This new formula is more complex and presents two new components: homogeneity based and object feature based. The object-based component does not take into account the homogeneity of the path between spel, meaning that between two spels of the same object there can be a big difference in intensity of the spels. The homogeneity based component takes into account the differences along the path between two spels. Both components are insufficient on their own. The first because it is possible to find a contiguous path in the domain that is very similar but do not belong to the object; the second because the scene can have two similar objects next to one another that it considers it as being just one object [21].

This affinity function is different from the ones, because it is a local process, where the only spels considered are the ones that are direct neighbours of one another. Because of this, it is first necessary to determine which spels in the neighbourhood are part of the object to calculate the components of the affinity function. This is where the scale concept is useful. Roughly, it can be described as all the pixels

located inside a circle around the seed pixel. These pixels need to belong to the same object as the seed to be considered as part of the Scale [22]. The circle just mentioned is normally defined as a hypperball. However, to use the scale to segment the object, it is needed a basic object definition to at least know the size of the object to be segmented. That region is defined by utilizing an algorithm that creates a hyperball that starts from the seed and grows along the object according to a homogeneity function, only stopping when the ball locates spels that do not belong to the object. After defining the scale, the spels that are contained in it can be now used to determine the affinity.

All the fuzzy connectedness methods will be explained in more detail in the next three chapters, where all the functions will be displayed analysed and tested with 3D CT image volumes.

Chapter 3. Fuzzy Connectedness

As stated in the introduction, the purpose of the Fuzzy Connectedness method is to segment the bone shown in the CT image volume. Two problems are considered two dimensional segmentations and three dimensional segmentations. From the CT scan a single slice is used for two dimensional segmentations, or a collection of ordered and consecutive slices, called a volume, is used for three dimensional segmentations. The two problems are different from one another but a similar approach will be used for both, with the necessary alterations for each one. Udupa *et. Al* [6] used the expression spel as a means to refer individual elements of an image, pixels in two dimensions and voxels in three, and this connotation will be used here as well.

The fuzzy connectedness segmentation method, developed by Udupa *et. Al* [6], as the name suggests, takes into consideration that the images are fuzzy by nature. As it was explained in the introduction, these images are susceptible to partial-volume effects, patient motion, and noise, that make the image fuzzy in aspect. Especially in a medical setting with live patients that might be in pain and moving during the time it takes to capture the images. As such, it makes sense to adopt an approach that is able to deal with the uncertainties and variations present in such images.

The main objective of this segmentation technique is to identify an object by qualifying the spels of an image according to their connectedness strength to a seed point. The closer the intensity value of a spel is to the intensity value of the seed, and the less distance there is between those two elements in the image, the higher their connectedness strength will be. The higher this strength, the more alike the spels are and the more probable it is that the spel belongs to the same object as the seed. As such, by calculating the connectedness strength from all the spels to the seed, it is possible to determine which are part of the same object as the seed and which are not.

When two points are right next to each other, it is simply necessary to calculate the fuzzy connectedness between them. However, when they are further apart, it is necessary to create a path of consecutive elements of the image and calculate the fuzzy connectedness between each pair of consecutive spells of the path. The path is nothing more than an ordered list of the spels that lead from one point to the other. The connectedness strength of this path is as high as the lowest connectedness strength of each pair of elements that make up that path. The strength of connectedness between any two points in an image is defined by the strongest path between those points [18]. It is possible to determine that strength of connectedness using a local fuzzy relation concept called affinity.

The affinity function will be explained in the next section, as well as the segmentation problem, considering two dimensions first, as it is easier to calculate, visualize and explain then changed to cover the third dimension. This section will also cover the algorithm itself; the multiresolution and the window processes that were developed; the experiments involved in that development and their results.

3.1 Fuzzy k-Connectedness

Before the affinity function can be explained, it is necessary to first explain the adjacency function, as it is used by the affinity to obtain its results. In this section both adjacency and affinity will be explained in more detail.

3.1.1 Adjacency

Adjacency is a function of the distance between two points, it is symmetric, and uses a Euclidean distance formula that distance [18]. The adjacency function is showed in the formula 3.1.

$$\mu_{\omega}(c,d) = \begin{cases} \frac{1}{1+k_1\left(\sqrt{\sum_{i=1}^{n}(c_i-d_i)^2}\right)}, & if \ \sum_{i=1}^{n}|c_i - d_i| \leq n \\ 0, & otherwise \end{cases} \quad (3.1)$$

Where c and d are the two spels; n is the dimension of the spels, in case it's a pixel its value is 2 and in case it's a voxel its value is 3; c_i and d_i are the individual coordinates and k1 is a non-negative constant.

As it is possible to see in Figure 3.1, each pixel has eight immediate neighbours, with which it can connect. However, it was only considered the 4 pixels indicated in red in that figure to form the connections, with the blue pixel *P*.

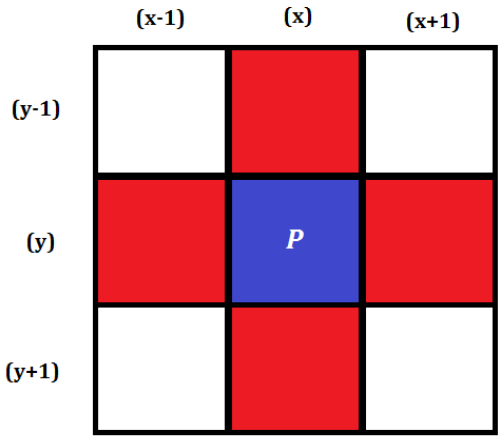


Figure 3.1 Simple four neighbours adjacency [28].

A sparse matrix was created to contain all the relations between spells, saving the coordinates of both the spels as well as the adjacency strength. This matrix connectivity information will be later used in affinity to correlate the different spels in an image to one another. The matrix only needs to have the information of each spel and its immediate connections, since all the spells are connected it is possible to construct all possible paths simply by considering the immediate neighbours of each spel. The sparse matrix only depends on the size of the image and not on the image itself, as the adjacency only considers the positions of the spels to one another. It was used the affinity function described in [24].

3.1.2 Affinity

Affinity is the link between the concept of the fuzzy connectedness, explained above, and the how it can be used to segment images using an algorithm. It takes into account the proximity of the spels, the adjacency, as well as their intensity levels, the closer the value of their intensities the higher the affinity value will be [1] .

There are an infinite number of possible paths to connect the two points, however, only the one with the strongest fuzzy connectedness is of interest. To determine the strength of path, it is necessary to know the affinity of all the consecutive pairs of elements that form the path. Because that strength is determined by the lowest affinity value between the spels that form the path. As such, Udupa *et al.*[1] determined that the strongest path between two non-consecutive points c and e, can be obtained by calculating the maximum of the minimum of the affinities between each pair of its consecutive elements, as it is shown in formula 3.2.

$$\mu_K(c,e) = \max[\min(\mu_K(c,d), \mu_K(d,e))] \quad (3.2)$$

Because of this, it is possible to determine the relations between all the pixels present in the image. However, to calculate all the connections between every single pixel in an image would be too time consuming, as it would require many iterations to calculate the best path for each of the possible pair of spels.

Affinity can also be used in the same way the connectedness was described in the beginning of the chapter, by calculating the affinity of each spel in the image to a seed, *o*. This simplifies the computation needed, since there is no need to compare every element of the image to every other element, instead, it is only needed to calculate the affinity of the seed spel *o* to all of the others elements of the image.

The values of the affinity can be stored in an image, with the same size of the original image, where each element of said image stores the correspondent

affinity value to the seed. This image, represents the map of the affinity values facilitating the visualization of the structures that are more similar to the seed spell and that can be part of the same object.

The most basic affinity function is presented in formula 3.3. Where k_2 is a non-negative constant and $f(c)$ and $f(d)$ are the intensity values of the spels in the image and $\mu_w(c,d)$ is the adjacency function. This affinity function is very simple and relies simply in the difference of intensities between adjacent spels.

$$\mu_k(c,d) = \frac{\mu_\omega(c,d)}{1+k_2|(f(c)-f(d)|} \tag{3.3}$$

However, this affinity function is too simple, because the similarity of the intensity of two spels that are immediately next to each other is the only criteria this function covers. As such, the affinity of the spels that are far away from the seed is only dependent on the variations of the spells of the path, which can cause high affinities on spels that are not part of the object, mainly when the boundaries are soft or blurry. This happens, because as long as the variation of the intensity consecutive spels along the way is small, their affinity will be high and they will be considered as part of the object where the seed is located.

Considering the problems of the previous formula, another function was considered. It is presented in the equation 3.4, it considers that the intensity values of the spels in the object intended to be segmented are within a certain range described as a mean with a standard deviation.

$$\mu_k(c,d) = \begin{cases} \mu_\alpha(c,d)\left[\omega_1 h_1\left(\mathrm{f(c)},\mathrm{f(d)}\right) + \omega_2 h_2\left(\mathrm{f(c)},\mathrm{f(d)}\right)\right], & if\ c \neq d \\ 1, & if\ c = d \end{cases} \tag{3.4}$$

Where ω_1 and ω_2 are weighted mean components, that follows the rule presented in formula 3.5; $\mu_\alpha(c,d)$ is the affinity function; the h1 and h2 functions are chosen from the formulas 3.6 to 3.9

$$\omega_1 + \omega_2 = 1 \tag{3.5}$$

$$g_1(f(c), f(d)) = e^{-\frac{1}{2}\left[\frac{\left(\frac{1}{2}\right)(f(c)+f(d))-m_1}{s_1}\right]^2} \tag{3.6}$$

$$g_2(f(c), f(d)) = e^{-\frac{1}{2}\left[\frac{|f(c)-f(d)|-m_2}{s_2}\right]^2} \tag{3.7}$$

$$g_3\big(f(c), f(d)\big) = 1 - g_1\big(f(c), f(d)\big) \tag{3.8}$$

$$g_4\big(f(c), f(d)\big) = 1 - \ g_2(f(c), f(d)) \tag{3.9}$$

Where m_1 and m_2 are the means, s_1 and s_2 are the standard deviations of the correspondent means. By using different values for the weighted means and choosing different functions for h1 and h2, it is possible to create many combinations of the equation presented in 3.4. As such, it is possible to adapt it to different situations and images. These affinity functions will be explained in more detail in the next section.

3.1.2.1 Function differences

Before explaining the affinity functions, it is needed to explain the m_1, m_2, s_1 and s_2 constants. These represent the mean and standard variation of the intensity of the spels that are part of the object of interest.

The function g_1 is fairly straightforward to understand and utilize, the smaller the difference of the intensity of the spels to the mean chosen, the higher the intensity value. It also takes into account the standard deviation, the higher its value, the bigger the range of the intensity values that are considered to be part of the object, resulting in a higher affinity value for the spels in that range. These properties are illustrated in Figure 3.2. It is a probability distribution, the black stripe in the middle of the curve represents the mean and the area in purple represents the region between m1-s1 to m1+s1. The orange area is the region between m1-2s1 to m1-s1 in the left side of the image and m1+s1 to m1+2s1 in the right side. The farther away the intensities of the spels are from the mean the lower their affinity values will be. The spels that have their intensity value in the red zone will have high affinity and are most likely part of the object. This affinity function will work

specially well if the intensity values are close to one another, as the standard variation will have a small value. Which reduces the amount of spels that will have high affinity, causing the pixels in the object to stand out by having high affinity values in contrast with the low intensities of the background.

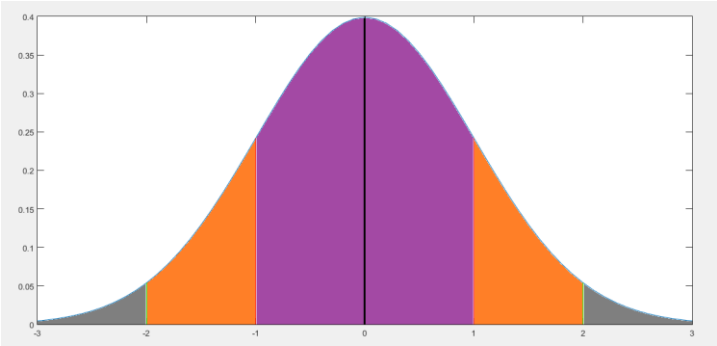


Figure 3.2 Normal distribution.

The function g2 takes into account the difference of the intensity of adjacent pixels instead of the intensity values of the spels. The closer the difference of the spels is to the mean, in this case m2, the higher its affinity value will be. This function is useful when the object intended to be segmented has slight variations of its intensity along its structure. Unlike the first function, the values do not need to be within a certain range, only the differences of intensity do, as such, it can include areas that can be described as a gradient of intensities either ascending or descending. It can be used in cases where the object has well defined boundaries that have big difference in value between inside and outside of the object as well, because the affinity values will drop abruptly in those places confining the segmentation in those thresholds. Like the other function, this one is also influenced by the standard variation, the higher its value the bigger the range of values of the differences of the intensities of the spel. The more uniform this gradient is, the lower the value of s2, because the variations will all have similar values.

The function g3 is a case of reverse affinity, also called enmity, of the function g1. This means, that the affinity has lower value the more similar the spel intensities are to the mean value considered. This function can be particularly useful when

the object intended to be segmented is surrounded by a certain structure, or background, where its spels have intensity values that can be described as a standard distribution with a small standard deviation. Such as a bone surrounded by a big muscle that has similar intensity values along its structure. It is useful in this situations, because the pixels that are part of the object are distinct from the background, as such, they will have high affinity. Since the muscle is surrounding the bone, as soon as the algorithm starts to evaluate the muscle it will present low affinity values, presenting the bone a region with high intensity and the rest of the image with low intensity.

The function g4 is also a case of enmity. It is the reverse affinity of the g2 function. However, m2 and s2 cannot be given high values, because it will consider the normally bad paths has having high affinity and the connectivity map will be presented as if all spels had high connectivity. Like in the previous function, it is best used when the area surrounding the object has more uniform characteristics than the object itself. In this case, it is best used when the area that surrounds the object has slight variations of its intensity but not necessarily around a certain mean, like a gradient, as explained in the g2 function. However, since high values of variation causes algorithm to not work as well, it is best used when the area surrounding the object has a lower intensity variation than the object.

3.2 The Algorithm

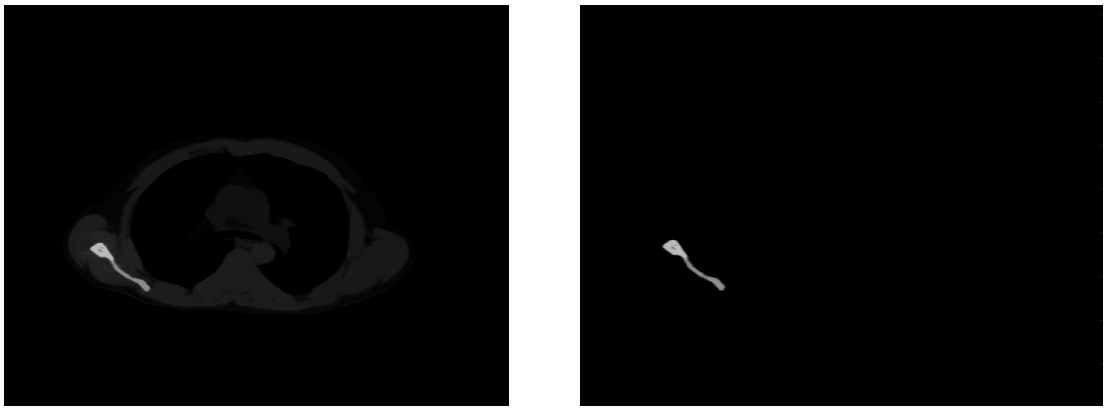
With the purpose of implementing the concepts presented above, the algorithm is described in this section. First it is explained the two possible algorithms when calculating the affinity of the spels of the map, and then the meta-code of the chosen method is exposed, as well as its explanation.

Two possible approaches to the generation of the affinity map are presented, the first, called $K\theta xFOE$, in which it is considered a minimal strength, θ, for the affinity of the pixels in the image. This means, that the algorithm, when calculating the possible paths between the seed o and each pixel, will stop as soon as it finds an affinity that has higher value than θ and move on to the next pixel. In case it does not find an affinity with higher value than θ to that spel, it attributes value 0 and continues to the next pixel. Because of this, the value θ has to be adapted to the object that is intended to be segmented, which means that to utilize this approach, it is necessary to have previous information on the image or on the strength of the affinity of the elements of the image to the seed.

The second approach is designated $kFOE$, in it, it is not considered a minimum value, as such, the algorithm calculates every relation possible and tries to find the best path from the seed to each pixel, returning the highest affinity possible. This process takes more time, however it presents the complete map of the image and requires no previous information regarding the image.

An example of the two methods is presented in Figure 3.3, in it, it is shown the resulting fuzzy connectedness map of the slice 79 of the first volume presented in the appendix A. To obtain the connectedness map, it was first run the $kFOE$ algorithm to obtain the affinity map displayed in image (a) and then the $k\theta_x FOE$ algorithm was run and the results displayed in image (b). There is a big difference in the time the algorithm takes, reducing the computation time from 10,01 seconds, using $kFOE$, to 0,27, using $k\theta_x FOE$. However, it is only possible to reduce the time this much because of the previous knowledge on the image and on the

structure obtained in the first segmentation. It is possible to see in image (a) that there is some affinity towards the rest of the structures of the body, however it is



(a) (b)

Figure 3.3 Comparison between algorithms: (a) Image obtained using the *kFOE* method (b) Image obtained using the *KθxFOE* method.

The algorithm that will be used is the *kFOE*, as it does not require previous knowledge of the image, even though it is slower than the *KθxFOE* method.

3.2.1 Meta-code

Input: original image C, seed point S, connectivity scene K created using the *adjacency* formula

Output: Connectedness map I

Begin

1. Create a new image I with the same size as the original one;
2. Give all the spels in that image the value 0 expect for the seed points;
3. Push all spels with $affinity(o, c) > 0$ into Q;
4. While Q is not empty do:
    a. $find\ fc = \max\big(I(Q)\big)$;
    b. id = find indexes of fc in Q and remove them from Q;
    c. $f = \min(fc,\ max_{d \in C}\ \big(K(id,\ \ d)\big))$;
    d. $for\ each\ i\ in\ f$

i. $if\ f(i) > I(i)\ then$
   1. $I(i) = f(i)$;
   2. Push all spels with $affinity(i, e) > affinity\ into\ Q$;

ii. $end\ if$;

e. $end\ for$;

5. $end\ while$;

end

3.2.2 Algorithm Explanation

The first step of the algorithm is to create a way of determining the adjacency of all the spels of the image. For two dimensions, a sparse matrix was considered to store all the possible adjacent connections. When considering the problem in three dimensions, two possibilities were considered, the first was creating another matrix to represent the z axes to calculate the distance and store all the possible combinations in a three dimensional matrix. The other was to make one of the matrices longer storing the multiple layers of information there, starting from the bottom slice, presented in the far left, to the top slice, in the far right. This means, that the matrix stores the information of one of the images and right next to it, in the same matrix, it stores the information of the next slice.

In the first case, the information is stored in a three dimensional manner, whereas the second is not. However, this is not relevant as both cases are defined by the adjacency, as it determines the relations between the different spells in the volume. As long as the relations between the voxels are all correctly presented in the connectivity scene, the slices can either be stored in any manner. It is even possible to switch the order of the slices or store them in different locations. Considering all the mentioned properties, the two dimensional sparse matrix was chosen, as it simply requires changes in the adjacency and no change in the algorithm itself. This was covered because of the way some 3D images will be displayed, with the slices next to each other in the same image.

The adjacency chosen was the immediate neighbours, this means the 4 closest spels in two dimensions and the 6 closest voxels in three dimensions. It was considered such a small Neighbourhood for the following reasons:

- The lack of improvement on the final results when considered a bigger neighbourhood;
- The increase in computation time is small in two dimension but when calculating in three dimensions it increases exponentially;
- The sparse matrix in the program used, Matlab, only allows to save in two dimensions, as such any relations between the z axes would have either to be stored in another way or created independently;

Then it is required a seed map, *FC* it is assumed a seed was already selected. A seed is one or more spels in an image that are chosen as bases. In this case it needs to belong to the object intended to be segmented. The algorithm will use this image to store the values of the fuzzy connectedness of each pixel in the original data to the seed points.

To begin the calculations a queue Q will be considered where all the spels that need to be evaluated will be stored, and their fuzzy connectedness value will be stored in *I*. Every time one of the pixels stored in Q is evaluated it is removed from it. Originally it would choose one of the elements of the queue at random, but it was altered to pick the ones with higher value to prevent obsolete calculations.

In the original algorithm it is considered one spel at a time and it retrieves information from the pixels adjacent to it, to calculate its fuzzy connectedness. In case there is any change in it is updated and all the adjacent spels are placed in the queue to be evaluated again. However, it is done in a different fashion, it is considered one pixel, from which it is already known its connectedness value and then calculate the fuzzy connectedness to its adjacent spels. The seed points are the ones used in the begging of the algorithm. It was used the fuzzy connectedness function in [25].

In case the new calculated value is higher than the ones already known and stored in *FC,* it is updated with the new value. The pixels that were updated are put in queue to be evaluated again. This doesn't bring any significant change other than being faster while producing the same results. Once *Q* is empty the algorithm stops and it is presented a map of the fuzzy connectedness of all the pixels to the seed.

3.3 Complexity Reduction Strategies

In this section it will be presented a proposed modification to the algorithm, in an attempt to decrease the complexity of the segmentation, a multiresolution and a region of interest limited processing strategies were implemented. Multiresolution is a technique that consists in the reduction the resolution of the images being studied in order to create an image where a faster but less precise segmentation can be performed. This initial segmentation will serve as a base for a more detailed and precise segmentation in the original image. The region of interest limited processing is a technique where a portion of the image is selected, limiting the amount of information that the algorithm needs to process. The portion selected has to contain all the essential information required by the segmentation algorithm in order to present the same results in a reduced timeframe. The base algorithm was already presented in the form of meta-code, and these strategies will be presented as a flowchart in Figure 3.4 and explained step by step.

The first step of the algorithm is to copy the volume and reduce the new image and down-scale it to half the size of the original image, resulting in a volume that has one eighth of the original voxels. This is done one additional time resulting in a volume that is 64 times smaller than the original one, as it was determined that reducing two times was enough to obtain good results. The end result has one fourth of the slices of the original size, because it is done in all the axis. In case the number of slices is not divisible by four, the last image is replicated the necessary number of times to meet that requirement. By doing so a multiresolution image pyramid is created.

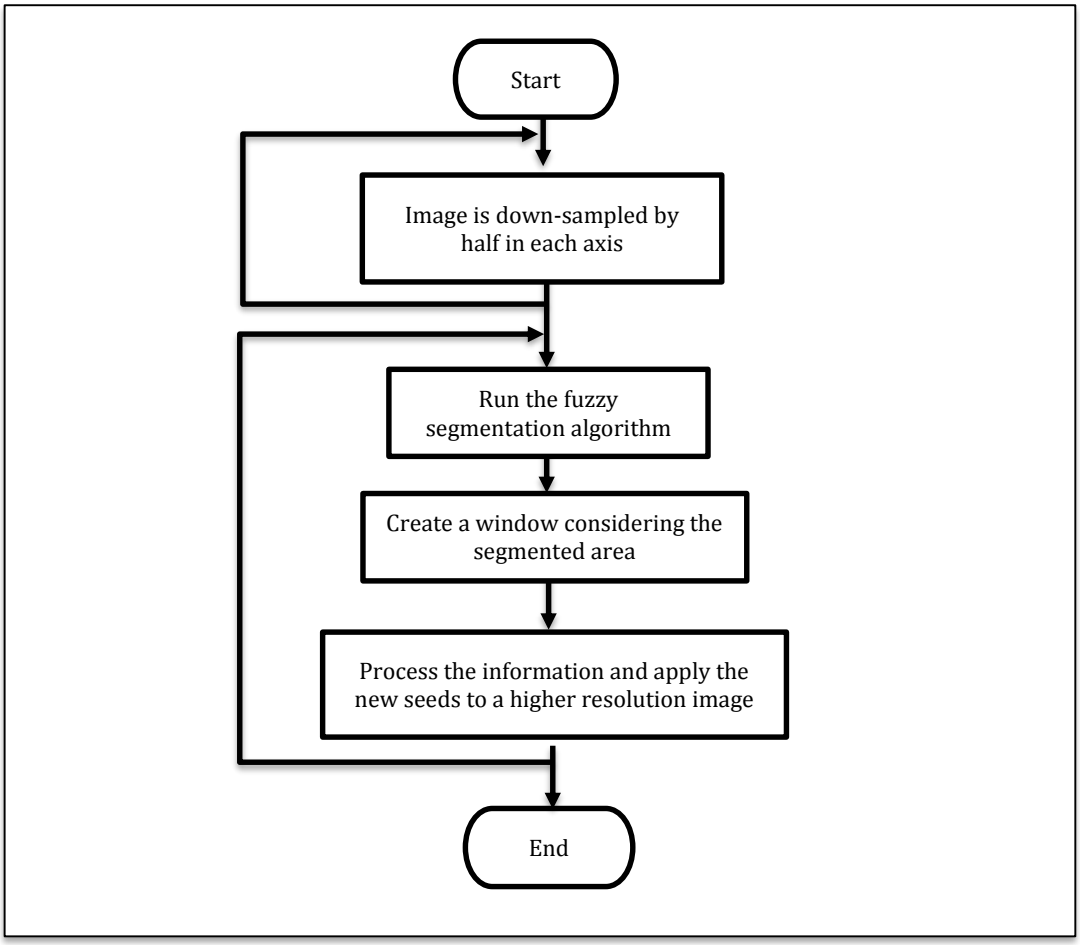


Figure 3.4 Flowchart of the multiresolution and window strategies.

The second step is to use the fuzzy connectedness segmentation algorithm in the reduced images to obtain a fast, but rough initial segmentation. Using this initial segmentation, it is possible to employ the region of interest limited processing strategy, which will be now explained. The segmented object is limited within a region that is smaller than the complete image, as such, a window can be created around said object that will be used in the segmentation in the higher resolution images that only contains this particular area. This window must be resized to the bigger images of the pyramid to surround the same region as it did in the lower resolution image.

The main function of segmentation is to reduce the information present on an image without the loss of the data that is considered important [11]. As such, using

only a portion of the full image prevents a lot of unnecessary calculations, as it is not necessary to consider a portion of the total spels. However, since the relevant information is still confined in the boundaries of the window, the final segmentation will not be affected. This translates into a faster process, as the computation time is reduced, but that maintains the results.

To create the window, a thresholding technique was run on the already segmented image, creating a binary image. Using that image, all the positions of the classified voxels are considered as part of the object in all of the slices. It is located the lowest and highest values of both the x and y coordinates and added a 3 voxel margin to create a window that is rectangular. It begins 3 voxels before the lowest coordinates and ends 3 voxels after the highest coordinates, thus making sure the object is completely inserted in the window, as it is shown in

Figure 3.5. That window is then up-scaled to fit the bigger volume, it is applied in all the slices of the volume, making sure that the relative position does not change when resizing.

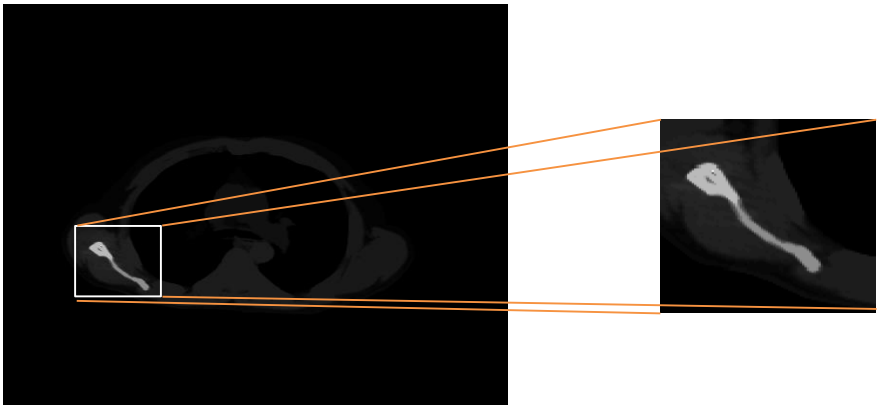


Figure 3.5 Example of the selection of the part of interest in the segmented image, using *Figure 1.1*.

The next step consists on using the information already obtained with the initial segmentation and increase the resolution. it is applied a threshold technique to the segmentation as a means of separating the object from the background,

resulting in a binary image. Since there was a resolution reduction each of these voxels represent 16 or 64 voxels of the original image, as such it is possible that some of the 16 or 64 voxels are not correctly identified. The voxels that have some chance of being incorrectly segmented are the ones present in the limits of the segmented area, as these contain voxels from the interior and exterior of the object. As such, they must be removed from to prevent their incorrect classification so that they can be classified in the higher resolution images. The resulting binary slices are then up-scaled by two and are replicated once to meet the size of the intermediary volume created before. The results of this process can be considered as the seed for the higher resolution images, because it uses the same algorithm and the seed map now has the same size of the intermediate image.

The final part of this algorithm is to repeat the steps from the step two onwards two more times. However, it is not necessary to create another window, as it is already being, as such that step will be skipped. By segmenting the intermediate image before the original one, a part of the possible incorrect segmentation will be prevented, as the process considers an increase of 16 to 1 voxels by up-scaling two times instead of the immediate 64 to 1 voxel increase.

3.4 Experiments and Results

In this section, it will be covered the experiments and results of the algorithms that were presented in this chapter. First, it will be covered the experiments in two dimensions, as it will cover which affinity functions are the best for the bone segmentations, that are the object of this project. The three dimensional experiments do not cover differences between the affinity functions used, instead it covers the experiments made to determine the effectiveness of the multiresolution and the window processes and their results. The computer used has 8GB of RAM and an Intel Core i7-4790 3.60 GHz CPU.

3.4.1 Testing the Different Affinity Functions in Two Dimensions

In this section, the experiments regarding the affinity functions presented in section 3.1.2 will be presented. When testing these affinity functions, there are a few properties to take into account, such as the ones of the object intended to be segmented, in this case the bones of the human body, as well as the ones of the surroundings of said object. When testing the affinity functions g1 and g2, it is necessary to consider the object proprieties, namely the intensities of its spels. Each function requires a mean and a standard variation values, as described before. It is considered as good values when the majority of the spels intensities are within the range of mean minus standard variation to mean plus the standard variation. When testing the g3 and g4 functions, the same properties have to be considered, however it has to be considered the background properties instead of the ones of the object.

Different values were calculated to *m* and *s* for each one of the functions in each individual scenario, in order to obtain good individual results. The *m* stands for the mean of the value of the intensity of the pixels and the *s* stands for the standard variation of the mean of the intensity of the pixels. To calculate the values of *m* and *s,* a thresholding segmentation technique was used on the image, with the value of 1150, followed by a grouping algorithm, to do a rough segmentation of the bone. In Figure 3.6, it is displayed the results of one of those segmentations.

Figure 3.6 Threshold segmentation of the spine.

The spels that were segmented this way, were then used to calculate the mean and standard deviation of their intensities to be used in the parameters of the affinity functions. However, the results obtained weren't always good. In Figure 3.7, it is displayed one of those examples, where the *m* and *s* used in the affinity function were the ones that were calculated in this manner. In this segmentation the seed of the scenario two, explained later was used. As such, it was necessary to change the values by hand by running some experiments with different mean and standard deviation values.

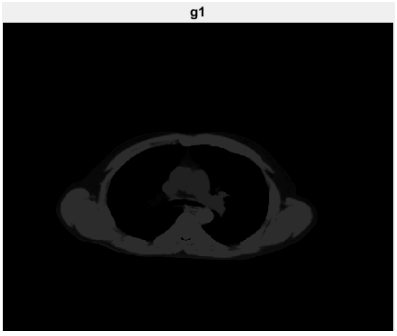


Figure 3.7 Segmentation of the spine using the calculated *m* and *s*.

While testing the different scenarios, the seed pixel and the image used were always the same to make sure the only variations occur because of the variables. After considering a number of possible scenarios, it is possible to compare the mean and standard variation values used in each one of them and take conclusions from those values. In case they are consistent along multiple scenarios, that means that that particular function and values can be used in any part of the volume. In case the values differ in most of them, it means that that function needs to be adapted to each specific situation and even though it may present good results, it cannot be used in an algorithm that will cover the whole body.

Taking into account how the affinity functions work, there are three main distinct scenarios to consider. The bones that are separated from other bones and have muscles surrounding them, e.g. the femur and shoulder blade; bones that are very close to other bones, e.g. spine and vertebrae; bones that are surrounded by a big variety of tissues and background like the vertebrae and the skull. As such, the next sections will cover the different scenarios individually.

3.4.1.1 Scenario one: Shoulder blade

The first scenario tested is the simplest of them, where the bone is surrounded by only one kind of tissue, in this case, muscle. To test this scenario, the shoulder blade was selected, as it is a thin bone with many variations of intensity along its length with no other bones touching it and surrounded by muscle. In Figure 3.8. it is shown the original image with the seed point displayed as a red point. It is located at the position (312,109) of the image in all the tests.

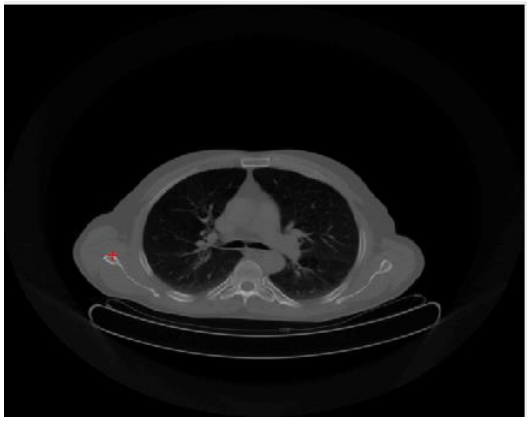


Figure 3.8 Original image. Slice 79 from the test CT. Seed located at the position (312,109).

In this slice, the bone region has a well separated intensity region from the other tissues. Because of this, it is possible to determine the values to give to *m* and *s* in the g1 function. The g2 affinity function considers the intensity variations of the image. Even though the bone itself presents many variations, causing differences of affinity along the bone, the variations from the bone to the background are greater. This effect is visible, as a portion of the bone presents a lower intensity than the other, due to the variations within the bone.

In this slice, as it is possible to see in the image in Figure 3.8, there is a region surrounding the bone, that has lower intensity than the bone. That region is muscle tissue and most of the intensity values of the pixels of that region can also be included in the range of a mean and a standard deviation. This means that it is possible to determine *m* and *s* of the g3 affinity function. The muscle that surrounds the bone has a lower variation of intensity that the bone itself. As a

result, it is possible to use the affinity function g4 by giving *m* and *s* lower values than the ones considered in g2. The bone can be segmented using this affinity function, because the variations in the muscle are smaller than the ones in the bone.

Affinity function	G1	G2	G3	G4
mean	1700	400	1050	40
Standard variation	300	200	50	120

Table 1 Mean and standard variation values obtained for the first fuzzy connectedness segmentation scenario.

The mean and standard variation values obtained are displayed in Table 1 and results of the segmentations are shown in Figure 3.9. Each one of those images is a fuzzy connectedness map where the intensity of the pixel. Each pixel represents the affinity of that pixel to the seed.

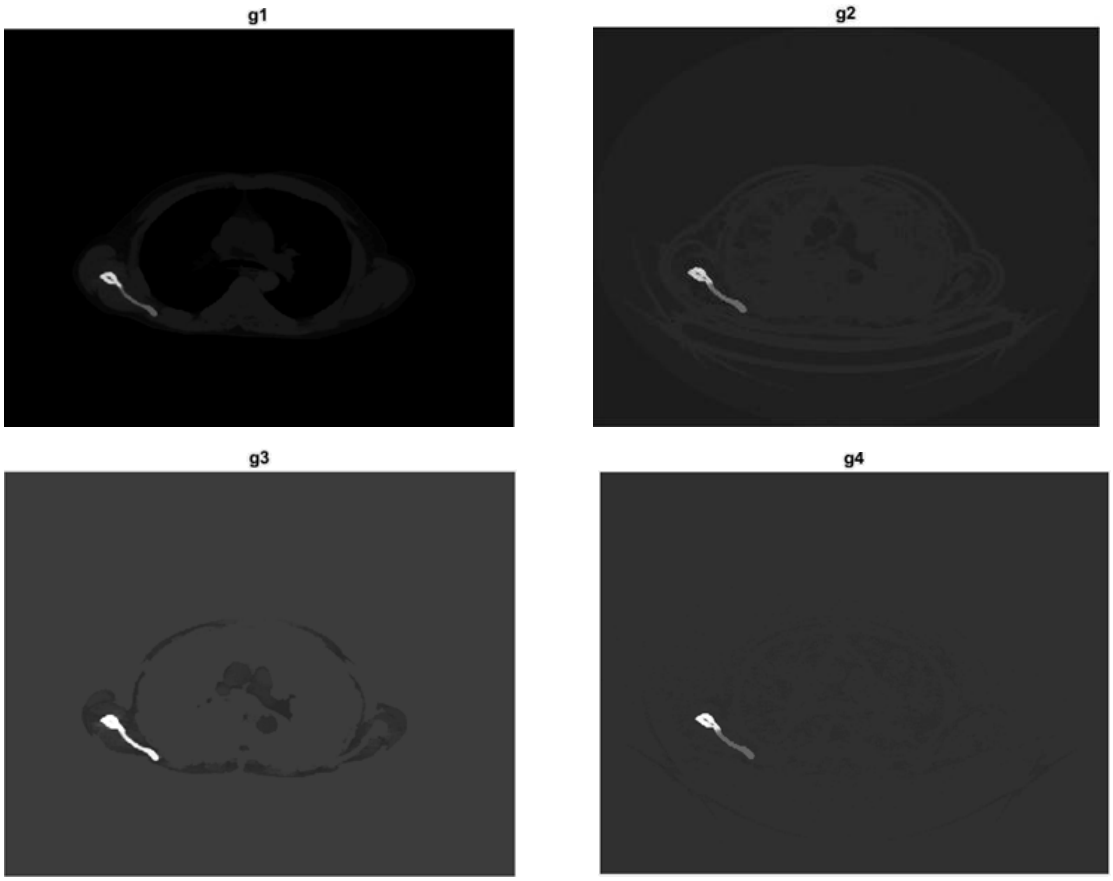


Figure 3.9 Affinity function results obtained by segmenting the image in *Figure 3.8*. Each image is labelled with the affinity function used to obtain it.

In the results obtained it is possible to distinguish the bone from the background in all of the affinity function results. However, it has a higher affinity in some of the functions than the others, especially in the g3 function. In Table 2 is displayed the time each segmentation took.

Affinity function	G1	G2	G3	G4
Time(s)	17.3	5.9	3.4	2.8

Table 2 How much time it took for the algorithm to run each of the affinity functions to complete the calculations in the scenario presented in 3.4.1.1.

3.4.1.2 Scenario two: Spine

In this scenario, the same image was used, but it was chosen another bone to segment. It was chosen the vertebra, where there is close proximity between bones, in this case the vertebra and the ribs. The seed point located at (360,240) directly in the spine, as it is shown in Figure 3.10.

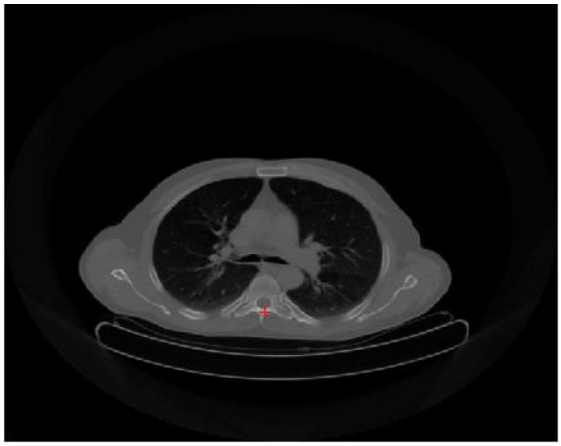


Figure 3.10 Original image. Slice 79 from the test CT. Seed located at the position (360,240).

In this scenario, the bones are not completely surrounded by muscle like in the shoulder, as such the variables of the functions g3 and g4 have to be altered. For the first one of the functions, it is necessary to lower its mean value and increase the standard deviation, in an attempt to consider both the muscle the air inside of the lungs. However, this causes the variations of the tissues around the bone to be quite significant.

In this scenario, the variations in the bone are even bigger than in the shoulder blade with some differences between the intensity of immediate spels being higher than 100. As such, the g2 function also had to be altered in order to consider these variations.

Affinity function	G1	G2	G3	G4
mean	1700	400	900	40
Standard variation	300	200	300	120

Table 3 Mean and standard variation values obtained for the second fuzzy connectedness segmentation scenario.

The mean and standard variation values used for each function are displayed in Table 3 and the results of the segmentations are shown in the Figure 3.11.

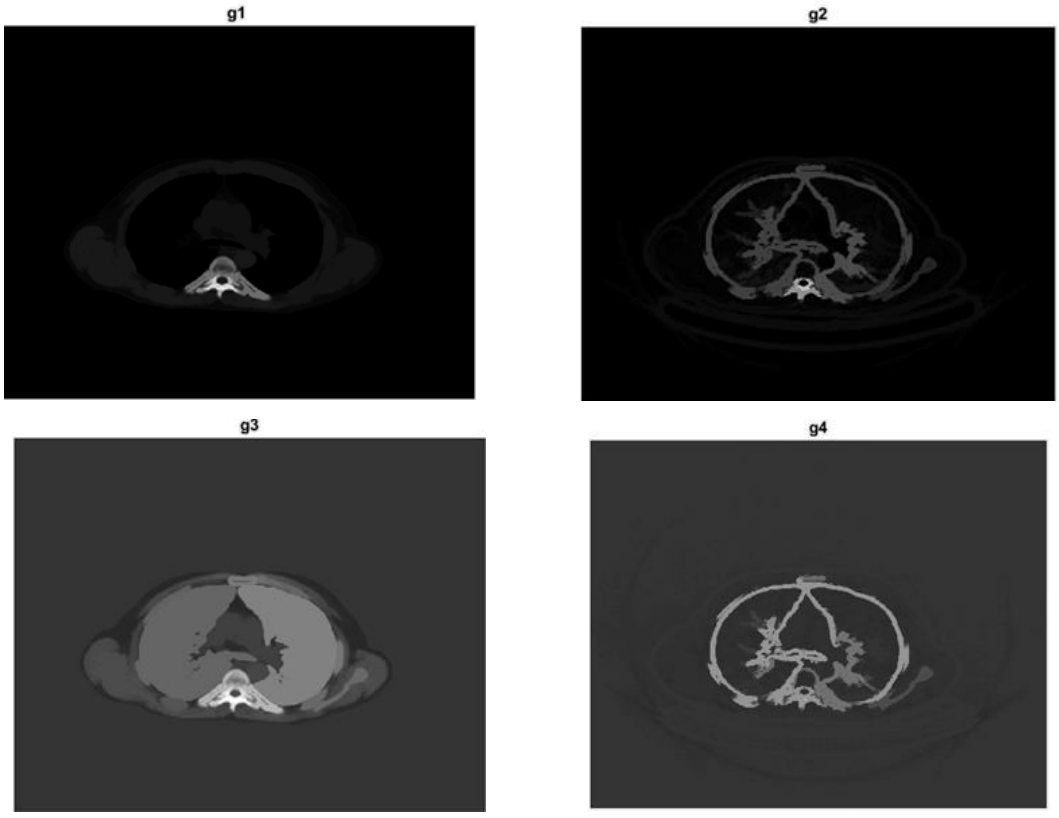


Figure 3.11 Affinity function results obtained by segmenting the image in *Figure 3.10*. Each image is labelled with the affinity function used to obtain it.

In the first image of the Figure 3.11, the g1 function, it is clearly visible the spine bone, the target of the segmentation, stands out from the rest of the image.

However, the algorithm branched to the ribs as well and they are presented as having high affinity. There is a region that has a higher intensity due to being closer to the seed and having a higher bone density. The softer parts inside of the vertebra and the ribs have lower affinity.

In the second image of the same figure, the g2 function, it is possible to see that the pixels that represent the walls of the lungs have a much higher affinity than they should have. However, there is an area with higher intensity in the spine than in the rest of the image, but it does not compromise the entirety of the bone. That area only represents the denser part of the bone.

In the third image, the same structures as in the first affinity function are visible. However, the background structures have a much higher intensity, giving the pixels that represent the lungs an affinity value well above what can be considered normal. In this particular case, it does not have a big impact in the end result, because with the right thresholding technique both image from the g1 and g3 affinity functions would have similar results. However, the threshold needs to be handpicked for this situation. In another similar scenario, it might not be possible to find such a solution.

In the last image the algorithm had a similar behaviour to the second affinity function. However, the affinity values for the vertebra and the walls of the lungs were almost identical, making the task of separating the two almost impossible. The time each affinity function took to complete is displayed in Table 4.

Affinity function	G1	G2	G3	G4
Time(s)	18.2	6.7	4.5	3.2

Table 4 How much time it took for the algorithm to run each of the affinity functions to complete the calculations in the scenario presented in 3.4.1.2.

3.4.1.3 Scenario three: Skull

In this scenario, the slice 30 from the same volume as the other ones was used. In this region, the tissue outside the bone is different from the tissue on the inside of the bone. On the outside there is skin and air and on the inside there is the brain. In his scenario there is also a limit, the contact of the skin with the air, that has a very abrupt intensity variation. The seed point located at (360,240) directly in the spine, as it is shown in Figure 3.12.

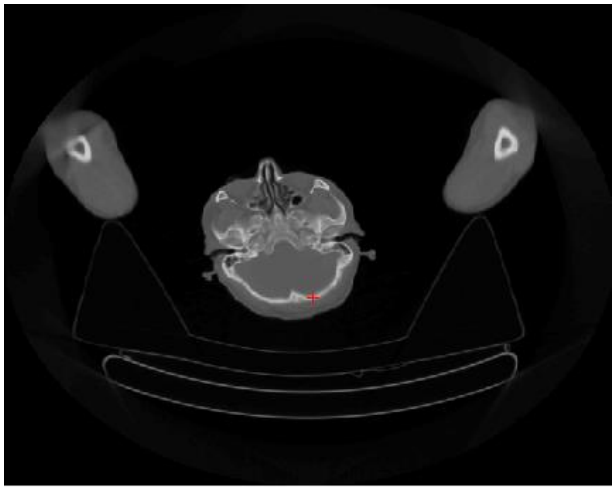


Figure 3.12 Original image. Slice 30 from the test CT. Seed located at the position (315,260).

In the skull region, there is some surrounding tissue and in some regions, it might be too thin, as it is only skin that separates the bone from the air. The skull itself is not too thick and has a big variation in shape, especially in the region of the face, which is also a good testing factor for the sturdiness of the affinity function, to correctly segment the spels of the skull.

Along the skull, there is a significant variation of the intensity of the pixels, as it is possible to observe in the Figure 3.12. As it was mentioned, the type of tissue that surrounds of the bone isn't constant along all of its surfaces, requiring some adjustments on the g3 and g4 functions as well. In an attempt to consider all the pixel intensities and variations, it was considered bigger standard deviations.

Affinity function	G1	G2	G3	G4
mean	1700	400	900	40
Standard variation	300	200	300	120

Table 5 Mean and standard variation values obtained for the third fuzzy connectedness segmentation scenario.

The values used in each of the affinity functions are shown in Table 5 and the results of the segmentations are shown in the Figure 3.13.

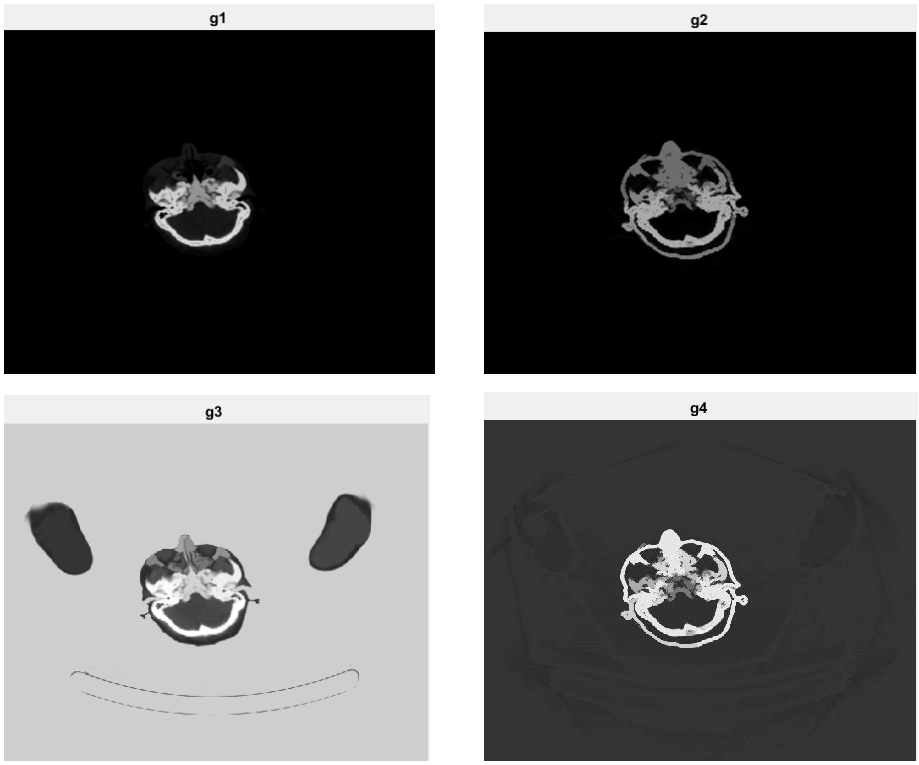


Figure 3.13 Affinity function results obtained by segmenting the image in *Figure 3.12*. Each image is labelled with the affinity function used to obtain it.

In the first image of theFigure 3.13, the algorithm, which started in the back of the skull and moved along to the front, did not segment the front part of the bone. The two appear to be too distant from the rest of the skull to be considered as part of it in this image. These parts are probably connected in another slice of the same volume, but in this particular one it seems to be two separate bones. In the second image, the result of the g2 affinity function, the algorithm behaved like it did in the lungs, segmenting both the skull and the limits between the inside and the outside

of the body. This happens because of the big difference in the intensity values that occur in the limit of the inside and outside of the head.

In the third image of the same figure, the result of the g3 function, the algorithm had a similar solution to the first affinity function. It also behaved similarly to the scenario presented in 3.4.1.2, as the skull is not surrounded by a constant tissue. It has a similar solution to the affinity function g1, however, the background has a much higher affinity that what is ideal. In the fourth image, the result of the g4 function, much like in the second image, the algorithm segmenting both the bones and the limits of the head has having high affinity. On Table 6 the time each affinity function took is displayed.

Affinity function	G1	G2	G3	G4
Time(s)	15.7	5.4	4.1	2.6

Table 6 How much time it took for the algorithm to run each of the affinity functions to complete the calculations in the scenario presented in 3.4.1.3.

3.4.1.4 Result Comparison

In this section, a comparison of the results obtained will be presented. In Table 7, it is presented the times each one of the affinity functions took to complete the segmentation of the image in each one of the scenarios presented above.

Affinity function	G1	G2	G3	G4
Scenario 1	17.3	7.5	4.5	3.8
Scenario 2	18.2	13.1	5.4	7.2
Scenario 3	15.7	5.4	4.1	2.6

Table 7 Comparison between the time taken to run the segmentation in the different affinity functions in each scenario.

As it is possible to observe, the first affinity function requires 2 to 5 times to complete than the other functions. However, that first affinity function is the only one that has constant values along all the scenarios considered. The g2 as well as g4 functions presented bad results in some of the scenarios and the g3 functions requires some adaptation in the different scenarios, as such the g1 function was the one selected for the 3D volume segmentation.

Affinity function (mean, standard deviation)	G1	G2	G3	G4
Scenario 1	1700, 300	400, 200	1050, 50	50, 50
Scenario 2	1700, 300	400, 200	900, 300	40, 120
Scenario 3	1700, 300	300, 100	900, 300	40, 120

Table 8 Comparison between mean and standard deviation of the affinity functions in each scenario.

3.4.2 Testing the Window and Multiresolution Processes

In this section, the experiments and results using the window and multiresolution processes, that were already covered in section 3.3, will be presented. When considering the segmentation problem in three dimensions, it is needed to take into account the size of the volume. The objective of this project is to segment the bone where the seed point was placed, as such, it is not necessary to run the algorithm in the entire volume to have a single bone as its result. It can be chosen be any bone of the ones presented in the CT image, however, some bones may not be successfully segmented. Running the algorithm in the whole volume, would take a considerable amount of time, because the increase of segmentation time is not linear with the increase of slices. This is due to the fact that when one of the values in the connectedness map is updated, it may cause many other voxels to be analysed again. The 3D models were constructed using the algorithm [25].

The initial segmentation is performed on a lower resolution image and when an image is reduced the finer details, such as the limits of the bones, may become blurry or even disappear. The algorithm may then consider that two bones that are close to one another as just one bigger bone, giving the voxels from both of them high affinity, this effect has been seen before in the g2 and g4 affinity functions shown above. This effect of the spread of the high affinity functions, causes a series of false positives and has been reported in other works and designated as leakage [23]. This effect has another implication, the bones are very close to one another, which can make the algorithm consider them as having high connectedness to the seed. This fact is another factor considered when limiting the adjacency to the immediate neighbours only. The result, when the leakage effect occurs, is not be a single segmented bone, but many more grouped as if they were one and even possibly the whole skeleton. In case this happens, the multiresolution process might actually cause worse results than the original ones and render the window process much less effective.

Figure 3.14 Spread effect on a segmentation of a lower resolution image, originated in the shoulder blade.

This continuous "spread" of high affinity values is not the desired result, as such, it was considered a method to prevent it as well as lowering the segmentation time. The limitation of the slices that will be considered in the segmentation, is the proposed method. By reducing the number of slices, there are less slices for the algorithm to analyse which reduce the time and there are fewer structures for the algorithm to leak to. However, this method brings some problems as well, because it limits the number of slices that is processed at a time, which means that the bone may not be completely incorporated in the slices selected for the process. The solution relies on a balance of the number of slices: enough to segment any bone but not too many that it will cause the problems mentioned above. It was chosen 40 slices as a good balance for the segmentations.

3.4.2.1 3D Segmentation methods

In this section, the results of the normal 3D segmentation and the one where the window and multiresolution methods are used will be compared. The experiments will be run in the slices 72 to 82 from the same test volume as the one used in the other sections of this chapter. It was used the affinity function g1 with mean value of 1700 and standard variation of 300.

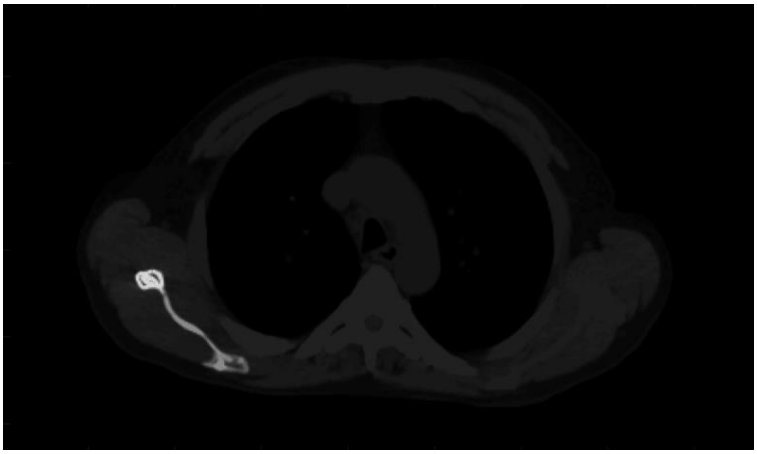


Figure 3.15 Slice of the 3D segmentation using the simple fuzzy connectedness algorithm.

The normal segmentation took 1150 seconds and the results are displayed in Figure 3.15. Like expected, the slice of the image was completely analysed which made the process take a considerable amount of time.

Figure 3.16 Slice of the 3D segmentation using the simple fuzzy connectedness algorithm with multiresolution.

The next test segmentation uses the multiresolution method and the results are displayed inFigure 3.16. The segmentation took a total of 807 seconds.

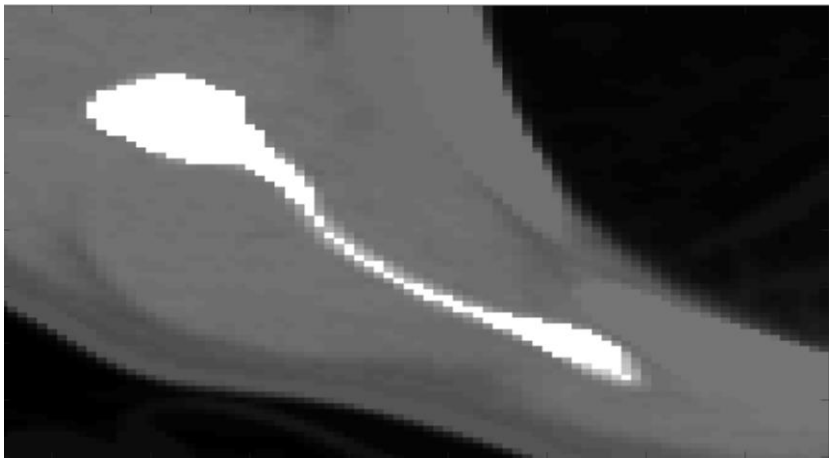


Figure 3.17 Slice of the 3D segmentation using the simple fuzzy connectedness algorithm with multiresolution and window processes.

The last segmentation tested uses both the multiresolution and the window methods. The results are shown in Figure 3.17 and it took a total of 12 seconds. In Table 9, the results are compared and it is possible to observe that the window and multiresolution processes are much faster. It was expected for the combination of the two methods to be faster than the normal segmentation. However, the multiresolution method by itself, needs to segment 3 progressively bigger image volumes, revealing that by having the region of interest partially segmented actually contributes for the reduction of the segmentation time.

	Normal	Multiresolution	Multiresolution and window
Time(seconds)	1150	807	12

Table 9 time taken by the different 3D segmentation processes.

The fuzzy connectedness method is heavily reliable on thresholds because of the way it works. The end result of each of the segmentations is a map of the connectedness of each voxel to the seed, as such, only the ones that are above a certain value are relevant to the segmentation. There are three important thresholds: the first one, that was called the segmentation threshold, is responsible for choosing which one of the voxels can be considered as bone and will be the seeds for the next segmentation; the second, that was called the identification threshold, is responsible for the limits of the window and the last

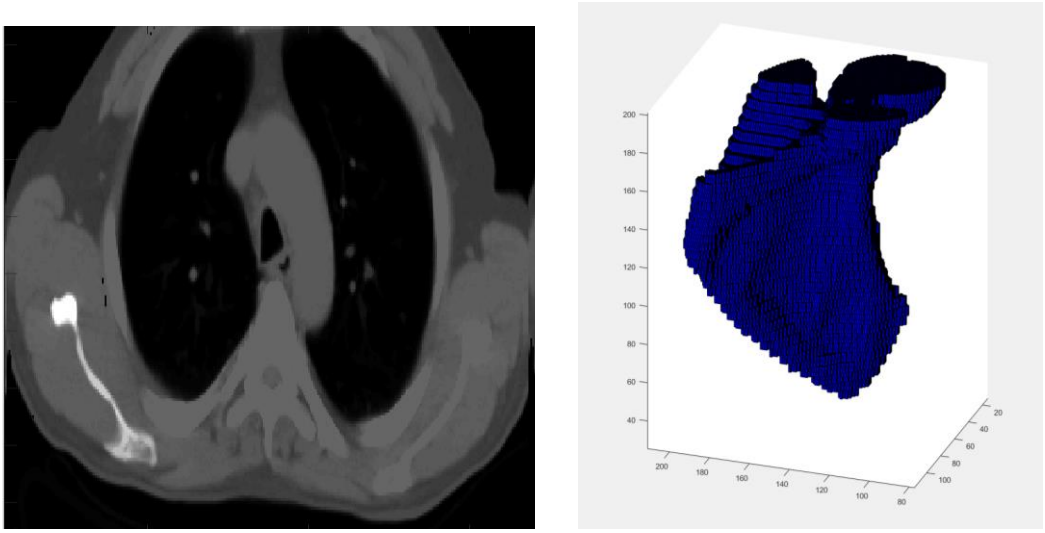
one, called the final threshold, is responsible for selecting what the voxels are considered to be part of the bone and will be utilized to create a 3D model of it. The next three sections will cover the selection of three thresholds, individually, in more detail with experiments and their results. The experiments used the same test CT image volume as the other sections, however it considered 40 images instead of just 10 to segment the shoulder blade.

3.4.2.2 Identifying threshold selection

In section 3.3, the creation of the window was explained, after running the first segmentation a threshold is applied to the image to determine which ones should be considered to create the window. The identifying threshold is the one used on that image, it may have a lower value than the segmentation threshold to make sure a bigger portion of the image is considered. However, it cannot have a higher value than the segmentation threshold as it may not consider portion of the image that was segmented as being bone.

There are some aspects to keep in mind about the value to attribute to this threshold. If its value is too low, it will include too much of the scene, making the method slow, as it needs to calculate many voxels that are not relevant for the final segmentation. In case it is too high, it will result in a final segmentation that is incomplete, because the window will be too small and a portion of the images that contains important information will not be considered.

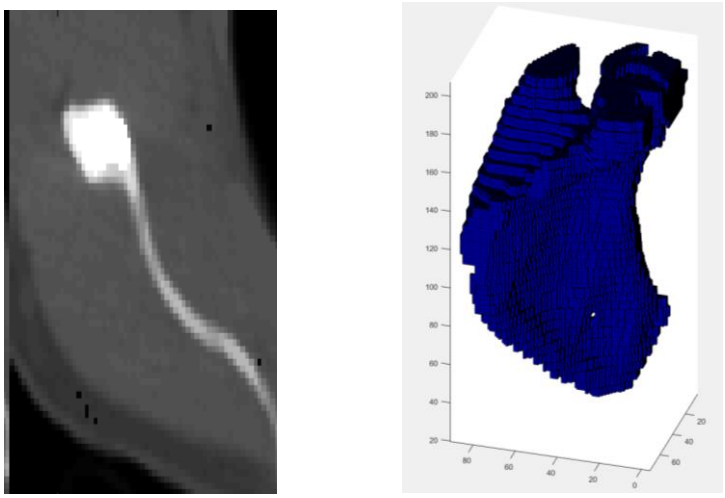
It is better to have a value that is too low rather than one that is too high, because in the first scenario, more of the scene will be incorporated in the window, and on the second, the segmentation will be incomplete. As such, with a lower value the final segmentation will still be obtained, only taking more time to do so. The figures 3.16 and 3.17 show the results of tests performed using a low and a high value for the identifying threshold respectively.



(a) (b)

Figure 3.18 Results of the segmentation using low affinity. (a) 2D slice. (b) 3D model of the segmented bone.

It is possible to observe in the slice of the final segmentation image, shown inFigure 3.18, that the window created is rather large and contains almost the entirety of the image. Resulting in a rather large 3D volume, even though it was reduced and requires a lot of computation time to process. The resulting 3D model it is still good, however it took 980 seconds instead of the normal 120 seconds.



(a) (b)

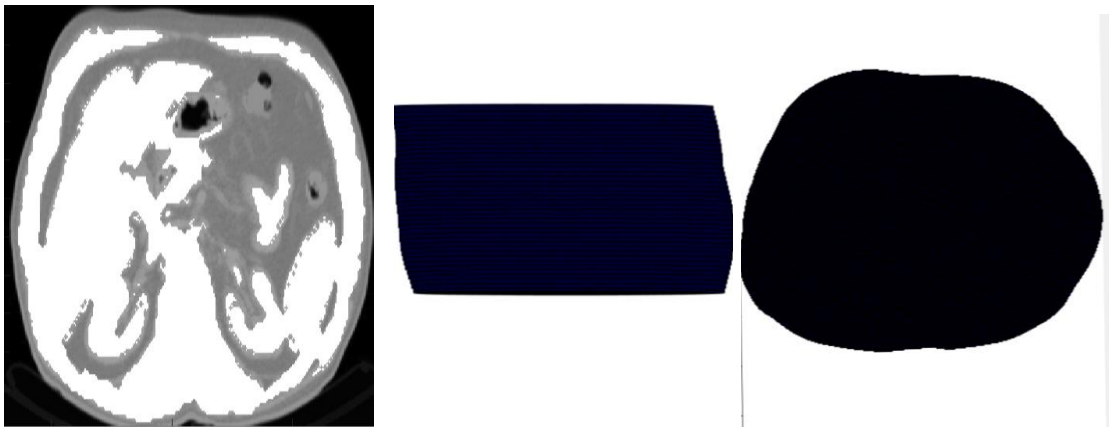
Figure 3.19 Results of the segmentation using high affinity. (a) 2D slice. (b) 3D model of the segmented bone.

It is possible to observe in Figure 3.19 that the window is much smaller and part of the bone was cut short. The resulting 3D segmentation is incomplete, because part of the bone was not considered in the final segmentation. The segmentation

was much faster than the previous tests taking only 84 seconds. After some testing in different scenarios, it was considered that the identifying threshold should be between 0.5 and the value chosen for the segmentation threshold. value chosen for the threshold was 0.6.

3.4.2.3 Segmentation threshold selection

The segmentation threshold, is also used at the same time as the previous threshold, on the same image, but for a different end. This threshold is responsible for defining which of the voxels are part of the bone and, as such, can be considered as seeds for the next segmentation. The value of said threshold cannot be too low, as it will consider elements that are not bone as being so. Since most of these points are not evaluated later, it is crucial that this value is not too high. In case too many points are wrongly segmented as bone, it will originate in many problems later, as the segmentation will be incorrect. The segmentation results when using a threshold that is too low are displayed in the Figure 3.20 .



(a) (b) (c)

Figure 3.20 Results of the segmentation using low segmentation threshold. (a) 2D slice. (b) 3D model of the segmented bone front view. (c) Same model view from above.

When this threshold is too low, the algorithm wrongly considers that many others structures belong to the bone originating in segmentations that can include the whole body, like in the example presented. In case it's too high, it won't save much time, as the seeds for the next segmentation in the intermediate and then in the

original images will be very similar to the first one. In the case it is too high, it will stop the process short, as there will be no segmented voxels as bone and there will be no seed points for the next segmentation to initiate from. Taking into consideration the properties presented above, a higher segmentation threshold is preferred as it may take a few more seconds to run but will still present good results. The seed value will always have value 1, however the boundaries are removed in each segmentation. As such, it was considered that segmentation threshold value should be between 0.5 and 0.9. The value chosen for this segmentation was 0.7

3.4.2.4 Final threshold selection

The final threshold is responsible for the selection of the voxels that are considered to be part of the bone in the final segmentation made on the image with the original resolution. The voxels selected here will be the ones that are considered for the 3D model. In this case, if the threshold value is too low, the bone will be incomplete presenting artefacts such as holes or thinner than normal bones, as it is possible to observe in Figure 3.21. In case the value is too high, the final result will have more voxels than it should have, considering other structures as bones, as it is shown in Figure 3.22 Figure 3.21.

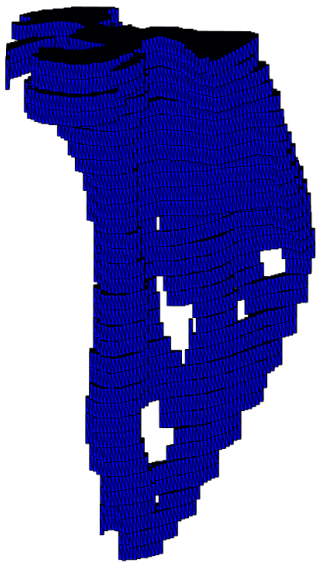


Figure 3.21 Results of the segmentation using a low final segmentation threshold.

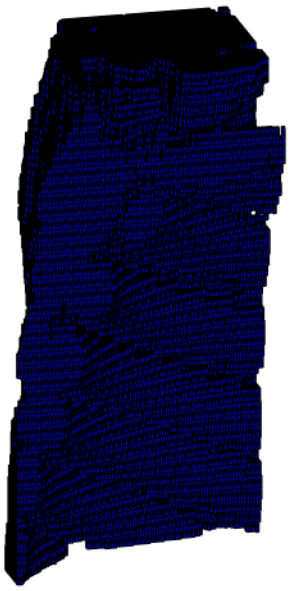


Figure 3.22 Results of the segmentation using a high segmentation threshold.

As it is possible to observe, in Figure 3.21, the bone has several holes on it, because those areas have a lower affinity value than the rest of the bone and are not considered by the threshold. In Figure 3.22, the 3D model presents structures that are not relevant, the ones displayed there, that are not present in the other 3D models, correspond to the ribs and the muscle that involves them. Some experiments were run using the volumes in appendix A and the values chosen for this threshold are between 0.3 and 0.7. The value chosen for the final threshold is 0.45.

Chapter 4. Iterative Relative Fuzzy Connectedness

In this section, the iterative relative fuzzy connectedness method, also developed by Udupa *et. Al* [20], will be presented and explained. It will be first explained and then the algorithm will be presented, as it is easier to understand the algorithm this way. In the last section of this chapter, it will be presented the experiments and results done using this method, as well as a proposed modification.

4.1 IRFC

The iterative relative fuzzy connectedness method uses the fuzzy connectedness method as its foundation, which was exposed and explained in the previous chapter in detail. The original method builds a connectedness map of each spel to the seed and presents it to the user to interpret. In contrast, this method is relative, and as the name suggests, it requires more than one seed as input to function, it uses the two, or more, seeds to create separate connectedness maps and compare them. The result of this process is an image with different regions, each one associated to the corresponding seed.

To do this comparison, it is run the fuzzy connectedness method for each seed, having that seed as the starting point. For each seed, it is necessary to know the type of tissue it is, or at least its properties, because each tissue has a different affinity function. After obtaining all the connectedness maps, they are compared between each other to build a new map. In this new map, with the same dimensions as the original image, each spel will be labelled according to the different connectedness maps. To each spel, it will be attributed the value of the seed of the connectedness map that has the highest affinity value for that spel. The final map presented is divided into different regions, each of which associated with one seed.

In the regions that are hard to segment, like blurry areas and regions that are hard to separate from one another, the relative fuzzy connectedness map may present some artefacts and mislabelled spels. To prevent this, an interactive component was added to it. After the initial segmentation, it is verified which spells have a higher affinity to the object than to the background and those spells are saved and eliminated from the next segmentation. This segmentation is performed using a different seed of the background and the corresponding affinity function.

This new segmentation is performed to verify if there are any spels that can be segmented as part of the object. In case there is a difference in the two relative maps, the spels that have higher affinity to the object in this new segmentation are saved again and eliminated from the next segmentation. This process is repeated until there are no further changes in the relative map, using a different background seed in each segmentation, making sure all the spels are correctly segmented and the final segmentation is presented.

4.1.1 Meta-code

In this section the meta-code is presented. The *FC* is the object affinity map and FC_b is the background affinity map. Each time

Input: original image C, seed map S with the seeds from all the classes, connectivity scene K,
Output: For each s in *S,* iteratively defined fuzzy object containing s and relative to the background *W.*

Begin

1. *For* each different seed value *s* in *S do*:
    a. Compute *FC* by running the fuzzy connectedness algorithm using the seed *s;*
    b. Create a new image *I* with the size of *C;*

c. Set *verify* = true; set K_s^i=*K;*

d. Separate the rest of the seeds into *W*;

e. While *verify* = true

i. Set *verify* = false;

ii. Compute FC_b by running the fuzzy connectedness algorithm using the *W* as seed and K_s^i as the scene;

iii. *Set* K_s^i=*K;*

iv. *For* all $c \in C$*:*

1. If *I(c)*=0 and *FC*>FC_b

a. Set *I(c)* = 1;

b. Set *verify* = false;

c. *for* all $d \in C, d \neq c$*:*

i. *set* $\mu_s^i(c,d) = 0$*;*

d. *end for;*

2. $end\ if$;

v. $end\ for$;

vi. $increment\ i\ by\ 1$

f. $end\ while$;

2. Output *I*

End

4.2 Experiments and Results

In this section, the experiments made with the method described above will be presented. It was used the same computer, as it facilitates the comparison of speed and performance with the other method. The tests were run in two dimensions and it was considered the image used in previous experiments, the slice 79 of the test volume.

4.2.1 Initial Experiments and Results

This method requires more information to function, because it needs more seeds than the fuzzy connectedness method. As such, experiments were run with increasingly amounts of seeds to determine if this method can be also included in the algorithm in a way that it can used by most users.

On the first experiment, it was simply used two seeds, one placed in the shoulder blade and the other in the background, as it is shown Figure 4.1,image (a). It does not have any sort of pre or post processing and it resulted in the image (b) shown in Figure 4.1. The segmentation took 372 seconds. As it is possible to observe, there are regions of the image (b) that are black, that means they were not segmented. No real structure is possible to be observed in the result, as such other tests were performed.

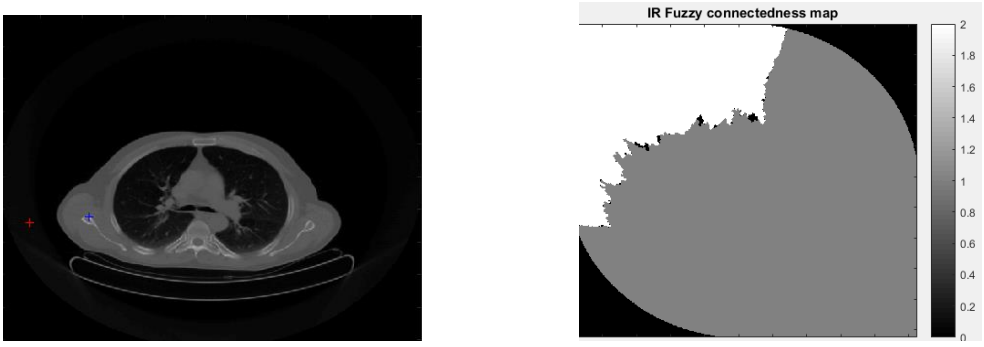
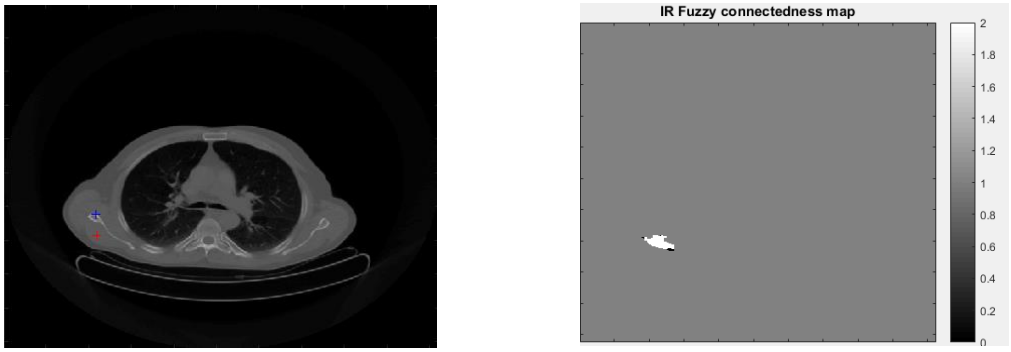


Figure 4.1 IRFC first segmentation test using two seeds. (a) The original image with the seeds. (b) The result of the segmentation.

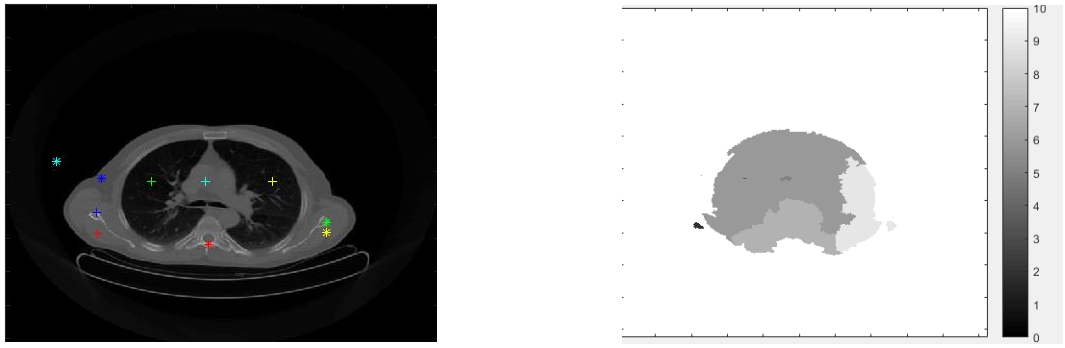
Since the first experiment did not show any results, the seed from the background was moved to the muscle surrounding the shoulder blade, as it is displayed in Figure 4.2, image (a). The segmentation took 273 seconds. It was not applied any sort of pre or post processing and the result is displayed in Figure 4.2 image (b). In this experiment, the amount of unsegmented pixels was diminished, however there is still no discernible structures in the segmented image.



(a) (b)

Figure 4.2 IRFC second segmentation test using two seeds. (a) The original image with the seeds. (b) The result of the segmentation.

Considering the nature of this method, it requires at least a seed for each structure, as such, it is not surprising the lack or results displayed so far. It was considered a test where multiple structures were given seeds, this would take the user some more time to do, but it would still be feasible. The seeds chosen are displayed in Figure 4.3, image (a). Each seed is in a different structure and each one have a different symbol and colour combination. The segmentation took 1745 seconds to complete and its results are displayed in Figure 4.3, image (b). It is possible to observe that there are areas surrounding some of the structures, however it is still not possible to discern any of the structures.

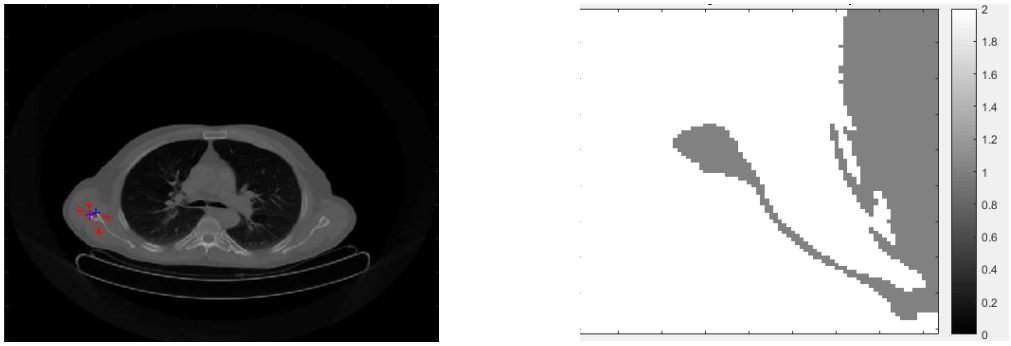


(a) (b)

Figure 4.3 IRFC segmentation test using ten seeds. (a) The original image with the seeds. (b) The result of the segmentation.

4.2.2 Tests Performed Using a Window

Considering the amount of seeds already chosen and the lack of results, the further increase of the seeds, that need to be carefully selected by a user, does not seem practical. Besides the increase of time is considerable from 273 with two seeds to 1745 seconds with ten seeds. As such, it was made an experiment where the window, obtained during the tests of section 3.4.2, was considered. By doing this, the amount of structures needed to be seeded and segmented is greatly reduced. It was chosen a seed for the shoulder blade and another for the muscle surrounding it, as it is shown in Figure 4.4, image (a). The results are displayed in image (b) of the same figure. As it is possible to see there is now possible to discern the structure of the shoulder blade clearly, however, the same effect of "spread" still occurred in this method.

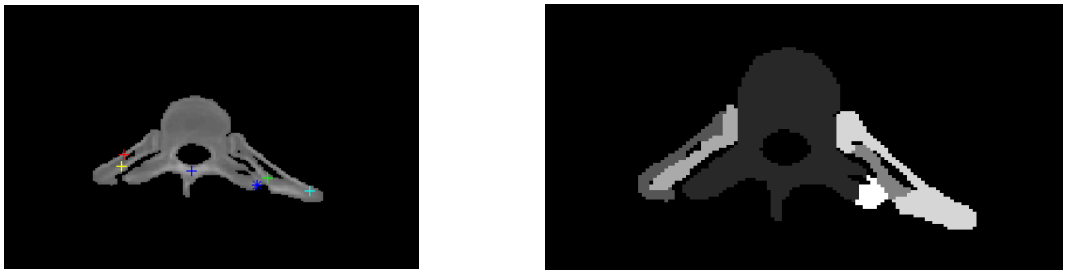


(a) (b)

Figure 4.4 IRFC segmentation test using ten seeds. (a) The original image with the seeds. (b) The result of the segmentation.

In practical terms, to be able to choose the seeds with the window in the fashion that was described above, it is necessary to run the method described in section 3 first. As such, it was considered another scenario, where there is an initial segmentation using the other method as a base, and then the method described in this chapter is used. Doing this experiment in the shoulder blade, would not show any new results, as such, it was considered the vertebrae and the ribs in the same image instead. This area was already described as having troubles with the "spread" effect, as such, it was run an experiment with multiple seeds in the vertebra and ribs, as it is displayed in Figure 4.5, to separate the structures from

one another. The results are shown in the same figure and it took 48 seconds to complete the segmentation of all the areas.



(a) (b)

Figure 4.5 IRFC segmentation test on an already segmented image using six seeds. (a) The original image with the seeds. (b) The result of the segmentation.

It is possible to observe, in Figure 4.5, that the vertebra was successfully separated from the ribs, however it is separated in two regions, one and six, with one of the ribs being formed from the region three and four and the other one with the region two and five. By grouping those regions, it is possible to segment each bone separately. By joining the correct areas together, it is possible to separate correctly the three different bones, as it is shown in Figure 4.6. In the picture b) the vertebra is formed by joining the regions labelled as 1 and 6. In the picture c) the left rib is formed by joining the regions labelled as 2 and 4. In the picture d) the right rib is formed by joining the areas labelled as 3 and 5.

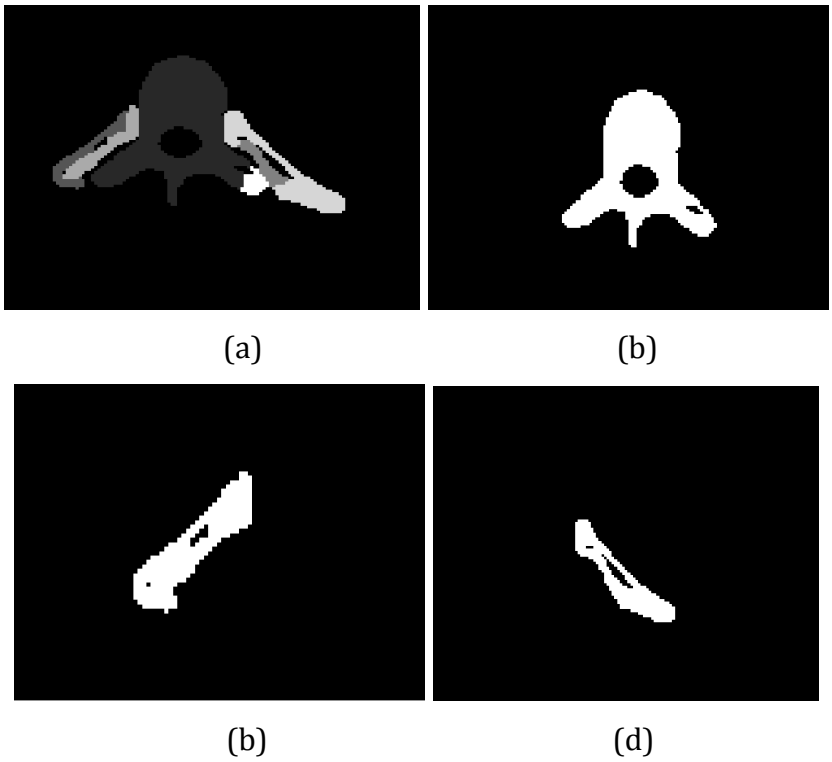


Figure 4.6 IRFC segmentation and the separation of each one of the structures. (a) Segmented image. (b) Separated vertebra. (c) Separated right rib. (d) Separated left rib.

4.2.3 3D Experiments and Results

When converting the algorithm to a three dimensional setting, there are some aspects to take into consideration. The changes in affinity and adjacency are already covered by the fuzzy connectedness method itself. However, it needs more seeds to function, as well as good positioning of said seeds. It will also take a considerable amount of time more, as it needs to run the method explained in the previous chapter several times to reach the final result.

To allow the method to separate the bone from the background, the muscle and even other bones, each of these structures needs to have a seed in them and it even may be required more than one seed to make this separation possible. In case there is a region that does not contain a seed, for example a muscle surrounding a bone, it may be incorporated into the bone, resulting in a region much bigger than it should be. As such, every structure must have at least one good seed, or it may

create a final image where the regions are bigger than the structures the seeds originated in.

As the algorithm progresses along the persons height, the position in the body changes, as such the structures that are present in each slice change along the volume. Because of this, the number of seeds required is greatly increased as each one of the structures requires at least one seed, contrary to the original method that only needed one. The more slices the algorithm considers the more seeds it needs to function, as it will consider all the structures involved in the segmentation to reach the final segmentation.

The computation time is also greatly increased, as the original fuzzy connectedness algorithm needs to be run a number times along all the images considered. In order to obtain the final fuzzy connectedness map with good results, each seed will require at least two initial segmentations using the complete volume and then a few more iterations with a portion of the volume considered.

Considering the properties mentioned above, it is not practical to use this algorithm in the same way as the previous one was used. It can, however be used to separate the different bones from an already segmented volume in the same fashion it was done in two dimensions. As such, it was performed an experiment using a 3D segmented image and then running the iterative relative fuzzy connectedness method. In Figure 4.7 it is possible to see a portion of the original image with the seeds present in the image in the middle. The results of the segmentation are shown in Figure 4.8. The vertebrae and ribs were divided into different regions, much like the previous experiment, and need to be group together to form the objects in 3D.

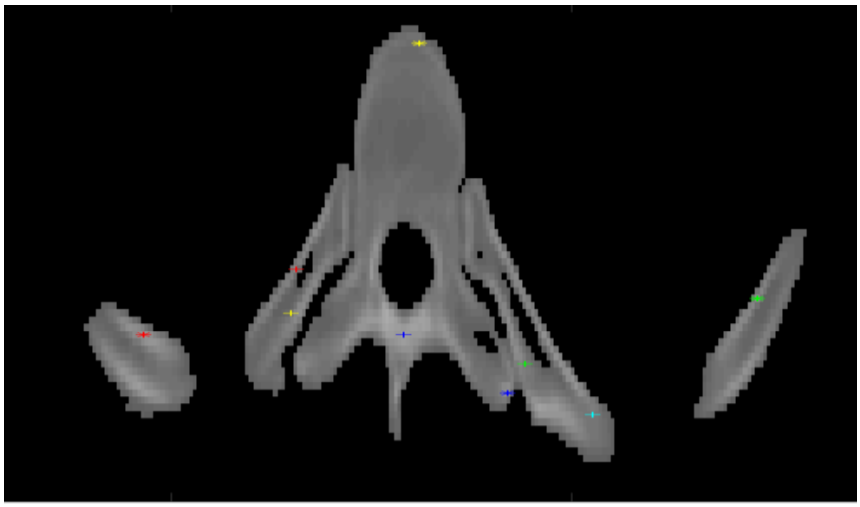


Figure 4.7 One of the slices of the already segmented image. The seeds that will be used in the IRFC segmentation are represented by the points with the different colours.

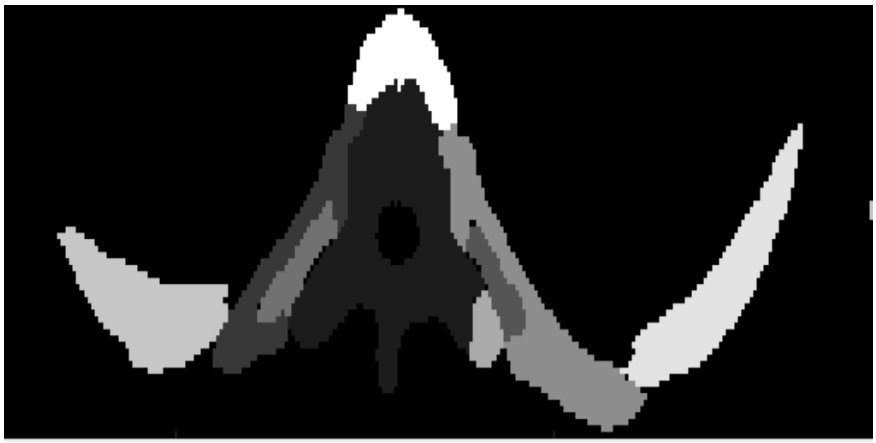


Figure 4.8 Results of the IRFC segmentation of *Figure 4.7*.

The areas segmented, as displayed in Figure 4.8, were then divided, like in the two dimensional segmentation to form the 3D models shown in Figure 4.9. The different images in that figure were obtained by joining the appropriate regions. To form Image (a) the regions with values 2 and 4 were joined; image (b), regions 1 6 and 9; image (c) regions 3 and 5; image(d) is simply region 7 and image (e) is region 8. In Figure 4.10, the complete 3D model is presented, however it is used another program to visualize it called *Blender*. No modifications were made to the 3D model, however, it chosen a program that allowed the better visualization than the model created by *Matlab.*

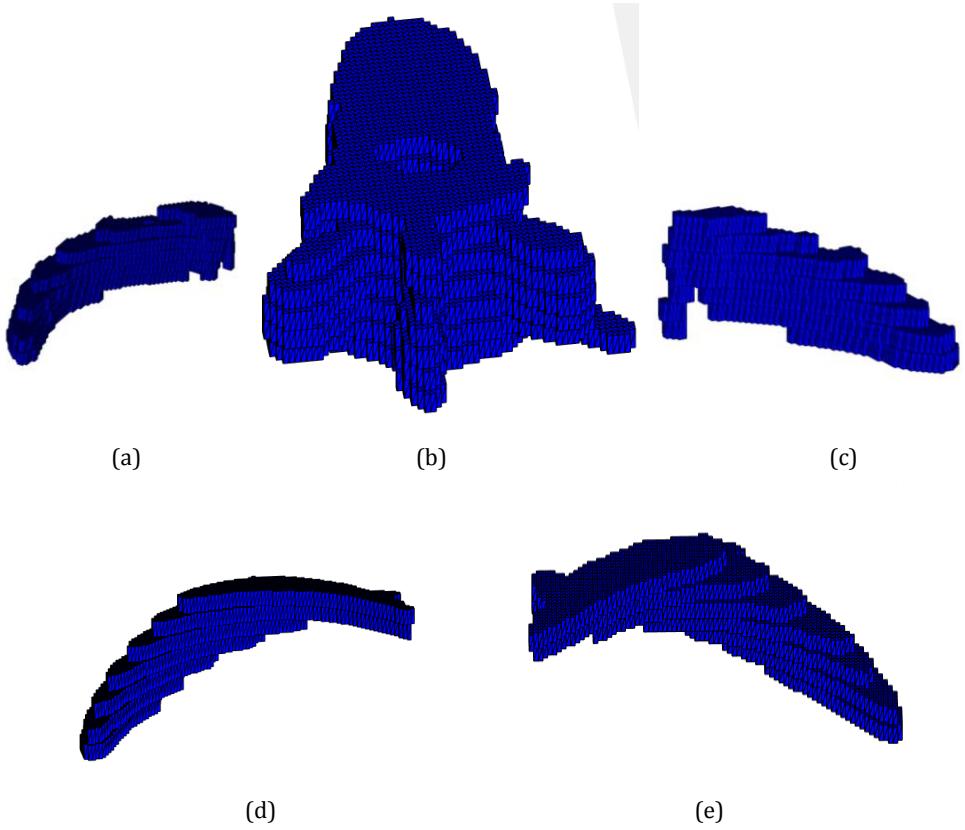


Figure 4.9 Separated 3D models of the regions of *Figure 4.8*. (a) right lower rib. (b) vertebrae. (c) left lower rib. (d) right upper rib. (e) left upper rib.

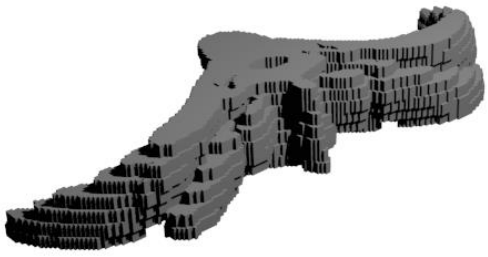


Figure 4.10 Complete 3D model of the *Figure 4.8*. The open-source program *Blender* was used simply to visualize the model without modifying it.

Chapter 5. Scale-based Affinity

This method was already mentioned in the State-of-the-art and as it explained, it uses the standard fuzzy connectedness as it's algorithm. However, it has an affinity function that is more complex and it will be explained in this chapter with the experiments and results presented in the end.

5.1 Algorithm

This method utilizes the same method of the simple fuzzy connectedness one, however, the affinity function is much more complex than the one presented before.

5.1.1 Adjacency

Adjacency is actually simpler in this method, as the distance to the seed point is not relevant for the calculation of the affinity, it is displayed in the formula 5.1.

$$\mu_\omega(c,d) = \begin{cases} 1, if \ \sum_{i=1}^{n} |c_i - d_i| \leq n \\ 0, otherwise \end{cases} \quad (5.1)$$

5.1.2 Affinity

This affinity is different than the others, as it only takes into consideration the relation between two spels that are directly adjacent to each other instead of the relations between each spel and the seed. That is the reason why the distance to the original seed is not important. However, the original seed is utilized to do a quick threshold segmentation to select the region around it, in order to obtain certain variables, that will be used in the calculations. Udupa *et al.* [21] selected the area manually in an attempt to choose the right spels for the segmentation. However, bone can be roughly segmented, even if with some flaws, through the

threshold method, as such, the region obtained from such a method, can be utilized to obtain these values that will be used later.

When calculating the affinity between spels, there are many possible affinity functions that can be used. Each one of these functions can be used in a specific situation, as they cover distinct proprieties of the image. However, in this scenario the affinity function that usually present better results are the g3 and g4 [21]. As such that one will be the one used. The possible functions are shown below in the formulas 5.2 to 5.6.

$$(\text{g1})\ \mu_k = \mu_\alpha \mu_\varphi \tag{5.2}$$

$$(\text{g2})\ \mu_k = \mu_\alpha \mu_\emptyset \tag{5.3}$$

$$(\text{g3})\ \mu_k = \mu_\alpha \sqrt{\mu_\emptyset \mu_\varphi} \tag{5.4}$$

$$(\text{g4})\ \mu_k = \mu_\alpha .1/2(\mu_\emptyset + \mu_\varphi) \tag{5.5}$$

$$(\text{g5})\ \mu_k = \mu_\alpha \mu_\emptyset^{\gamma} \mu_\varphi^{1-\gamma} \tag{5.6}$$

The homogeneity-based component φ, measures the degree of hanging togetherness of the spels because of the similarity of a specified object feature. The object-feature-based component φ measures the degree of local hanging togetherness of spels because of their similarities of intensities. These two components will be described in the next two sections.

5.2.2.1 Homogeneity-based component

The homogeneity-based component measures the homogeneity of the intensity of the spels within the object. It can be described using the formula 5.7. and the components are also explained in this section.

$$\mu_\varphi(c,d) = \frac{|D^+(c,d) - D^-(c,d)|}{\sum_{e \in B_{cd}(c)} \omega_{cd} \|c-e\|} \tag{5.7}$$

ω_{cd} *is a homogenity function with parameters that depend on* $r(c)\ r(d)$

The ball function $B_{cd}(c)$ is shown in formula 5.8; the $D^+(c,d)$ and $D^-(c,d)$ functions are in formulas 5.11 and 5.12; the $\delta_{cd}^+(e,e')$ and $\delta_{cd}^-(e,e')$ components are displayed in functions 5.9 and 5.10. The homogeneity functions W_Ψ and ω_{cd} are described in formulas 5.13 and 5.14 respectively.

$$B_{xy}(z) = \{e \in C \mid \|z - e\| \leq \min(r(x), r(y))\} \tag{5.8}$$

$$\delta_{cd}^+(e,e') = \begin{cases} f(e) - f('e), & if\ f(e) - f('e) > 0 \\ 0, & otherwise \end{cases} \tag{5.9}$$

$$\delta_{cd}^-(e,e') = \begin{cases} f(e) - f('e), & if\ f(e) - f('e) < 0 \\ 0, & otherwise \end{cases} \tag{5.10}$$

$$D^+(c,d) = \sum_{\substack{e' \in B_{cd}(d) \\ c-e=d-e'}} [1 - W_\Psi(\delta_{cd}^+(e,e'))\omega_{cd}(\|c-e\|] \tag{5.11}$$

$$D^-(c,d) = \sum_{\substack{e' \in B_{cd}(d) \\ c-e=d-e'}} [1 - W_\Psi(\delta_{cd}^-(e,e'))\omega_{cd}(\|c-e\|] \tag{5.12}$$

The intensity variations, from the spels to the mean, inside of an object tend to be fairly random and can be described with the standard variation model. As such, if the variations from the mean are all summed together, the total will be approximately 0 [21]. The variations from between objects tend to be much higher, as such, depending on the positions of c and d, the variation between them will either be increasing or decreasing, while the variations inside the object, when summed will have little to no variation.

That being said, it can be assumed that the smaller values of $D^+(c,d)$ and $D^-(c,d)$ can be interpreted as the components inside the object and the larger parts represent the variations between objects. Considering that the neighbourhood is small, the component created by its variation is unlikely to have a variation comparable to the inter-object variation which led to the creation of μ_φ.

The homogeneity functions mentioned above can have different forms and parameters, but the ones used here are the ones described in the formulas 5.13 and 5.14. Where $a_{\Psi 1}$, $a_{\Psi 2}$, a_{10}, a_{20} and m_0 are obtained using the initial threshold segmentation, but that will be explained in more detail later.

$$W_{\Psi 2} = \begin{cases} 1, 0 \leq x \leq a_{\Psi 1} \\ \frac{a_{\Psi 2} - x}{a_{\Psi 2} - a_{\Psi 1}}, \ a_{\Psi 1} \leq x \leq a_{\Psi 2} \\ 0, \ x > a_{\Psi 2} \end{cases} \tag{5.13}$$

$$W_{02} = \begin{cases} 0, \ x < m_0 - a_{20} \\ \frac{x - (m_0 - a_{20})}{a_{20} - a_{10}}, \ m_0 - a_{20} \leq x \leq m_0 - a_{10} \\ 1, \ m_0 - a_{10} \leq x \leq m_0 + a_{10} \\ \frac{(m_0 + a_{20}) - x}{a_{20} - a_{10}}, \ m_0 + a_{10} \leq x \leq m_0 + a_{20} \\ 0, \ m_0 + a_{20} < x \end{cases} \tag{5.14}$$

5.2.2.2 Object-feature-based component

This component also considers the spels of the object, however, it will consider a filtered version that takes into account the ball defined by the formula 5.15.

$$B_r(c) = \{e \in C | \ \|c - e\| \leq r(c)\} \tag{5.15}$$

The filtered version is obtained using the function shown in formula 5.16.

$$f_a(c) = \frac{\sum_{e \in B_r(c)} f(e) \omega_{cd} \|c - e\|}{\sum_{e \in B_r(c)} \omega_{cd} \|c - e\|} \tag{5.16}$$

The object-feature-based component is described in formula 5.17.

$$\mu_{\emptyset}(c, d) = \begin{cases} \frac{\min[W_o(c), W_o(d)]}{\max[W_b(c), W_b(d)] + \min[W_o(c), W_o(d)]}, if \ \min[W_o(c), W_o(d)] \neq 0 \ and \ c \neq d \\ 1, if c = d \\ 0, otherwise \end{cases} \tag{5.17}$$

W_o *and* W_b Are intensity distribuition functions for the object and the background respectively, that have already been described in formula 5.14, where the values of m_0, a_{10} and a_{20} are different for the object and background and were also obtained during the initial segmentation.

5.1.3 Scale-based Component

To calculate the relations between the selected spels and the rest of the picture, a local region, that is selected according to a homogeneity relation to the selected spels, is taken into consideration. That region is called scale and is estimated using the function *fraction of object* FO_k that is shown in the formula 5.18.

$$FO_k(c) = \frac{\sum_{d \in |B_k(c) - B_{k-1}(c)|} W\Psi(|f(c) - f(d)|)}{|B_k(c) - B_{k-1}(c)|} \quad (5.18)$$

$$B_k(c) \textit{ is a hiperbal with radious k with as its origin}$$

In the scale computation algorithm, the ball radius is increased by one unit at a time and the $FO_k(c)$ is checked for the fraction of the object that is contained in the scale.

5.1.4 Algorithm Steps

In this section, all the functions mentioned above will be put together to form the various steps of the algorithm. The flowchart presented in Figure 5.1, illustrates the steps taken to calculate the connectedness map using the scale-based affinity.

As it is possible to observe in Figure 5.1, the first step, after the selection of the seed, is to do the initial segmentation using a thresholding technique. The spells contained in the segmented region are used to calculate m_{0b}, which is the mean of their intensities and the a_{20b}, which is the value of the standard variation of the intensities times the number of standard variations considered. This number is the same throughout the algorithm, in this case it was considered 3 standard

variations and a_{10b}, is 0. The same is done for the spels in the background obtaining the m_{bs} and a_{2bs}, with a_{1bs} being 0 too. The object values will be used in the W_Ψ homogeneity functions, and the background values will only be used in the object-feature-based component.

Still considering the segmented region, it is also measured the difference between the pixel intensities in order to determine the mean of the variations and its standard variation. Those values will be used to calculate m_{psi}, which is the mean just obtained, and a_{2psi}, which is the mean plus the standard variation just obtained times 3, the number of standard variations. a_{1psi} is 0. These values will be used in the w_{cd} homogeneity functions, as constants.

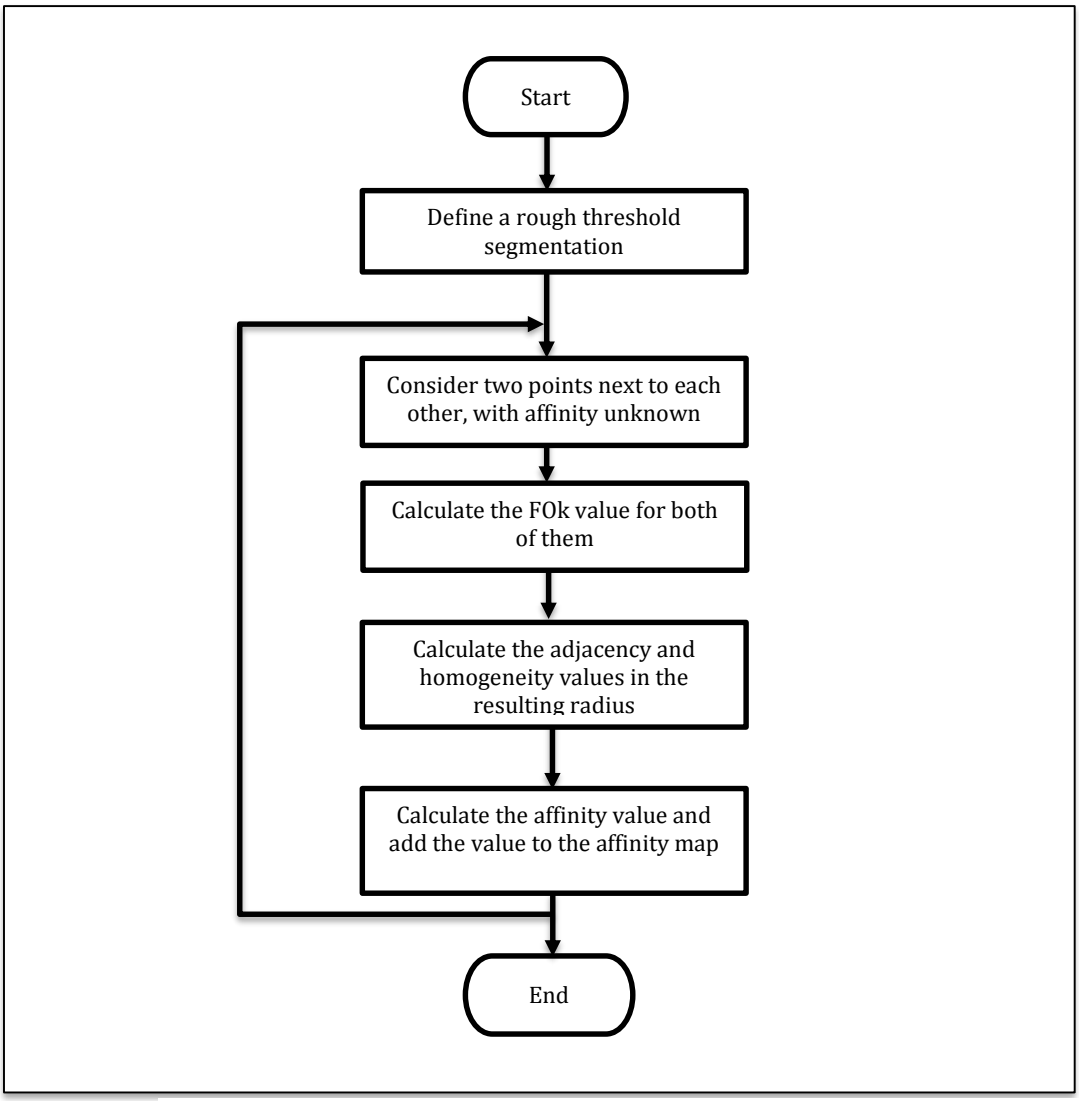


Figure 5.1 Flowchart of the Scale-based fuzzy connectedness method.

The next few steps will be repeated until all the possible neighbour pixel pairs have been covered. First, it is chosen a pair of spels to which the affinity is still unknown, then the *FOk* function is run on those two pixels to determine the radius of the hyper-ball. It is run, constantly adding one unit to its radius until the function has a value that is lower than 0.85. Then the maximum radius is returned and compared between the two pixels to determine which one has the lowest one. That is the radius used in the other functions while determining the affinity between these two spels. Using the values obtained, $D^+(c,d)$ and $D^-(c,d)$ are calculated to determine the homogeneity-based component of the algorithm. In the radius of the hyper-balls of both spels, the filtered values are also calculated, $f_a(c)$ and $f_a(d)$ so that the object-feature-based component can be determined. After obtaining both of the components they are used in formula 5.4 to determine the affinity. This value will then be stored in a connectedness map, with the same size as the original image where all the affinities will be stored and presented when everything is calculated.

5.2 Experiments and Results

In this section the results obtained will be shown. The affinity function chosen, as it was explained above, was the g3 affinity function, provided by formula 5.4, and the g4, displayed in the formula 5.5. The seed was placed in the point (337,244), where the vertebra is located. The results of the experiment took 2354 seconds for the affinity function g4 and the results can be seen in Figure 5.2 image (a). The images (b) and (c) show the individual components of the segmentation, and image (d) shows the results when using the g3 affinity function. The results show that all the bones in the image have a high affinity to the seed, as it was expected. However, this results are not very useful, they do show the bones more distinctly than in the normal image, but they do not make any distinction between the bones close to the seed and the ones that are far away from it. The experiment took almost 4 hours for a single image and the results still need to be processed again to be of any use. As such, it was not made further experiments or attempts to make the code faster, because it simply took too much time to be of any use in the

prototype. It is a method that analyses the interaction pixel by pixel, which might be good in situation where more precision is required, such as medical images taken to study the bones of the foot, but not in a full body CT image like the ones used here.

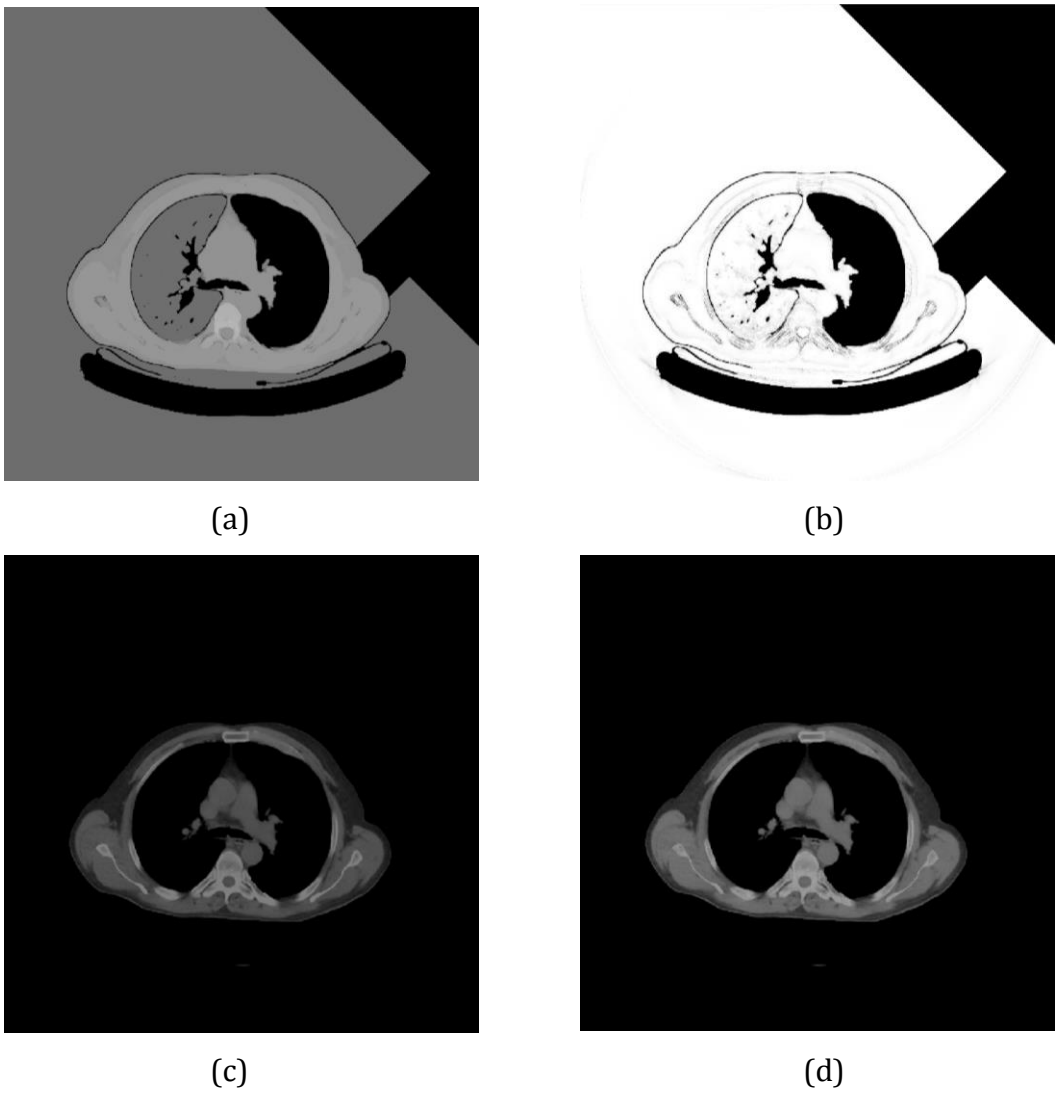


Figure 5.2 Results of the scale-based fuzzy segmentation. (a) connectedness map obtained using the g4 affinity function (b) Homogeneity-based component. (c) Object-feature-based component. (d) connectedness map obtained using the g3 affinity function.

Chapter 6. Prototype

The goal of this work is to implement the algorithms that were mentioned above in a way that a user can select and segment a bone from a CT image volume and obtain a 3D model of said bone. In this section, the prototype developed to this end, will be presented, first the users end and interactions and then what is happening while the prototype is working.

6.1 Users End

All of the techniques used in the prototype have already been presented in the other sections, as well as, the results from the experiments using each and every one of them. In this section, it will be presented what the user sees and what they can interact with.

It was created a GUI that gives the user an easy way of interacting with the program. The first menu that appears, allows the selection of the desired image volume. It allows the user to select a single slice, for 2D segmentation, and a portion or the complete image volume for the 3D segmentation. A radial button was added to allow for the fast selection of the full folder, as it is possible to observe in Figure 6.1, that shows this first menu.

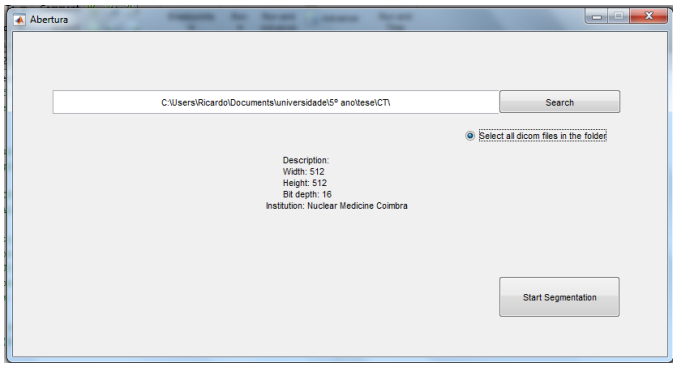


Figure 6.1 GUI first menu.

After the images have been chosen, a new page, that contains two images, appears. As it is possible to see in the Figure 6.2, the image on the left of the page corresponds to the current slice of the volume, the image on the right presents a front view of the volume, it has a white line to indicate the position of the slice that is being seen in the left image. The left image has a scroll bar, that allows the user to scroll along the slices of the dataset and a button below for when the user wants to select the seed point. The scroll bar on the right allows the user to perform a simple thresholding technique, where the bar corresponds to the value of the threshold.

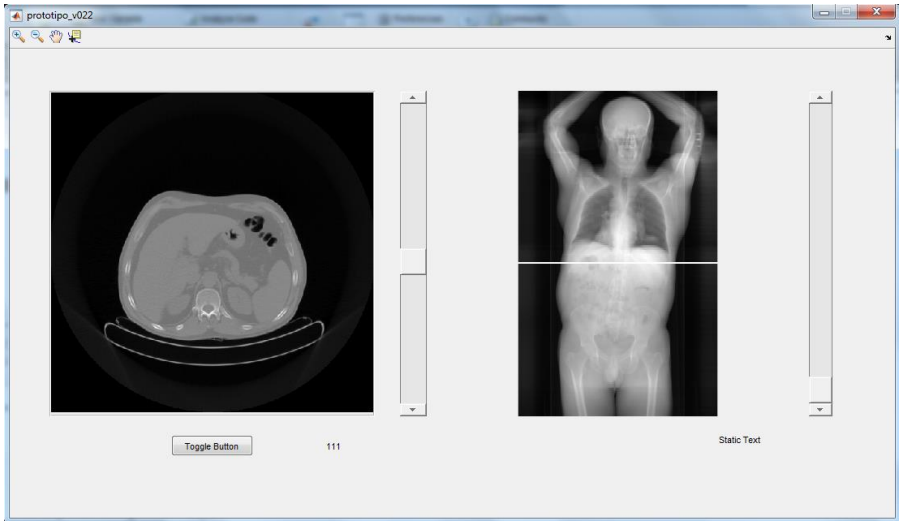


Figure 6.2 GUI, second menu.

After the seed point has been chosen, the segmentation begins, it may take between a few seconds to a few minutes depending on the image selected and the computer being used. When it is complete, a 3D model of the segmented bone is presented and it can be rotated and analysed.

6.2 Techniques Used

Now, that the steps of how the porotype can be used have been explained, it is possible to explain what happens in the prototype itself, in the background and the techniques used that are not exposed to the user. To do so, the algorithm was run selecting the test image volume, as it is described in the Appendix A. The seed was chosen from the slice 79 of that volume, with a total of 40 chosen slices of that same volume.

After the seed has been chosen by the user the multiresolution method, is implemented. First a number of slices is chosen above and below the current slices. All the slices are then subjected to the multiresolution process and window process explained in detail in Chapter 3. This first step of this method is to reduce the slices chosen in every dimension by four as it is shown in Figure 6.3.

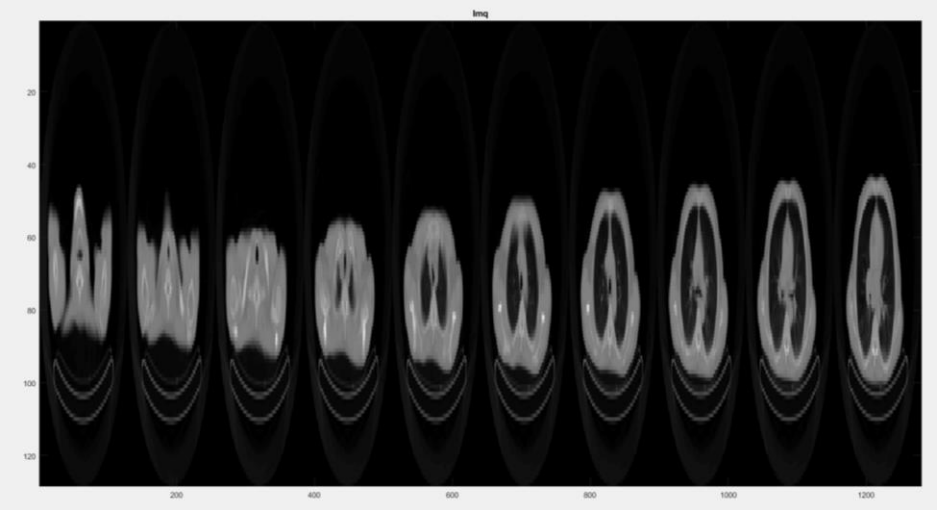


Figure 6.3 Reduced image volume.

After the reduction, it is run the initial segmentation using the fuzzy connectedness method. The results of this process can be seen in Figure 6.4.

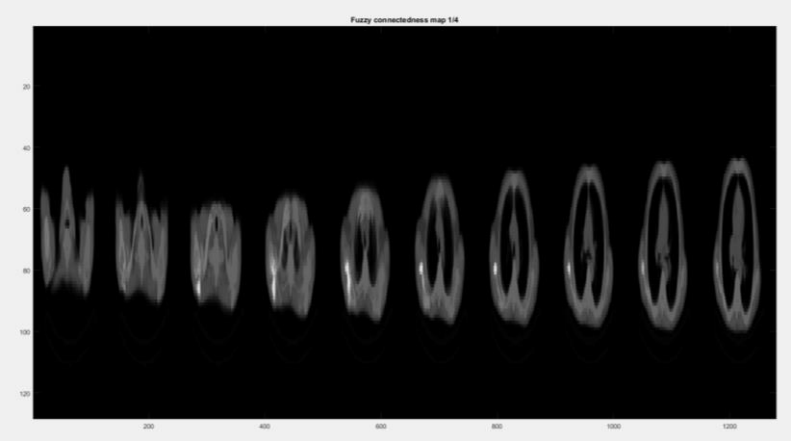


Figure 6.4 Segmented image of the reduced volume.

The next step is the window. This consists in selecting a window around the segmented structure according to the identifying threshold mentioned in chapter 3. In Figure 6.5, it is shown the results of the chosen area.

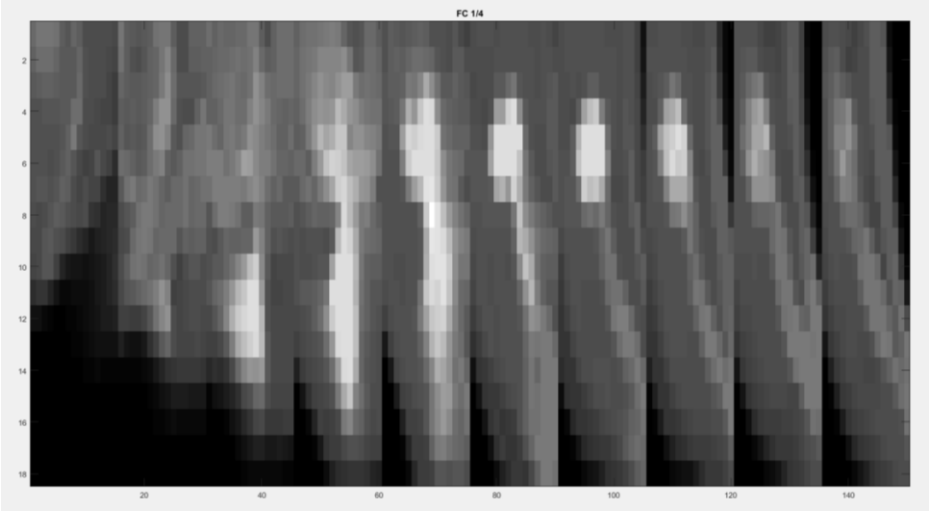


Figure 6.5 Window of the segmentation of the reduced image.

After the size of the window has been determined, it is then applied in the original image and the intermediate image. This originates two images that are only a portion of the original ones and that only contain the bone of interest. The results are shown in Figure 6.6 and Figure 6.7, they correspond to the intermediate and original images.

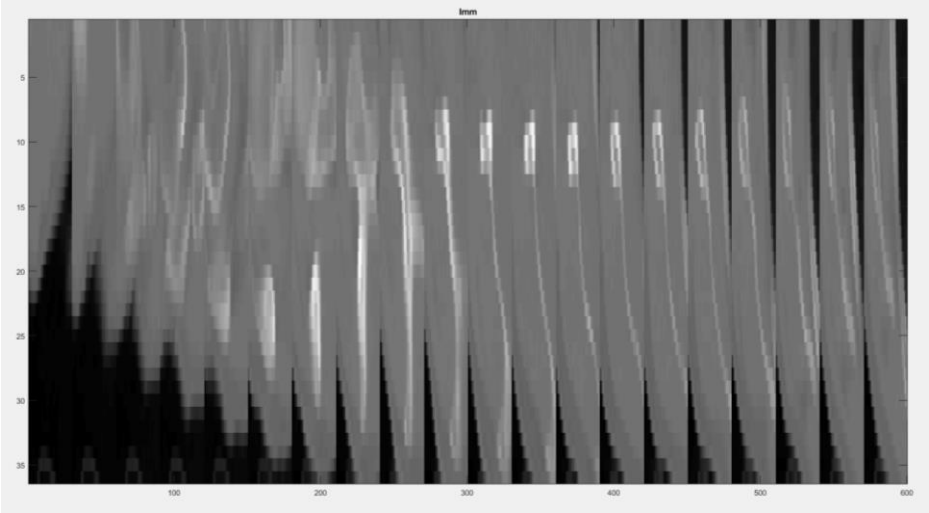


Figure 6.6 Window of the intermediate image.

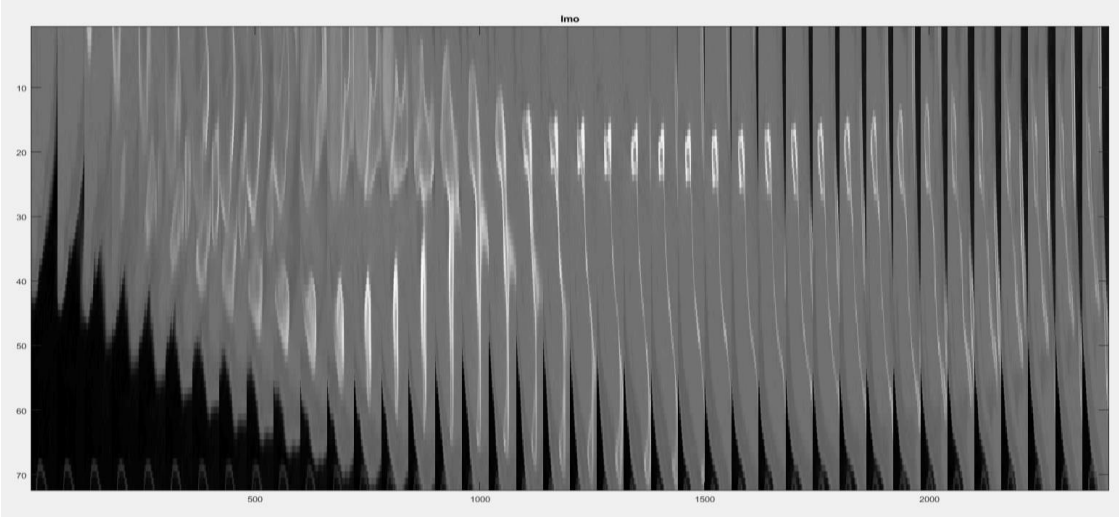


Figure 6.7 Window of the original image.

Using the windowed segmented image, shown in Figure 6.5 a threshold technique is used to select the parts of interest, because the parts of interest have a higher intensity value in the image. The binary image is then amplified to have the same dimensions as the intermediate image shown in Figure 6.6. the resulting image is shown in Figure 6.8.

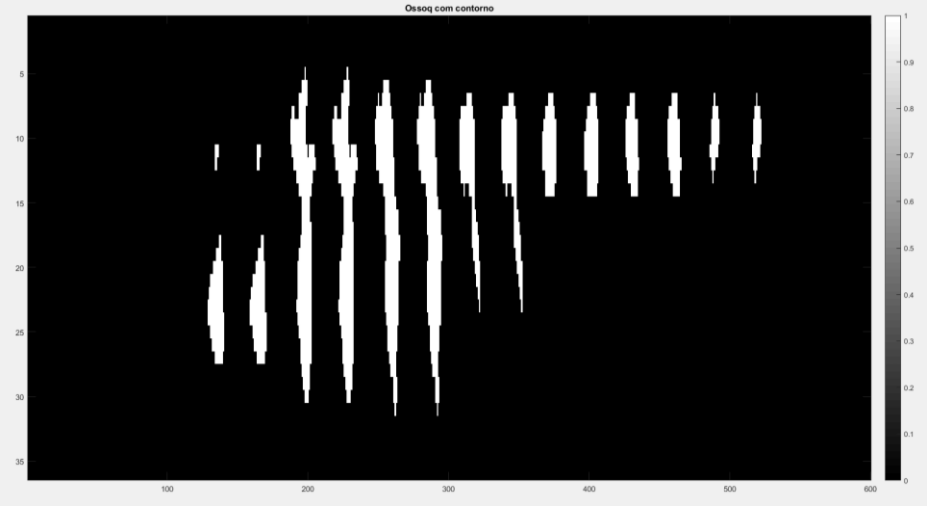


Figure 6.8 Binary image of the window of the segmented image.

From Figure 6.8, the boundaries have been removed to make sure that there are no artefacts in the higher dimensional image, as it is shown in Figure 6.9. This image will be used as the seed map in the next fuzzy connectedness segmentation.

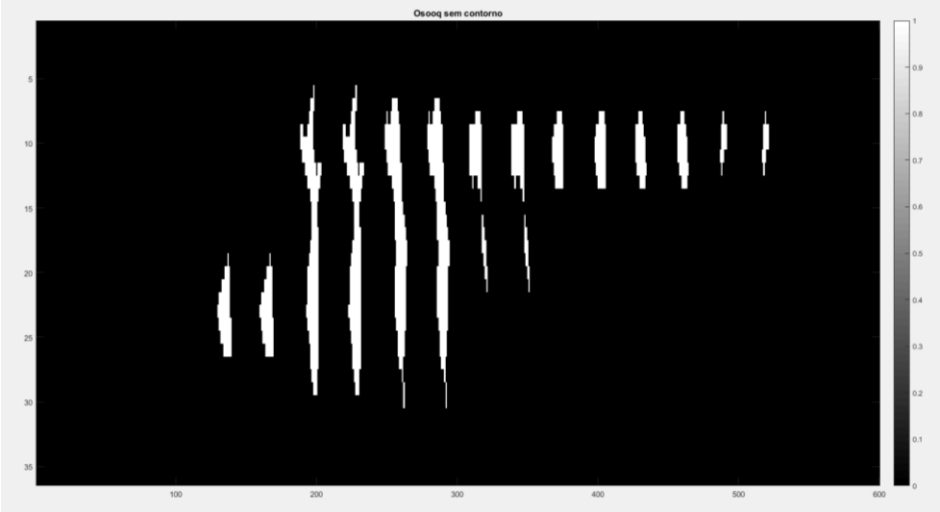


Figure 6.9 Seeds of the window of the intermediate image.

The results of the segmentation mentioned above are shown in Figure 6.10 and the process begins again.

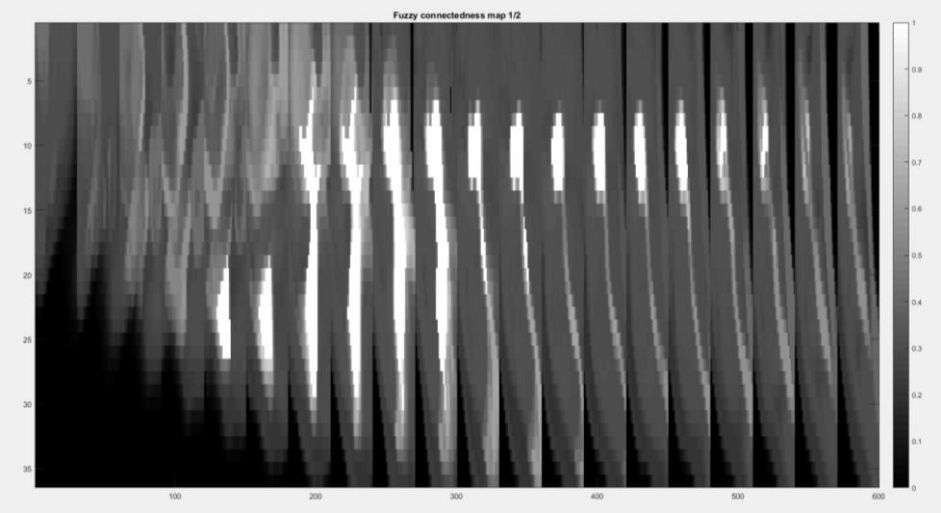


Figure 6.10 Segmented window of the intermediate image.

Using the segmentation presented in Figure 6.10, the thresholding technique is run. The resulting image is then amplified and the boundaries of each of the structures are then removed. The results of these processes are shown in Figure 6.11

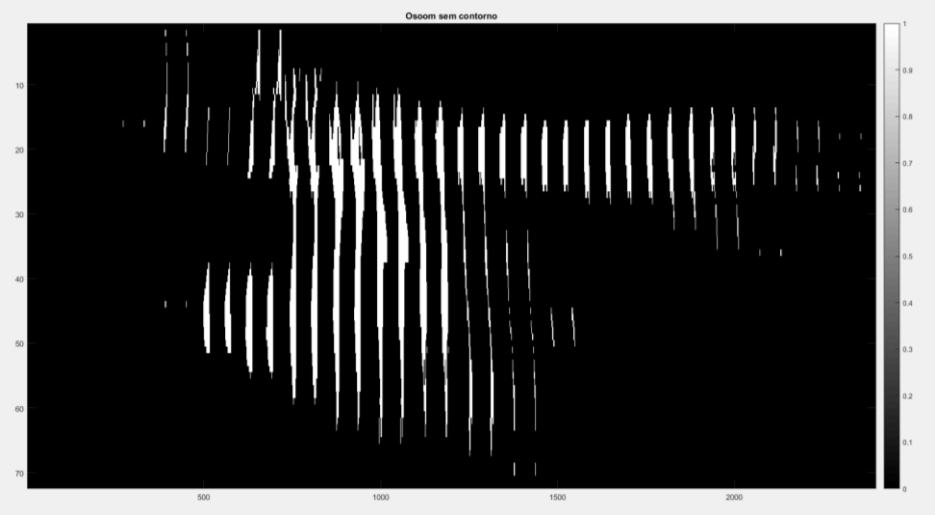


Figure 6.11 Seeds of the window of the original image.

The image shown in Figure 6.11 is then used as the seed map for the segmentation of the original image, shown in Figure 6.7. The result of the fuzzy connectedness segmentation process is shown in Figure 6.12.

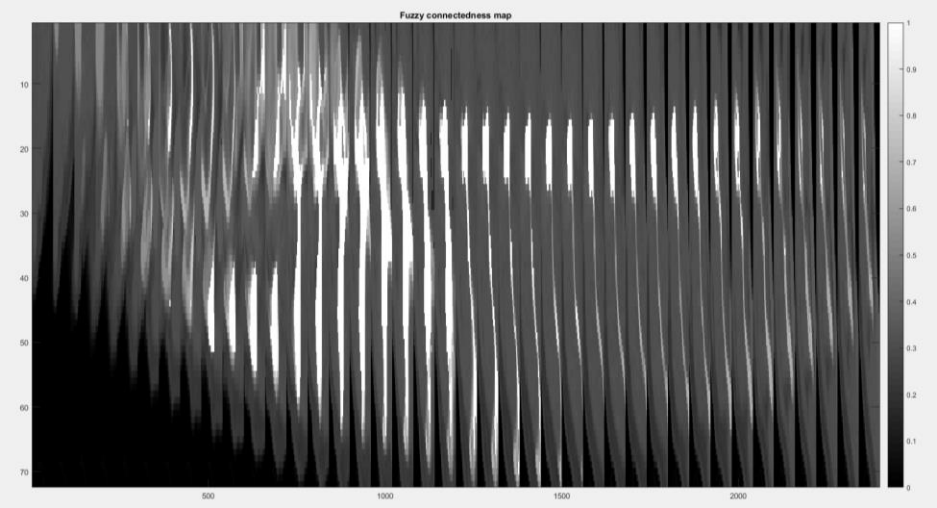


Figure 6.12 Final segmentation of the window of the original image.

After the segmentation is complete, a threshold is applied to the image, using the final threshold mention in the chapter 3, to select the regions that constitute the bone. This information is then used to create a 3D model of the segmented bone, that is presented in Figure 6.13.

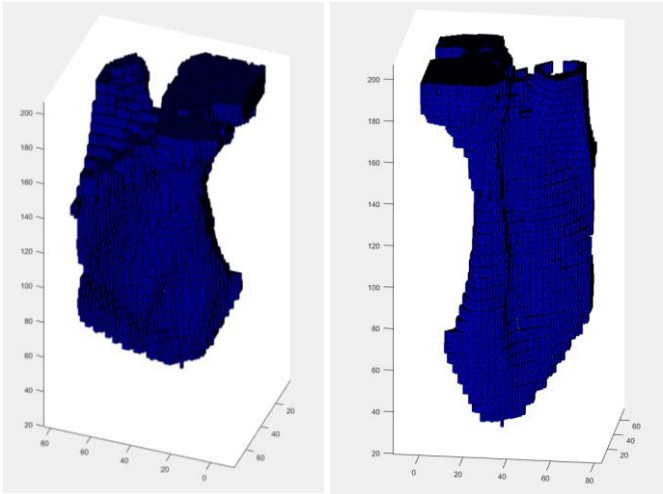


Figure 6.13 3D model of the segmented image. (a) and (b) are different views of the same model.

Chapter 7. Conclusion

There was no access to already segmented CT images, as such the results obtained can only be qualified in a qualitative way. The results of the fuzzy connectedness segmentation, presented in chapter 3, seem promising, especially with the implementation of the multiresolution and window methods. When these strategies were implemented, the segmentation took considerably less time to complete, reducing the segmentation time to almost one tenth of the original time, while using the same number of slices, and maintaining the results of the area of interest. The isolation of the region of interest, achieved with these techniques, also allowed more slices to be selected while maintaining a relatively fast segmentation of the area, which is perfect for the goal of developing a prototype to segment a selected bone from the image volume.

The results are not perfect, however, as the algorithm does tend to suffer from the leakage effect, an effect that creates a number of false positives, which means, that more spels, than the ones intended, are considered to be part of the bone. These false positives are not easily removable from the final segmentation once they have been mislabelled. Because only the spels situated in the boundaries of the segmentation are re-evaluated.

There were several attempts to try to prevent this effect, however, there wasn't a lot of success when using only one seed in this segmentation method. This may be due to the fact that the only parameters that can be modified to stop the leakage effect are the mean and standard variation of the affinity function. However, by changing those values from the calculated ones, it caused other artefacts in the segmentation images, such as, holes in the bones, especially visible in the 3D model of said segmentation.

The iterative relative method, explained in chapter 4, was not implemented in the prototype, even though it had promising results. Due to this methods iterative and relative nature, it is required to run the fuzzy connectedness algorithm a great number of times to present a final good solution with the regions well separated from one another. When using this algorithm in a three dimensional setting it gets rather challenging as the user needs to choose all the necessary seeds in all the slices to have a good result. To do so, the user needs to have a good perception of the human body and of the dataset itself to check every slice to make sure they have all the seeds required. Besides, segmenting a three dimensional image requires a lot of processing power, as the fuzzy connectedness algorithm needs to be run multiples times, with many different seeds as its starting point.

When considering the already segmented bone, the problem changes from identifying the target bone to separating the different bones that may be segmented together. This is especially useful in the spine, as it was shown in chapter 4. However, the number of seeds required for this was rather high, even though the number of slices segmented was only 7. As such, this method is much more dependent on the seeds chosen to work properly. In case it would be used on the prototype, it would require more interaction with the user, to select a number of seeds, and some previous knowledge on the algorithm itself and how it behaves to be able to choose those seeds.

As an attempt to get over the obstacle of the multiple seeds, an automatic seed generator was considered to allow the use of the prototype without requiring too much effort, or knowledge from anyone using it. The first one considered was a random seed generator. Each time the algorithm is run, the image is segmented in a different manner, as it is expected with a random seed generator, as such, the structures were not divided in a consistent manner. Some regions contained more than one different bone, while others covered very few spels, proving the algorithm too inconsistent and unreliable too be used in the prototype.

It was also considered an algorithm where the seeds were chosen according to their distance to the original one, the seed chosen for the first fuzzy connectedness method. By having a certain distance between each seed, each seed could potentially be placed in a different region. The results were the same as the random generator, presenting too much inconsistency in the location of the seeds, because they were all dependent of the first initial seed chosen. As such, each segmentation would also have seeds in different places.

The scale-based algorithm, in case it was implemented in the algorithm, it would not be hard for the user to interact with, as the seed chosen, as long as it is bone, is not very relevant. However, the processing power and the time require to do a segmentation of one single image far surpasses the practical uses in this kind of prototype, with the segmentation taking more than 4 hours. Doing this in a 3D fashion is simple not practical and not efficient for the results obtained.

The prototype allows the user to select the bone they want to be segmented, by selecting the seed of the algorithm, which is relevant in the final result, because the algorithm is seed dependant. When choosing the seeds, there are, in fact, some seeds that will not present any segmentation of the bone whatsoever, because the seed selected is from another tissue. It is also possible to choose a spel that is part of the bon that has a big intensity difference to the mean, resulting in the bone having low affinity values and the resulting window be small.

The prototype presents the final solution as a 3D model of the segmented bone from the image volume. This model is completely dependent of the seed and the values chosen for each one of the threshold parameters and techniques. If the threshold techniques do not contain the right values, as it was demonstrated, in chapter 3, the segmentation may present many artefacts or even not show a model whatsoever.

Many tests were performed using many different values for each one of those thresholds, however, there are many variations along the different bones of the regions of the body, as well as from CT image to CT image as there are different sizes and resolutions and proximity of the machine to the target. Human beings are also different from one another, with different height, making some bones longer or shorter, and different weight. Since the human skeleton generally represents 15 percent of the mass to support the weight of the rest of the body, someone who is heavier may have denser or larger bones that someone who is thinner.

The project was too ambitious, because creating an algorithm that can be used in any part of the body takes more time and resources, in this case segmented medical images, than the ones available. A more detailed work in a specific part of the body would have been more productive way of facing the problem of medical imaging segmentation. In future work, the algorithm and the prototype can be improved, with the time to develop a better way of interacting with the interactive relative fuzzy connectedness method and the segmented medical images, to be able to quantify the efficiency of the algorithm.

Appendix A Materials

In order to develop and test the performance of the proposed algorithm, two volumes were utilized, reserving the rest of the volumes for testing the prototype.

Test Volumes

Test image volume - The volume consists of 209 slices obtained from a single CT scan. The exam was performed in the department of nuclear medicine in the hospital of the university of Coimbra. Resolution: 512 by 512, side 1,178mm with voxel height 5mm.

Experiment image volume - The volume consists of 208 slices obtained from a single CT scan. The exam was performed in the department of nuclear medicine in the hospital of the university of Coimbra. Resolution: 512 by 512, side 1,178mm with voxel height 5mm.

Public Volumes:

Osirix Bebrix Foie (OSXBF)[27]- The volume consists of 244 slices of CT images from one single test scan. Resolution: 512 by 512 with voxel height of 2mm.

Osirix Bebrix Veineux (OSXBV) [27]- The volume consists of 394 slices of CT images from one single test scan. Resolution: 512 by 512 with voxel height of 2mm.

Bibliography

[1] S. Faisan, N. Passat, V. Noblet, R. Chabrier, J. P. Armspach, and C. Meyer, "Segmentation of head bones in 3-D CT images from an example," *2008 5th IEEE Int. Symp. Biomed. Imaging From Nano to Macro, Proceedings, ISBI*, pp. 81–84, 2008.

[2] X. Huang, "Medical Image Segmentation," no. i, pp. 1–35.

[3] G. Läthén, *"Segmentation Methods for Medical Analysis"* no. 1434. 2010.

[4] G. Hermosillo, C. Chefd?hotel, K.-H. Herrmann, G. Bousquet, L. Bogoni, K. Chaudhuri, D. R. Fischer, C. Geppert, R. Janka, A. Krishnan, B. Kiefer, I. Krumbein, W. Kaiser, M. Middleton, W. Ou, J. R. Reichenbach, M. Salganicoff, M. Schmitt, E. Wenkel, S. Wurdinger, and L. Zhang, *Image Registration in Medical Imaging: Applications, Methods, and Clinical Evaluation*. 2011.

[5] A. R. Natsheh, P. Vs, N. Anani, D. Benchebra, A. El-, and P. Norburn, "Segmentation of B Bone Structure in Sinus s CT Images Using g Self-Organizing Maps s," *Digit. Signal Process.*, pp. 290–293, 2010.

[6] M. Kabade, M. Manohar, and P. A. S. Patil, "A Review on Techniques of Image Segmentation," vol. 5, no. 3, pp. 3–5, 2016.

[7] S. Cagnoni and a B. Dobrzeniecki, "Segmentation of 3D medical images through algorithms," pp. 1–19, 1997.

[8] D. J. Brenner and E. J. Hall, "Computed Tomography — An Increasing Source of Radiation Exposure," pp. 2277–2284, 2007.

[9] D. Wang, H. Lu, J. Zhang, and J. Z. Liang, "A knowledge-based fuzzy clustering method with adaptation penalty for bone segmentation of CT images," *Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, vol. 6, pp. 6488–6491, 2005.

[10] D. L. Pham, C. Xu, and J. L. Prince, "CURRENT METHODS IN MEDICAL IMAGE SEGMENTATION," *Annu. Rev. Biomed. Eng.*, vol. 2, no. 1, pp. 315–337, 2000.

[11] R. Deklerck, J. Cornelis, and M. Bister, "Segmentation of medical images,"

Image Vis. Comput., vol. 11, no. 8, pp. 486–503, 1993.

[12] J. Wu, P. Davuluri, K. Ward, C. Cockrell, R. Hobson, and K. Najarian, "A new hierarchical method for multi-level segmentation of bone in pelvic CT scans.," *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, vol. 2011, pp. 3399–402, 2011.

[13] J. Zhang, C.-H. Yan, C.-K. Chui, and S.-H. Ong, "Fast segmentation of bone in CT images using 3D adaptive thresholding," *Comput. Biol. Med.*, vol. 40, no. 2, pp. 231–236, 2010.

[14] L. I. Wang, M. Greenspan, and R. Ellis, "Validation of Bone Segmentation and Improved 3-D Registration Using Contour Coherency in CT Data," vol. 25, no. 3, pp. 324–334, 2006.

[15] M. Lata, "Medical image segmentation," vol. 2, no. 4, pp. 1209–1218, 2010.

[16] L. Grady, "Random walks for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1768–1783, 2006.

[17] A. Fabijanska and J. Goclawski, "The Segmentation of 3D Images Using the Random Walking Technique on a Randomly Created Image Adjacency Graph," *IEEE Trans. Image Process.*, vol. 24, no. 2, pp. 524–537, 2015.

[18] J. K. Udupa and S. Samarasekera, "Fuzzy Connectedness and Object Definition: Theory, Algorithms, and Applications in Image Segmentation," *Graph. Model. Image Process.*, vol. 58, no. 3, pp. 246–261, 1996.

[19] J. K. Udupa, S. Member, P. K. Saha, and R. A. Lotufo, "Disclaimer : ' Relative Fuzzy Connectedness and Object Definition : Theory , Algorithms , and Applications in Image Segmentation ' Relative Fuzzy Connectedness and Object Definition : Theory , Algorithms , and Applications in Image Segmentation," 2002.

[20] K. C. Ciesielski, J. K. Udupa, P. K. Saha, and Y. Zhuge, "Iterative relative fuzzy connectedness for multiple objects with multiple seeds," *Comput. Vis. Image Underst.*, vol. 107, no. 3, pp. 160–182, 2007.

[21] J. K. Udupa, S. Member, and P. K. Saha, "Fuzzy Connectedness and Image Segmentation," vol. 91, no. 10, 2003.

[22] P. K. Saha, J. K. Udupa, and D. Odhner, "Scale-Based Fuzzy Connected Image Segmentation : Theory , Algorithms , and Validation," vol. 174, pp. 145–174, 2000.

[23] Z. Xu, U. Bagci, C. Jonsson, S. Jain, and D. J. Mollura, "Efficient Ribcage Segmentation from CT Scans Using Shape Features," *36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, pp. 2899–2902, 2014.

[24] Joakim Lindblad. Adjacency (https://searchcode.com/codesearch/view/13890019/), search code.

[25] Joakim Lindblad. Absolute Fuzzy Connectedness (https://searchcode.com/file/13890018/image_analysisII/Lab3/FuzzConn/afc.m), search code.

[26] Adam H. Aitkenhead (2010). Converting a 3D logical array into an STL surface mesh (https://www.mathworks.com/matlabcentral/fileexchange/27733-converting-a-3d-logical-array-into-an-stl-surface-mesh), MATLAB Central File Exchange.

[27] Public Dicom datasets [Online] (http://www.osirix-viewer.com/resources/dicom-image-library/)