SEGMENTATION, REGISTRATION AND VISUALIZATION OF MEDICAL IMAGES FOR TREATMENT PLANNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÖZGÜR TUNCER

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE

DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2003

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Mübeccel DEMİREKLER
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Mete SEVERCAN
Supervisor

Examining Committee Members

Assoc. Prof. Aydın ALATAN _____

Prof. Dr. Mete SEVERCAN _____

Prof. Dr. Nevzat GENCER _____

Prof. Dr. Semra CİĞER (Hacettepe University) _____

Dr. Uğur Murat LELOĞLU (TÜBİTAK-BİLTEN) _____

ABSTRACT

SEGMENTATION, REGISTRATION AND VISUALIZATION OF MEDICAL IMAGES FOR TREATMENT PLANNING

TUNCER, Özgür

M.S. Department of Electrical and Electronics Engineering
Supervisor : Prof. Dr. Mete Severcan

September 2003, 81 pages

Medical imaging has become the key to access inside human body for the purpose of diagnosis and treatment planning. In order to understand the effectiveness of planned treatment following the diagnosis, treated body part may have to be monitored several times during a period of time. Information gained from successive imaging of body part provides guidance to next step of treatment. Comparison of images or datasets taken at different times requires registration of these images or datasets since the same conditions may not be provided at all times. Accurate segmentation of the body part under treatment is needed while comparing medical images to achieve quantitative and qualitative measurements. This segmentation task enables two dimensional and three dimensional visualizations of the region which also aid in directing the planning strategy.

In this thesis, several segmentation algorithms are investigated and a hybrid segmentation algorithm is developed in order to segment bone tissue out of head CT slices for orthodontic treatment planning. Using the developed segmentation

algorithm, three dimensional visualizations of segmented bone tissue out of head CT slices of two patients are obtained. Visualizations are obtained using the MATLAB Computer software's visualization library.

Besides these, methods are developed for automatic registration of two-dimensional and three-dimensional CT images taken at different time periods. These methods are applied to real and synthetic data. Algorithms and methods used in this thesis are also implemented in MATLAB computer program.

Key words: Medical Imaging, CT, Segmentation, Registration, Visualization, Region Growing, Iterative Closest Point

ÖZ

TEDAVİ PLANLAMASI AMACIYLA TIBBI GÖRÜNTÜLERİN BÖLÜTLENMESİ, ÇAKIŞTIRILMASI VE GÖRSELLEŞTİRİLMESİ

TUNCER, Özgür

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Mete Severcan

Eylül 2003, 81 sayfa

Tıbbı görüntüleme, tanı ve tedavi planlaması amacıyla insan vücudunun içine erişebilmenin anahtarı olmuştur. Tanı sonrası uygulanacak tedavi planlamasının etkisinin anlaşılması için, tedavi altındaki vücut bölgesinin belirli bir sure içerisinde defalarca görüntülenmesi gerekebilir. Vücudun belirli bölgesinden sıralı elde edilen görüntülerin birbirleriyle karşılaştırmasıyla kazanılan bilgi, tedavinin bir sonraki aşaması için yol gösterici olmaktadır. Sıralı çekimler sırasında aynı koşullar yaratılamayabileceğinden, farklı zamanlarda elde edilen görüntülerin veya veri kümelerinin karşılaştırılabilmesi, görüntülerin veya veri kümelerinin çakıştırılmasına bağlıdır. Niteliksel ve niceliksel ölçümlerin elde edilebilmesi için tıbbi görüntülerin karşılaştırılması sırasında vücudun tedavi altındaki bölgesinin doğru bölütlenmesi gerekir. Bu bölütleme, planlama stratejisine yön vermeye yardım eder ve ilgili bölgenin iki boyutlu ve üç boyutlu görüntülerinin oluşturulmasını sağlar. Bu tezde, farklı bölütleme algoritmaları araştırılmış ve kafa bilgisayarlı tomografi görüntülerinden kemik dokuyu bölütleme için hibrid bölütleme algoritması geliştirilmiştir. Geliştirilen bölütleme

algoritması kullanılarak kemik dokusu bölütlenmiş kafa tomografi görüntülerinin farklı hastalara ait üç boyutlu modelleri görselleştirilmiştir. Tez sırasında verileri görselleştirmek için MATLAB bilgisayar programının görselleştirme kütüphanesinden faydalanılmıştır.

Bunlara ek olarak farklı zamanlarda alınan iki boyutlu ve üç boyutlu bilgisayarlı tomografi görüntülerinin otomatik olarak çakıştırılabilmesi için yöntemler geliştirilmiştir. Bu yöntemler sentetik ve gerçek veriler üzerinde uygulanmıştır. Bölütleme ve çakıştırma algoritmaları ve yöntemleri de MATLAB bilgisayar programı içinde gerçeklenmiştir.

Anahtar Kelimeler: Tıbbi görüntüleme, bilgisayarlı tomografi, bölütleme, çakıştırma, görselleştirme, bölge büyütme, yinelemeli yakın noktalar

To my family and Orfoz

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to Prof. Dr. Mete SEVERCAN for his supervision, valuable guidance and helpful suggestions.

I would like to express my sincere appreciation to Dr. Hakan EL and Prof. Dr. Semra CİĞER for their support in providing head CT images used throughout the thesis. I am also grateful to ASELSAN Inc. for facilities provided for the completion of this thesis.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURES

CHAPTER 1

1. INTRODUCTION

1.1 Introduction to the project

Medical Imaging gained an increased importance over the past thirty years after the invention of the computerized tomography and various researches have been carried out since then. The discovery of computerized tomography scan revolutionized medical diagnosis, allowing doctors to see exactly what is going on inside the body. Combined with powerful, robust and fast segmentation, registration and visualization algorithms, it is now possible to carry out surgical planning even during the operation [1].

Medical imaging can be divided into several phases which start with digitizing the images and gathering them to obtain a dataset. If the dataset under investigation is from a CT or an MRI machine, it is in the form of slices. Interpolation is needed when these slices are not adequate or the dataset formed from these slices is not isotropic. Following the interpolation, the body part under investigation should be segmented out from the whole data set. Individual slices can be combined to provide a three-dimensional visualization. If datasets from various imaging techniques or datasets taken within a time period are to be used, registrations of the datasets are needed. Segmented and registered datasets can then be used to identify an abnormality within the body and to form a treatment plan.

1.2 Aim and scope of the study

In this thesis; various segmentation algorithms are investigated for the purpose of bone segmentation from head CT slices. Methods for registration of

two dimensional and three dimensional automatic registrations of datasets are studied. Visualizations of segmented bone tissue and visualization results of registration algorithms are shown. Within this study, head CT dataset provided by Hacettepe University Orthodontics department are used in developing the algorithms. These slices were taken with 2 mm slice thickness with a variable pixel resolution from dataset to dataset and they are in DICOM format. In a dataset, there are more than two hundred slices. Aim of the study is to extract the bones within the slices in order to construct the three dimensional visualization of the craniofacial complex. Visualizations of the craniofacial complex at different stages of treatment will reveal the effectiveness of the orthodontic treatment.

Originally images in the CT dataset are recorded as 16-bit and have a resolution of 512 by 512. In order to make the computation easier and to speed up the execution time, the image content is converted to 8-bit format and size of the images reduced using cubic spline interpolation with MATLAB built-in functions.

1.3 Outline of the thesis

Chapter 2 is devoted to literature survey about medical imaging. Different types of medical imaging techniques are investigated and brief information is given about them. In the subsequent sections, the phases of the medical imaging covering interpolation, segmentation, registration and visualization are studied. Chapter 2 ends with information given about Digital Imaging and Communications in Medicine (DICOM), a standard for medical imaging.

In Chapter 3, segmentation algorithms are investigated. Each algorithm is tested with real data and algorithms are discussed according to the results. Chapter 3 describes the hybrid segmentation algorithm developed to segment the bone structure from the head CT slices. Chapter 3 also includes, three dimensional visualizations of head CT slices of two patients segmented by the hybrid segmentation algorithm. These visualizations are obtained using the visualization library of MATLAB computer software.

In Chapter 4, two dimensional and three dimensional automatic registration algorithms are investigated. Experimental results of the registration algorithms using real and synthetic data are given. At the end of the chapter, developed methods are described both for automatic registration of two and three dimensional images.

Chapter 5 is devoted to the conclusions of results obtained from all of the algorithms investigated throughout the thesis. Within this chapter, suggestions for future work is also provided.

In Appendices, the MATLAB programs used in realizing the available algorithms and developed algorithms are given.

CHAPTER 2

2. LITERATURE SURVEY

2.1 Introduction

Medical Imaging dates back to 1895 when Wilhelm Roentgen accidentally discovered that a cathode-ray tube could make a sheet of paper coated with barium platinocyanide glow, even when the tube and the paper were in separate rooms. Roentgen decided that the tube must be emitting some sort of penetrating rays. He named them X for unknown. Shortly afterward, he found that if he aimed these X-rays through a person's hand at a chemically coated screen, he could see the bones in the hand clearly on the screen. In fact, the very first radiograph of human anatomy was Mrs. Roentgen's left hand, revealing the skeleton inside which is shown in Figure 2.1 [2].

Figure 2.1 First X-Ray image showing the bones of Mrs. Roentgen's left hand

Over the next few decades, X-rays grew into a widely used diagnostic tool. Since bones show up clearly as white objects against a darker background, Roentgen's rays proved particularly suited for examining fractures and breaks, but they could also spot cancer tumors, respiratory diseases such as tuberculosis or black lung, and a variety of other tissue abnormalities via the injection of a contrast material [3]. The great strengths of X-rays are their high resolution with details as small as 0.1 millimeter and their ease of use. Using X-rays for medical diagnosis has some drawbacks. X-rays do not distinguish well between tissues of similar densities. In addition to this, X-ray machines expose potentially harmful radiation which limits the usage areas and frequency.

For more than half a century, the science of medical imaging grew steadily but slowly, as incremental improvements was made in the X-ray technique. With the introduction of computerized tomography in the early 1970's, the growing interest in medical imaging speeded up. Computerized tomography relies on taking a series of X-rays, sometimes more than a thousand, from various angles and combining them with a computer. Computerized tomography made it possible to build up a three-dimensional visualization of any part of the body. Also using the interpolation techniques, it is possible to obtain two dimensional slices from any angle and at any depth just as if the body is sliced with a sharp knife and taken a picture of the result. The CT scan revolutionized medical diagnosis, allowing doctors, for instance, to see if a head injury had produced any bleeding in the brain or to make out the shape and extent of tumors in cancer patients [4],[5].

CT was the first of what has proved to be a flood of new medical imaging tools, each of which reveal different information about the body. Although the tools work according to various physical principles, the new medical imaging techniques all depend on computers to construct the images from a mass of data that is collected electronically instead of on film. These new techniques open up a wealth of possibilities, only some of which have been reaped, but they also demand that researchers solve a host of new problems in order to realize those possibilities.

2.2 Computerized Tomography

Tomography refers to the cross-sectional imaging of an object from either transmission or reflection data collected by illuminating the object from many different directions. The impact of this technique in diagnostic medicine has been revolutionary, since it has enabled doctors to view internal organs with unprecedented precision and safety to the patient. The first medical application utilized X-rays for forming images of tissues based on their X-ray attenuation coefficient [3]**.** More recently, however, medical imaging has also been successfully accomplished with radioisotopes, ultrasound, and magnetic resonance; the imaged parameter being different in each case.

Fundamentally, tomographic imaging deals with reconstructing an image from its projections. A projection of an image at a given angle is the integral of the image in the direction specified by that angle, however, in a loose sense, projection can be thought as the information derived from the transmitted energies, when an object is illuminated from a particular angle. The energy source herein can be a beam of photons; can be ultrasonic waves or microwaves each of which determine the type of tomographic imaging.

There are numerous nonmedical imaging applications which utilize the computerized tomography. Researchers have already applied this methodology to the mapping of underground resources via cross-borehole imaging, some specialized cases of cross-sectional imaging for nondestructive testing (NDT), the determination of the brightness distribution over a celestial sphere, and three-dimensional imaging with electron microscopy [6].

Three types of computerized tomography has been developed which have all different application areas. These are X-Ray CT, Emission CT and Ultrasonic CT that will be explained briefly.

2.2.1 X-Ray Computed Tomography

X-Ray CT, the first of all CT methods developed, utilizes X-rays for forming images of tissues based on their attenuation coefficient. Although X-Ray computerized tomography has found its biggest use in medical imaging there are several applications in the area of nondestructive testing (NDT) of materials and industrial objects. X-ray tomography is widely used for the analysis of the bone structure since bones have an attenuation coefficient much larger than any other tissue in the human body. Figure 2.2 shows different slices of a child head recorded on a CT machine.

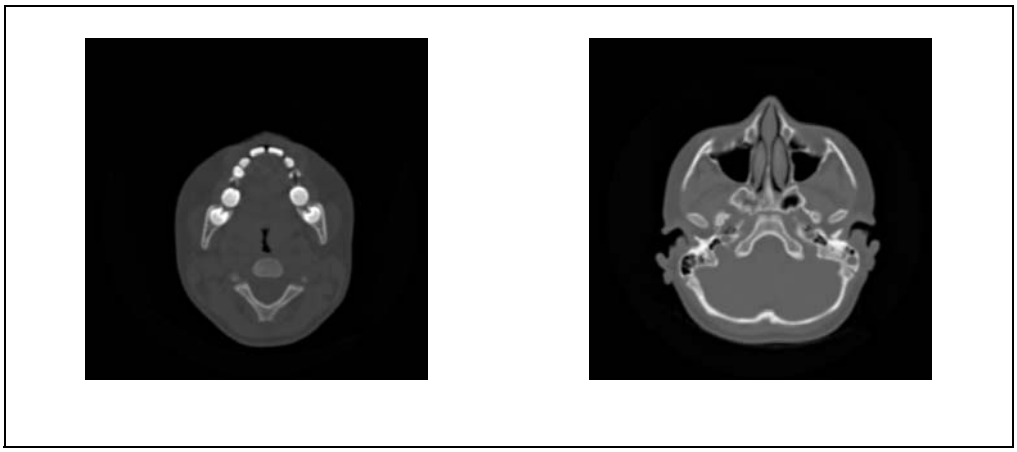

Figure 2.2 CT slices of different parts of the head

2.2.2 Emission Computed Tomography

Emission CT uses the decay of radioactive isotopes to image the distribution of the isotope as a function of time. These isotopes may be administered to the patient either by injection or by inhalation. Thus, for example, by administering a radioactive isotope by inhalation, emission CT can be used to trace the path of the isotope through the lungs and the rest of the body. There are two types of emission CT: Single photon emission CT where the result of the decay is a single photon and positron emission CT where the decay produces a single positron. Figure 2.3 shows a sequence of positron emission tomography images showing the existence of Parkinson's disease in the head.

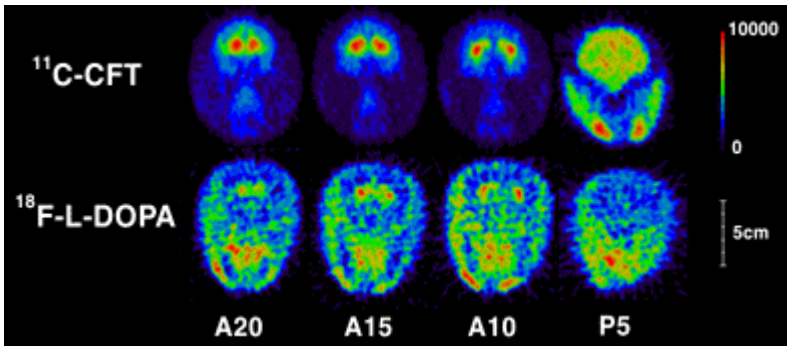

Figure 2.3 PET images of the head showing the existence of Parkinson's disease

2.2.3 Ultrasonic Computed Tomography

Similar to X-ray tomography, an ultrasonic wave transmitter illuminates the object and by measuring the energy on the far side of the object a line integral of the attenuation can be estimated in ultrasonic computed tomography. Since the propagation speed of ultrasonic waves are much smaller than X-rays, the exact pressure of the wave as a function of time can be measured. Using this wave function, it is possible to measure the delay caused by the examined object which will correlate to the refractive index of the object under examination.

2.2.4 Magnetic Resonance Imaging

Although the principles of nuclear magnetic resonance have been well known since the 1950s, only since 1972 has it been used for imaging. In the sense that the images produced represent a cross section of the object, MRI is a tomographic technique. Magnetic resonance imaging is based on the measurement of radio frequency electromagnetic waves as a spinning nucleus returns to its equilibrium state. Any nucleus with an odd number of particles such as protons and neutrons have a magnetic moment, and, when the atom is placed in a strong magnetic field, the moment of the nucleus tends to line up with the field. If the atom is then excited by another magnetic field it emits a radio frequency signal as the nucleus returns to its equilibrium position. Since the frequency of the signal is dependent on not only the type of atom but also the magnetic fields present, the position and type of each nucleus can be detected by appropriate signal processing. Two of the more interesting atoms for MRI are hydrogen and phosphorus. The

hydrogen atom is found most often bound into a water molecule while phosphorus is an important link in the transfer of energy in biological systems. Both of these atoms have an odd number of nucleons and thus act like a spinning magnetic dipole when placed into a strong field.

MRI's multi-planar capabilities and sensitivity to tissue differentiation makes it the procedure of choice for detecting abnormalities or lesions in most parts of the body. MRI has been successfully utilized in the detection of abnormalities in brain, neck, spine, chest, heart, breast, abdomen, pelvis, musculoskeletal body parts. Figure 2.4 shows an MRI image of the internal organs. Image is taken from GE medical systems website.

Figure 2.4 MRI image of internal organs

2.3 Phases of Medical Imaging

Medical imaging starts with choosing the type of imaging technique discussed above that will be used to diagnose the body part under investigation. Sometimes more than one imaging technique can be combined to explicitly determine the location of the abnormality within the body. The phases after this step can be subdivided into several categories, which include restoration, interpolation, segmentation, registration and visualization.

2.3.1 Restoration

Acquired images from the imaging equipment most often have to be restored initially before using them. Restoration covers the elimination of a noise source from the image content, adjusting the contrast in the whole image or in some sub region of an image or reducing the effects of image artifacts. For example, teeth filling, prostheses, fiducial markers are all sources of artifacts in a CT image that limit the reliability of the results. Although some methods [7],[8] have been developed to reduce the effect of these artifacts to some extend, if possible the source of artifact should be removed before the scanning operation for proper segmentation, registration and visualization.

2.3.2 Interpolation

Interpolation is required in imaging, in general, whenever the acquired image data are not at the same level of discretization as the level that is desired. In biomedical imaging systems, most of the case, the distance between adjacent image elements within a slice differ from the spacing between adjacent image elements in two neighboring slices. In addition, the spacing between slices may not be the same for all slices. These anisotropic data need to be converted to isotropic discretization or converted to level of desired discretization for visualization, manipulation, and analysis of data. If images from different viewpoints or from arbitrary planes are required such as the saggital or coronal view, interpolation is also needed.

If we want to register data acquired for the same object of study from two modalities or from the same modality at two separate time instances, one of them needs to be rediscretized to the discretization level of the other. Moreover, the resolution of the datasets may differ. Upon registering these datasets, level of discretization of one of them needs to be converted to that of the other. Although the scanner resolutions are improved with the developing scanner technology, these problems will continue to exist and interpolation will always be needed.

According to [9], interpolation techniques can be divided into two groups: scene-based and object-based. Scene-based methods use the intensity values of the given scene to find the interpolated scene intensity values. Nearest neighbor interpolation, linear interpolation, quadratic interpolation, cubic and B-spline interpolations are examples for the scene-based interpolation techniques. In object-based methods, some object information extracted from the given scene is used in carrying out the interpolation process. Shape-based interpolation is an example of object-based methods. It is used in applications which required slice-by-slice help from a user for the difficult segmentation task. In [9] different types of 3-D image interpolation methods are evaluated and shape-based averaging method was found to be the most accurate among the other methods evaluated. Also in [10] interpolation methods in medical image processing such as truncated and windowed sinc, nearest neighbor interpolation, linear interpolation are compared according to their runtime speed, computational complexity, qualitative and quantitative error determinations.

2.3.3 Segmentation

Image segmentation is separation of structures of interest from the background and each other. The goal of image segmentation is to find regions that represent objects or meaningful parts of objects. Medical image segmentation deals with segmentation of the body structures for visualization and volume estimation of objects of interest, detection of abnormalities, tissue quantification, preprocessing for image registration, preprocessing for surface registration, classification, and more. These can be segmentation of bones or coronal arteries in a CT dataset or segmentation of brain or lungs in an MRI dataset. Image segmentation techniques can be classified into several different classical approaches.

2.3.3.1 Threshold Techniques

Threshold techniques are based on the thresholds which are usually selected from the image histogram. It is said that all pixels whose value is between

two threshold values belong to one region. The fact that the thresholds are derived from the histogram says that these techniques don't take into account spatial information of the image and they have problems to cope with the noise as well as with blurred edges on the image.

Global thresholding is the simplest statistical thresholding technique, where pixels are classified based on their intensity values. However, choosing the right intensity method is difficult and varies from one dataset to other. The selection of the threshold can be interactively manual or automatic. In manual selection, the threshold is operative sensitive. Other thresholding techniques include iterative threshold selection, adaptive thresholding, variable thresholding, double thresholding [11].

2.3.3.2 Deformable Models

Deformable models are geometric descriptions of contours or surfaces which evolve under a suitable energy. These include methods based on snakes [12] which are energy minimizing splines influenced by imaging forces and methods based on balloons [13] which are an improved version of snakes that do not need to be initialized around edges.

2.3.3.3 Edge-based methods

Edge based methods try to locate the places of rapid transition from one region to the other of different brightness or color value. The basic principle is to apply some of the gradient operators convolving them with the image. High values of the gradient magnitude are possible places of rapid transition between two different regions, which we call edges. After this step of finding edges on the image, they have to be linked to form closed contours of the regions. Laplacian of Gaussian, Canny and Sobel are some of the operators used in edge detection [14].

2.3.3.4 Region-based Methods

Region-based methods are complementary to the edge-based methods. The point in region-based methods is to group pixels of the same or similar brightness or color to the regions according to the given criteria of homogeneity. These methods look at the neighboring pixels of the given pixel and merge them into the region if criteria of homogeneity are satisfied. Homogeneity criteria is based on some threshold value, the choice of which is problematic, because operator usually has to play a lot with the right choice of the thresholds, and thresholds always depend on the image data.

2.3.3.5 Mixed, hybrid methods, other techniques

Mixed methods use the available segmentation methods within several stages of the segmentation task. [15] is a good example where curve evaluation, region growing and region competition methods are combined to segment the carpal bones from CT slices and they called the hybrid method skeletally coupled deformable model. In addition to the classical methods, discussed above, there are segmentation techniques based on neural networks [16], mathematical morphology [17] and watersheds [18].

2.3.4 Registration

Registration is a fundamental task in image processing used to match two or more pictures taken, for example, at different times, from different sensors, or from different viewpoints. Matching a target with a real-time image of a scene for target recognition, monitoring global land usage using satellite images, matching stereo images to recover shape for autonomous navigation, and aligning images from different medical modalities for diagnosis are some of the application areas where registration of images is heavily used [19].

Medical image registration deals with the registration of images gathered from different or same modalities taken at same time or within a time period. This could be the registration of CT images taken before and after

diagnosis to determine the development of the bone tissue or registration of MRI and CT images to visualize the brain and the bone structure at the same time.

Registration starts with selection of features from the two datasets to be registered. According to the features used in registration, registration methodology can be divided into to subgroups: Extrinsic and intrinsic registration. Extrinsic features rely on artificial objects such as stereotactic frames, attached to the patient, objects which are designed to be well visible and accurately detectable in all of the pertinent modalities. Registration of the acquired images is comparatively easy, fast, can usually be automated, and, since the registration parameters can often be computed explicitly, there is no need for complex optimization algorithms. [20] is an example for extrinsic registration where the authors used a fixture which they call "SIP Lab Innsbruck frame" which is shown in Figure 2.5, and proposed semi-automatic and fully automatic ways of registering CT and MRI datasets. The main drawback of extrinsic registration is its prospective character. Provisions must be made in the pre-acquisition phase, and often the invasive character of marker objects is a problem. Most of the time, the data is gathered without using special equipment and registration relies on the intrinsic methods. Intrinsic methods rely on the patient generated image content only. Registration can be based on landmarks, anatomical or geometrical, alignment of segmented binary structures, surfaces, or directly on to measures computed from the image grey values. Geometrical features consist of corners, local curvature or extrema generally localized in an automatic fashion [21], [22].

Following the selection of the features, whether intrinsic or extrinsic, a method has to be developed to register two datasets. First of all, type of the transformation involved in the registration process has to be decided. The transformation can be rigid in which the shape and size do not change or can be non-rigid.

Figure 2.5 “SIP Lab Innsbruck frame” fixated on the patient’s head

After deciding the type of transformation, the transformation matrix can be calculated by selecting the same features from two datasets. For a rigid transformation, at least four conjugate pairs have to be selected to calculate the nine unknowns in rotation matrix and three unknowns in the translation matrix. Selection of candidate conjugate points in three dimensional sets is a difficult task so most of the time registration of features is carried out using algorithms. The most popular algorithm used for registration is the iterative closest point algorithm [23]-[28], where the features used in registration can be points, curves or surfaces.

2.3.5 Visualization

Visualization of three dimensional biomedical volume images has traditionally been divided into two different techniques in [29]: Surface rendering and volume rendering. Both techniques produce a visualization of selected structures in the three dimensional volume image, but the methods involved in these techniques are quite different, and each has its advantages and disadvantages. Selection between these two approaches is often predicated on the particular nature of the biomedical image data, the application to which the visualization is being applied, and the desired result of the visualization.

2.3.5.1 Surface Rendering

Surface rendering techniques rely on the extraction of contours that define the surface of the structure to be visualized. These techniques represent the surface by a mosaic of connected polygons. This technique is advantageous since it requires relatively small amount of contour data which directly affects the rendering speed. Its disadvantage comes from the discrete nature of the surface polygon placement which makes this technique prone to sampling and aliasing artifacts on the rendered surface.

2.3.5.2 Volume Rendering

Volume rendering is one of the most versatile and powerful image display and manipulation techniques. Volume rendering has one such big advantage over surface rendering that, it does not need prior surface or object segmentation. Volume rendering technique uses the whole volume and since medical datasets containing hundreds of slices are large, computation speed is lower than the surface based rendering technique. Figure 2.6 shows a volume rendering of the head obtained from CT slices taken from medical imaging website of Siemens corporation.

Figure 2.6 Volume rendering of head

Virtual endoscopy, neosurgery, cardiac and coronary artery disease, craniofacial surgery planning and radiation treatment planning are different application areas where medical visualization is heavily used [30]-[33].

2.4 Digital Imaging and Communications in Medicine (DICOM)

Digital Imaging and Communications in Medicine standard (DICOM) was created by the National Electrical Manufacturers Association (NEMA) to aid the distribution and viewing of medical images, such as CT scans, MRI images, and ultrasound images. DICOM imaging allows interchange of medical images taken from different manufacturers of imaging equipments. DICOM standard also includes a network protocol utilizing TCP/IP for creating a network of imaging equipments. DICOM is used in medical profession including cardiology, dentistry, endoscopy, mammography, ophthalmology, orthopedics, pathology, pediatrics, radiation therapy, radiology, surgery and even in veterinary [34].

DICOM datasets contain a DICOMDIR file which stores the directory structure of the datasets and the way the images are related to each other. The images also have their own DICOM information that include the name of the patient, the imaging machine, pixel resolution, image position patient information in the slice, slice thickness used to obtain that slice and other information. Throughout the thesis version 6.5 of MATLAB is used to extract the information and the image content from the DICOM files using “dicominfo” and “dicomread” functions.

CHAPTER 3

3. SEGMENTATION OF BONE TISSUE FROM HEAD CT IMAGES

3.1 Introduction

In order to detect the changes within the bone structure over a period of time, first of all bone tissue has to be segmented out from other tissue parts. Segmentation of bone tissue from a CT dataset is somewhat easier than segmentation of other body parts because bones have higher X-ray attenuation coefficient and this makes them to be distinguishable from the other parts of the body. Although bones are easily identified, before developing an algorithm for segmentation, bone structure has to be considered. Bone in human and other mammalian bodies is generally classified into two types as shown in Figure 3.1: Cortical bone, also known as compact bone and Trabecular bone, also known as cancellous or spongy bone. These two types are classified on the basis of porosity and the unit microstructure. Also bones are classified according to their shape: Long, short, flat and irregular bones. Craniofacial bones which we want to segment are considered to be flat bones with spongy bone structure in between compact bone structure.

Different parts of the bone appear different while recorded on a CT slice. Spongy bone regions seem texturized and darker and compact bones appear brighter and have no texture with good edges. Also because of the nature of imaging, some edges of the bone are diffused and because of this diffusion, narrow inter-bone regions collapse together making segmentation difficult.

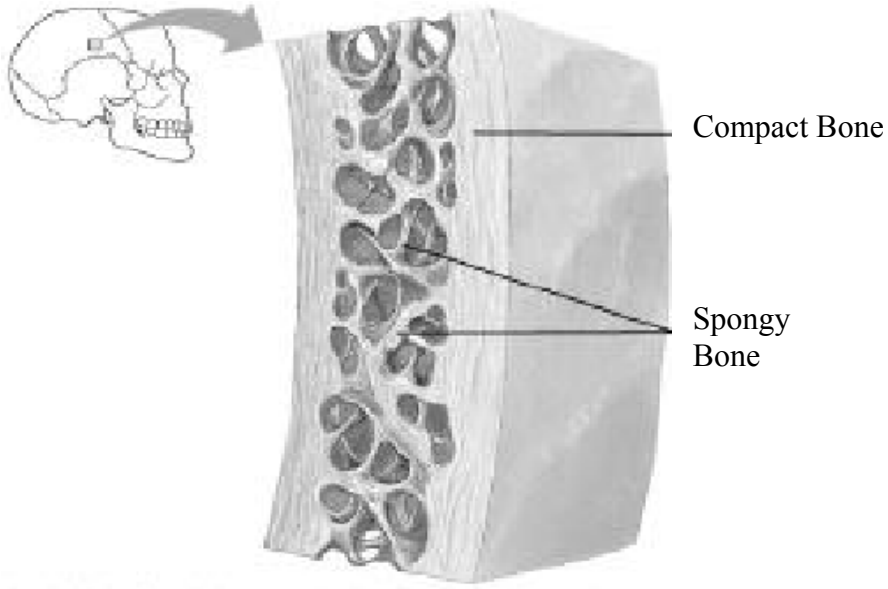

Figure 3.1 Craniofacial Bone Structure

Figure 3.2 clearly shows a diffused edge and types of bone regions in different portions of a CT slice.

During the segmentation of craniofacial bones, classification of bone structure whether compact or spongy is not important for our case. Important issue is extraction of exact shape and position of the bone structure. The algorithm has to classify bone and non-bone regions for this purpose.

3.2 Thresholding using the EM algorithm for segmentation of bone

Among the segmentation algorithms presented in Chapter 2, the basic and the easiest segmentation algorithm is the thresholding algorithm. Thresholding algorithm segments a body part according to a threshold that user provides or a threshold found automatically using the image content. For more than hundred slices, manual thresholding is somewhat a tedious, operator sensitive and a difficult task. Also the illimunation of the X-rays within different cross sections

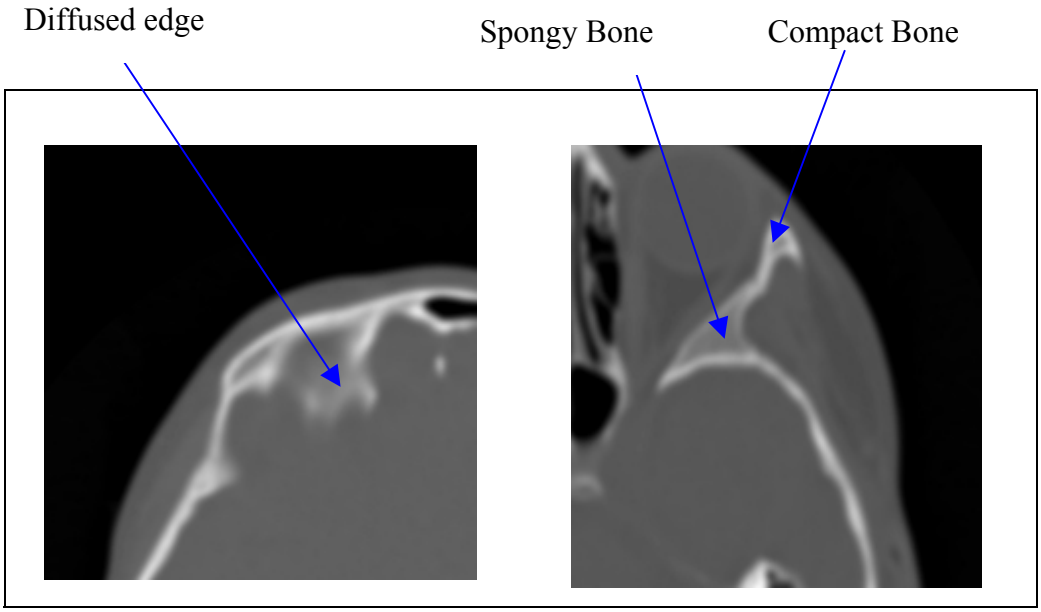

Figure 3.2 CT slice showing the bone structure

of the head are different, making a global threshold for the bone structure unavailable. Therefore, thresholds for different slices have to be selected individually according to the image content. The easiest way for selecting an automatic threshold is to think of the image histogram as consisting of sum of Gaussian distributions with different means and standard deviations. Figure 3.3 shows a head slice and its histogram where bone and non-bone regions are identified and circled in the histogram.

In an 8-bit image format, the brightest region in the image corresponds to 255 and the darkest region is 0 giving 256 discrete levels to describe the image content. Careful examination of the CT slices showed that all values lower than 50, correspond to noise and these values fall below any threshold level for bones. The circled regions in Figure 3.3, shows the non-bone and bone regions in the histogram. Each of these regions is considered to be Gaussian with a mean and standard deviation.

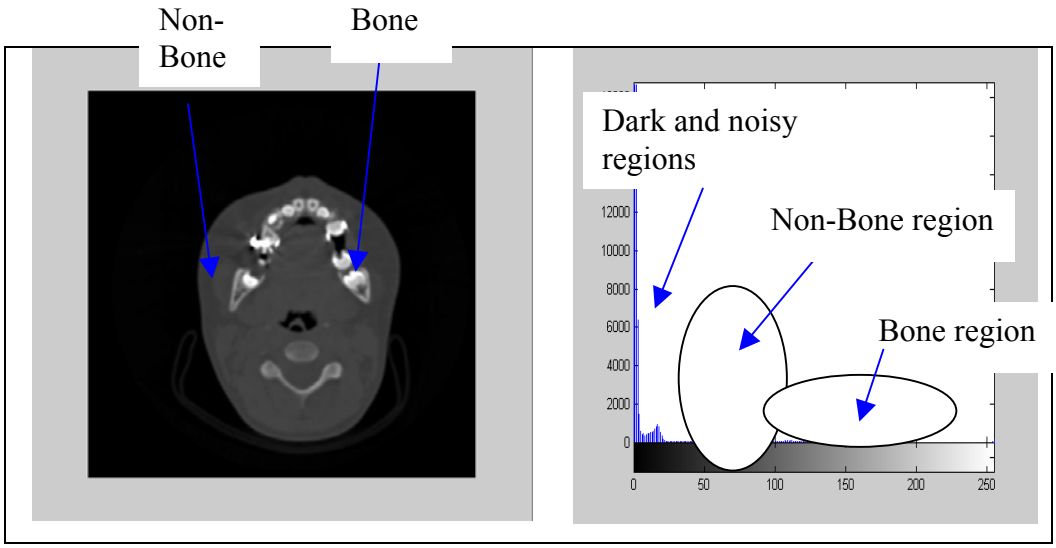

Figure 3.3 Head CT image and its histogram

Expectation maximization algorithm [35] is used to find the optimal threshold that divides these regions from each other. Expectation maximization algorithm simply guesses the parameters mean and standard deviation, calculates the weights of the Gaussian probability distribution functions and using these guesses, probabilities of the bone and non-bone regions in the histogram are calculated. This is the expectation step of the algorithm. Calculated probabilities and weights are used to create new guesses for the means and standard deviations which will be used in the next iteration of the algorithm and this constitutes the maximization step of the algorithm. This process continues until the parameters do not change a lot. After mean values and standard deviation are calculated, threshold is selected as mean value summed with standard deviation multiplied by a constant which user provides. The results of the thresholding algorithm are given in Figure 3.4**.**

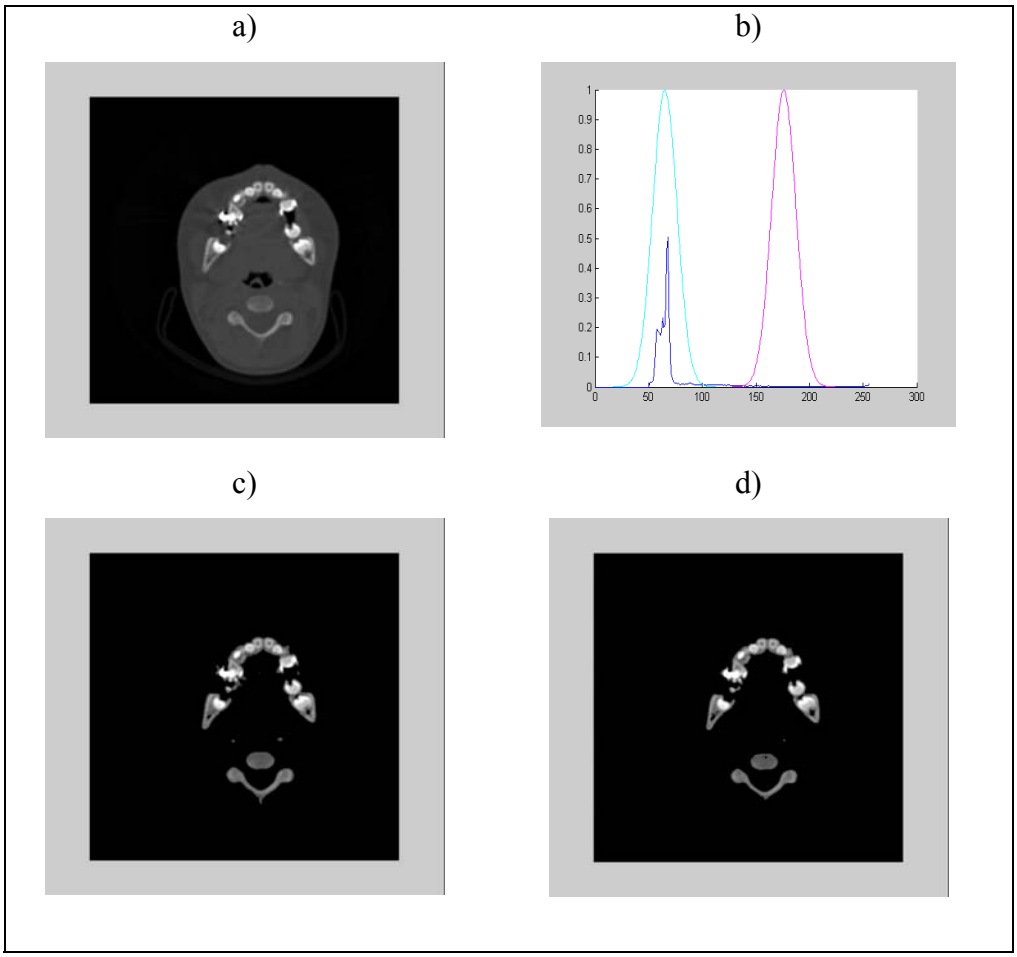

Figure 3.4 a)Original Image b) Histogram with gaussian curves and c) thresholded image using the EM algorithm with multiplier 0.5 d) and multiplier 1

The segmentation algorithm based on thresholding works well on separating the bone out of the image but it has some drawbacks. Because the algorithm only considers the intensity values and not the shape, even in the optimal threshold, there are some holes inside the bones and diffused edges are incorrectly segmented. As described above, spongy bone structure is texturized and the intensity values are not flat within the region of interest. The intensity level inside these bone structure is sometimes lower than the optimal threshold and this will result in gaps within the bone. Also the distribution of intensity values of bones, in general, does not fit to Gaussian distribution. In contrast to thresholding algorithm, the segmentation algorithm must consider both the spatial information as well as the intensity information.

3.3 Segmentation Using the Zero Crossings of Laplacian of Gaussian

Another type of segmentation method described in Section 2.3.3.3 which takes the spatial information into consideration, is the edge-based method. Edge based methods try to determine the places of rapid transition from one region to the other with different brightness or color value. Several edge detection operators are available. Canny, Sobel, Prewitt, Roberts, zero-crossings of Laplacian of Gaussian are the prominent ones that are used mostly.

The edge detector used in detection of bone boundaries should give closed contours and should not give local edges which are inside the bone. Such an operator satisfying these properties is the zero-crossings of Laplacian of Gaussian algorithm which is also known as Marr & Hildreth edge detector. In this algorithm, the image is filtered with a Laplacian of Gaussian operator whose standard deviation and size can be adjusted. Following the detection of zero-crossings of the gradient image, the edges in the image are found. Figure 3.5 shows the effect of the choice of different values of standard deviation on the edge image.

As seen from the images, increasing the value of the standard deviation of Gaussian filter decreases the numbers of closed contours but on the other hand, because of the smoothing property of the Gaussian filter, the locations of the edges are shifted. Also detection of bone contours after the edge detection operator is hard, since the intensity information is lost. Therefore, before applying the edge-based operator, a simple thresholding applied to image will get rid of most of the non-bone regions. For this purpose, EM algorithm described in Section 3.2 is used with a threshold multiplier constant less than one. This will eliminate most of the non-bone regions and the classification of bone and non-bone regions will be easier. Resulting set of closed contours can be filled and this can be used as a mask to segment the bone from image. Smaller regions can be eliminated according to their area.

The problem with edge-based methods is the differentiation between bone and non-bone regions because sometimes they are enclosed within the same contour. Edge based methods have a trade-off between the detail, the number of

contours and exact location of the boundaries. When accuracy of the boundaries is needed edge based method based on Marr-Hildreth operator is not suitable. Figure 3.6 shows the edge map after thresholding and applying the edge operator.

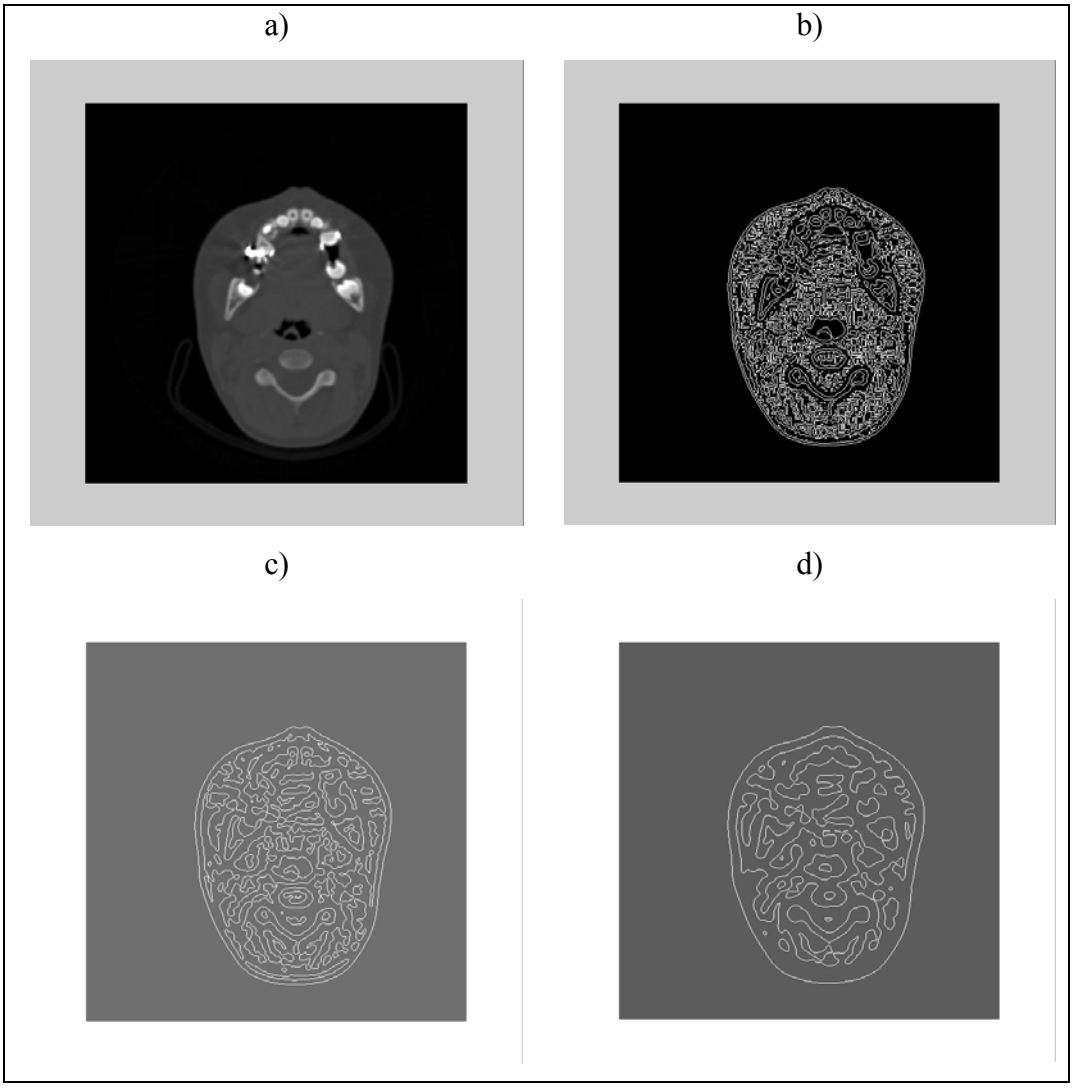

Figure 3.5 a) Original Image b) Zero-crossing operator applied to image standart deviation 1 c) standard deviation 3 and d) standard deviation 5

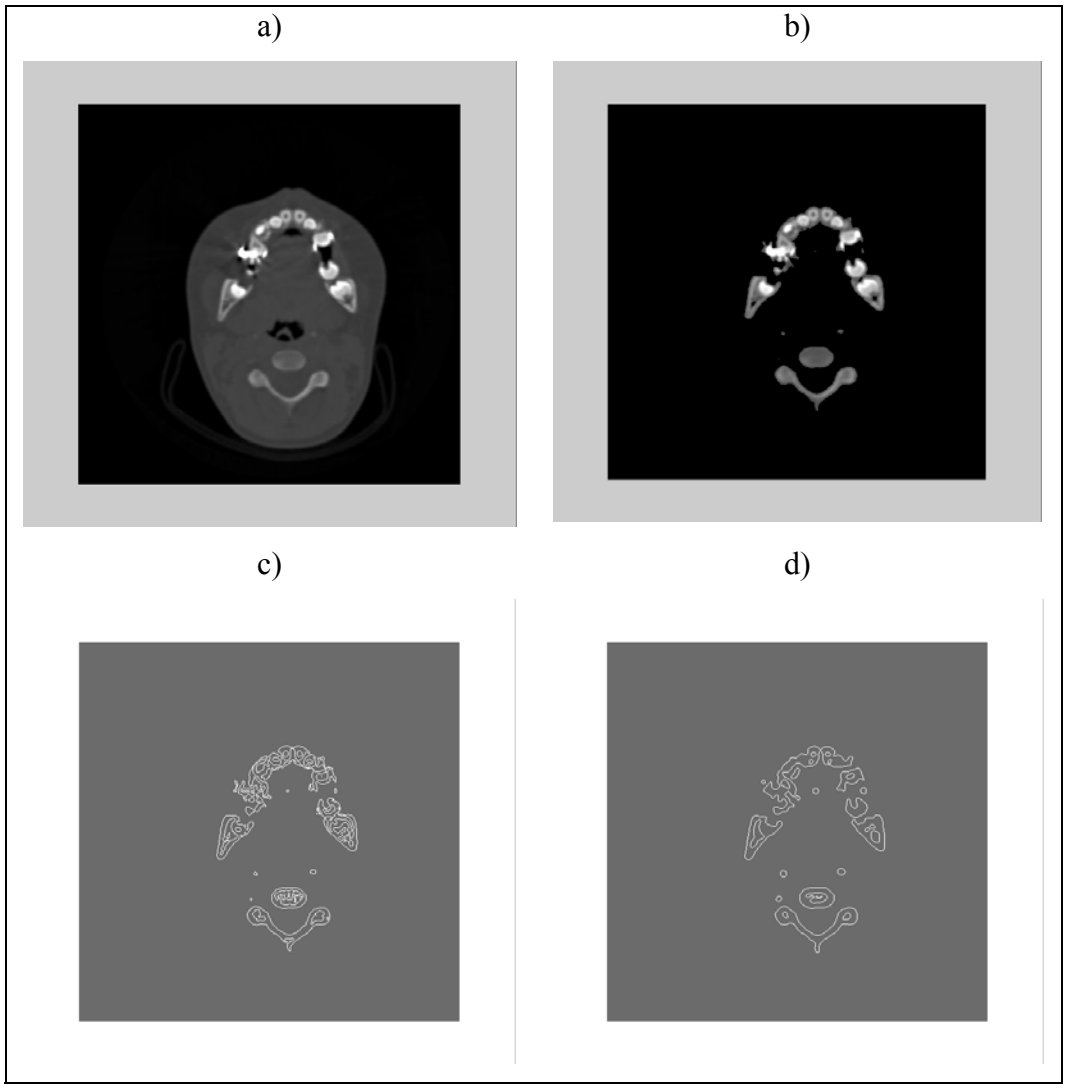

Figure 3.6 a) Original Image, b) Thresholded image c) Zero-crossing operator result with standard deviation 1 and d) with standard deviation 3

3.4 Region Based Methods for Segmentation of Bone

In addition to edge-based methods, there are region based segmentation methods. Region based methods split the image into regions and according to a homogeneity criterion the regions are combined to construct the required segmentation. The best techniques are those based on the assumption that the image can be partitioned into regions that can be modeled by simple planar and biquadratic functions. This type of region growing algorithm has the problem of

determination of initial partitioning size of the image. In order to increase the accuracy of the segmentation smaller patches can be selected. [11]

Region growing is a common method of grouping pixels within an image based on their eight-way connectivity and grayscale. A special case of region growing algorithm is the seeded region growing algorithm which is based on conventional postulate of region growing algorithms where the criteria of similarity of pixels is applied, but the mechanism of growing regions is closer to the watershed algorithm [36]. Instead of tuning homogeneity parameters, seeded region growing is controlled by choosing usually a small number of pixels, known as seeds. These seed pixels are chosen by the user according to his opinion as to what should be the regions to be extracted on the image. So, we start with the number of seeds which have been grouped into n sets: A_1, A_2, ..., A_n. Sometimes individual sets could consist of single point. At each step of the algorithm we add one pixel to some of the sets A_i, i =1,..., n. Let T be the set of unallocated pixels which border at least one of the regions, which means that T is the set of all pixels which are on the borders of regions formed up to now:

$$T = \left\{ x \notin \bigcup_{i=1}^{n} A_i \,\middle|\, N(x) \bigcap \bigcup_{i=1}^{n} A_i \neq 0 \right\} \tag{3.1}$$

where N(x) is the set of immediate neighbors of the pixel x.

In our case, we consider 8-connectivity which contains all the neighbors of the pixel. Each step in the algorithm takes one pixel from the T set and adds it to one of the regions with which neighbors N(x) of the pixel intersect, actually label it with the label of that region. Then all pixels from N(x) are examined and distances from their neighboring regions are calculated. According to these distances, pixels are put into T set in increasing order. The distance measure simply indicates how far the intensity of the regarded pixel is away from the mean intensity value of growing regions. It is defined as:

$$\delta(x) = g(x) - mean_{y \in A_{i(x)}}\left[g(y)\right] \tag{3.2}$$

If pixel under consideration has more than one neighboring regions with different running means, region which will include the pixel has to be decided. For this purpose, distances of the pixel from all its neighboring regions are calculated. Decision is done by comparing all of the distances and finding their minimum. Minimum distance from the neighboring regions of the pixel is described as:

$$\delta(z) = \min_{x \in T}\{\delta(x)\} \tag{3.3}$$

If all the neighbors of the pixel under interest belong to same region as found in the comparison test, then the pixel is included in that region. If some of its neighbors belong to a different region, the pixel is labeled as boundary pixel which separates the regions.

In the implementation of seeded region growing, simple sorted list is used as the data structure for storing elements of set T. Sorted list used in the region growing algorithm is called sequentially sorted list (SSL) in which only coordinates of the pixels and distance delta from their neighboring regions are stored. When a new pixel is considered, it is taken from the SSL and processed. When we add a new pixel to the list we have to add it according to its distances from the neighboring regions. The pseudo code of seeded region growing algorithm is given below:

Initialization:

Label seed points according to their initial grouping.

Put neighbors of seed points (the initial T) in the SSL.

Region Growing:

While *SSL is not empty* **do**

Remove first pixel *y* from the SSL.

Test the neighbors of this point:

if all neighbors of y which are already labeled (other than boundary label) have the same label **than**

Set *y* to this label.

Update running mean of corresponding region.

Add neighbors of *y* which are neither already set nor already in the

SSL to the SSL according to their value of delta

else

Flag *y* with the boundary label.

Figure 3.7 shows the result of seeded region growing applied to a part of image. In this example three seeds are selected. One of them corresponds to the non-bone region, one of them for the teeth and the last is used for the bones. The resultant segmentation has three regions. Teeth and bone region can be combined for visualization purposes. MATLAB code of implemented seeded region growing algorithm is given in Appendix A.

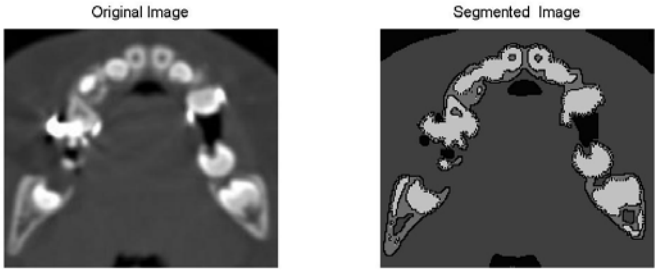

Figure 3.7 Result of seeded region growing algorithm

Choosing the seeds for the region growing algorithm is user dependent. Also, since the values of previous entries of SSL are not updated to reflect their differences from the new region, mean value and the final segmentation result is dependent on the way seed pixels are selected. An improved version of the SRG algorithm [37] eliminates pixel order dependencies while trading off from execution speed.

When we have two regions to segment, bone and non-bone, automatic segmentation of CT slices becomes easier. Bright clumps pixel are selected to be seeds and the region growing algorithm is based on these seeds. The selection of brightness is dependent on a threshold; all pixels having intensities higher than this threshold are assigned as seeds. Using this property, a simplified version of the seeded region growing algorithm can be used to segment the head CT slice. The input parameters to the simplified algorithm are seed level and a threshold value. The region growing algorithm used here, searches all pixels within the image and their neighbors. If a pixel intensity value is larger than seed level it is immediately assigned as bone region. If it is less than seed level, all neighbors of the pixel are searched and intensity differences between these pixels and the running mean are calculated. If the smallest difference is less than the threshold, that pixel is assigned as a bone pixel and growing process continues on searching the neighbors of this previously assigned pixel. All pixels that are assigned as bone or non-bone are stored in order to prevent the algorithm to reprocess the same pixel. The critical issue here is the selection of seed level and threshold parameters. In order to automate the method, threshold and seed level gathered from image content can be used. Figure 3.8 shows the result of the simplified region growing algorithm applied to image. MATLAB code of simplified region growing algorithm is given in Appendix B.

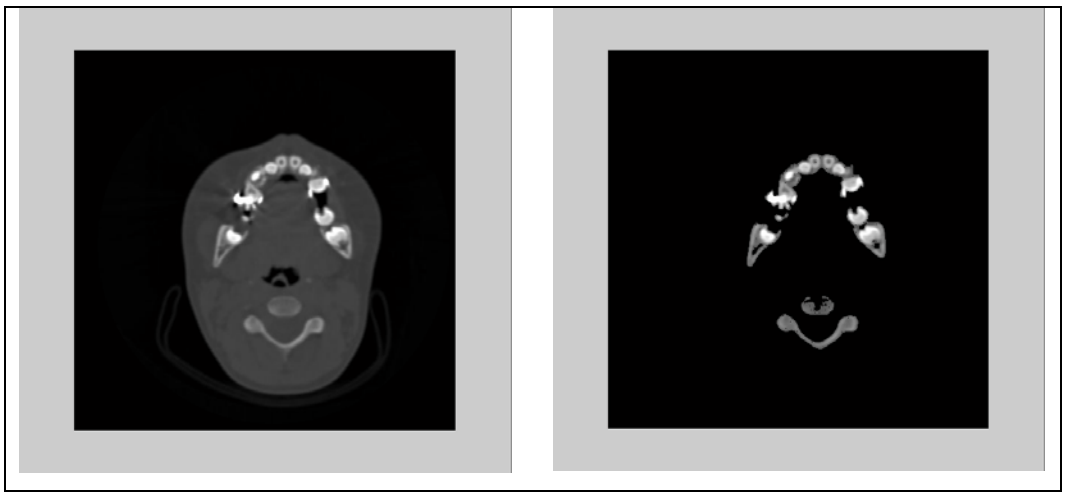

Figure 3.8 Original Image and result of region growing algorithm

In simplified region growing algorithm, segmentation is sensitive to threshold selection. Choosing a small threshold will result in thinned regions and choosing a large threshold will enlarge bone tissue regions and merge the unconnected regions such as teeth. Also different regions have different intensity profiles and using a global running mean results in some regions to be considered as non-bone.

3.5 Hybrid Algorithm Developed for the Extraction of Bone Tissue

All of the segmentation methods considered above have some drawbacks and finding a best operator satisfying the requirements of a segmentation task is hard. According to the requirements of a segmentation task, some of the operators can be combined and subtasks can be assigned to them in several stages of segmentation. Such methods are called hybrid methods. Using the discussed segmentation operators, a hybrid method is developed for our task. In the first step, the hybrid method uses the thresholding algorithm to presegment the image to decrease the size of the non-bone regions. Thresholding divides the image into several parts that can be labeled by using the connected component labeling algorithm. Connected component labeling algorithm finds all connected components in an image and assigns a unique label to all points in the same component. Also the bounding box enclosing these parts are detected to create sub images whose location in the original image is known. Simplified region growing algorithm is applied to each of these regions having an area larger than a predefined level. All other regions not satisfying the minimum area constraint are deleted. The results of the region growing on each segment are collected to form the output image. Seed level is chosen as the mean value of the sub image and threshold is the standard deviation within that part of the image. Figure 3.9 shows the result of the hybrid algorithm. Hybrid algorithm segments teeth better than the algorithms discussed above. Also applying the region growing algorithm on the different components of the image prohibits an overall running mean exceeding some of the regions average mean value which prevents holes within bone tissues. MATLAB code of the hybrid segmentation algorithm is given in Appendix C.

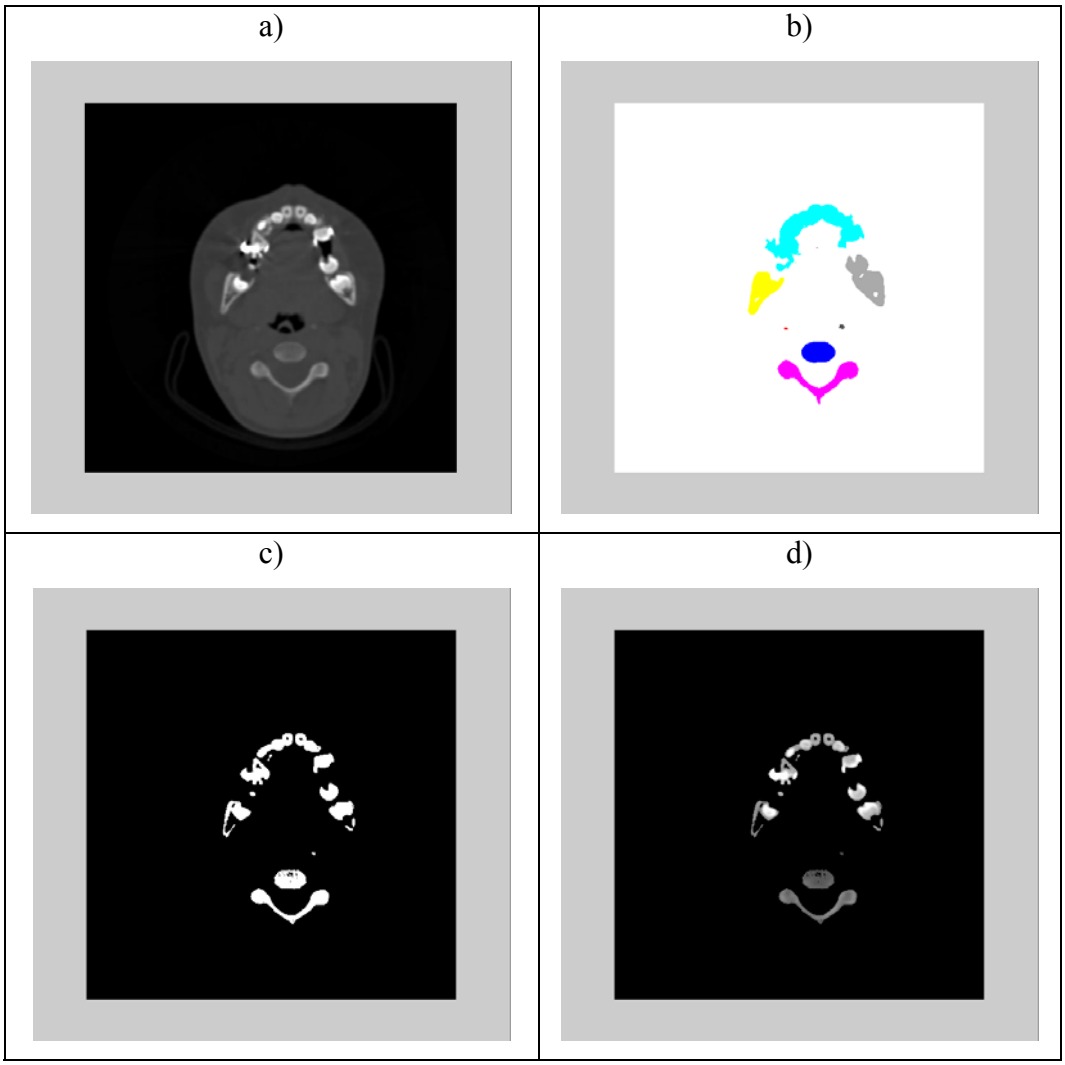

Figure 3.9 a) Original Image, b) Connected component labeling of the image, c) Mask after hybrid algorithm and resultant d) segmented Image

3.6 Visualizations following the hybrid segmentation algorithm

In this section, visualizations of the segmented bone structure of head CT slices are shown. Head CT slices are segmented by the hybrid segmentation algorithm that is described in Section 3.5. These visualizations are realized using Visualization Library of MATLAB version 6.5**.** MATLAB code used in obtaining the visualizations is provided in Appendix D. Head CT images of two patients who are subjected to orthodontic treatment are used in obtaining these visualizations.

Figure 3.10 shows several visualizations of the patient prior to segmentation. Figure 3.11 shows segmented visualizations of the same views. Figure 3.12 and 3.13 belong to another patient.

Visualizations of the craniofacial bone show the abnormalities in the jaw and teeth in both patients. Combining these visualizations with the visualizations obtained during the treatment, a planning strategy on how to correct these abnormalities can be formed.

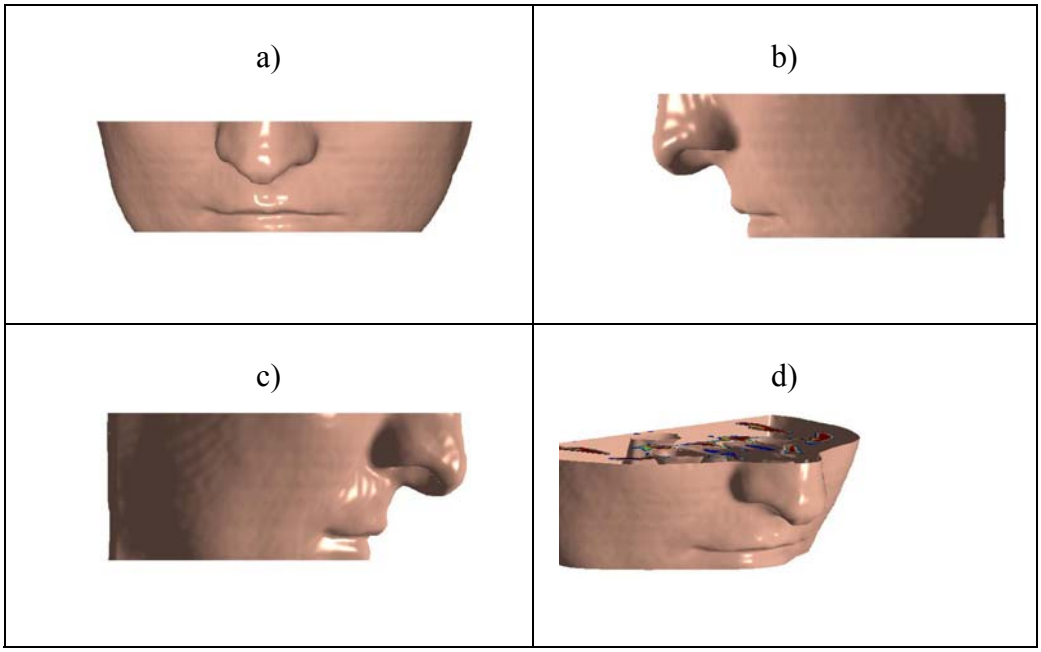

Figure 3.10 a) Front view b) Right view c) Left view and d) profile view of first patient

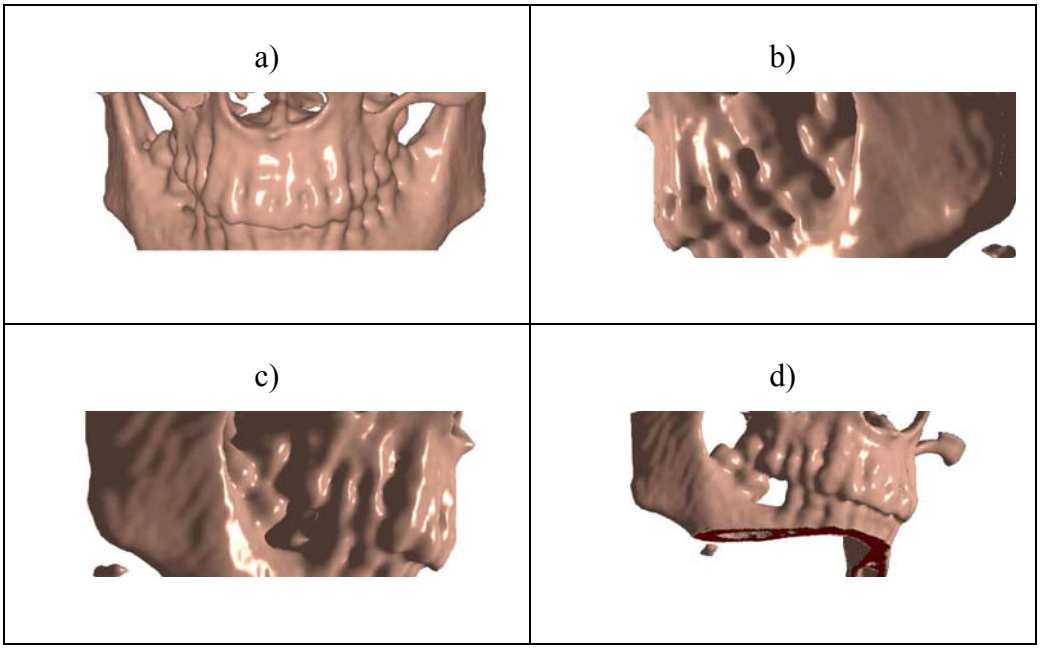

Figure 3.11 a) Front view b) Right view c) Left view and d) profile view of segmented bones of second patient

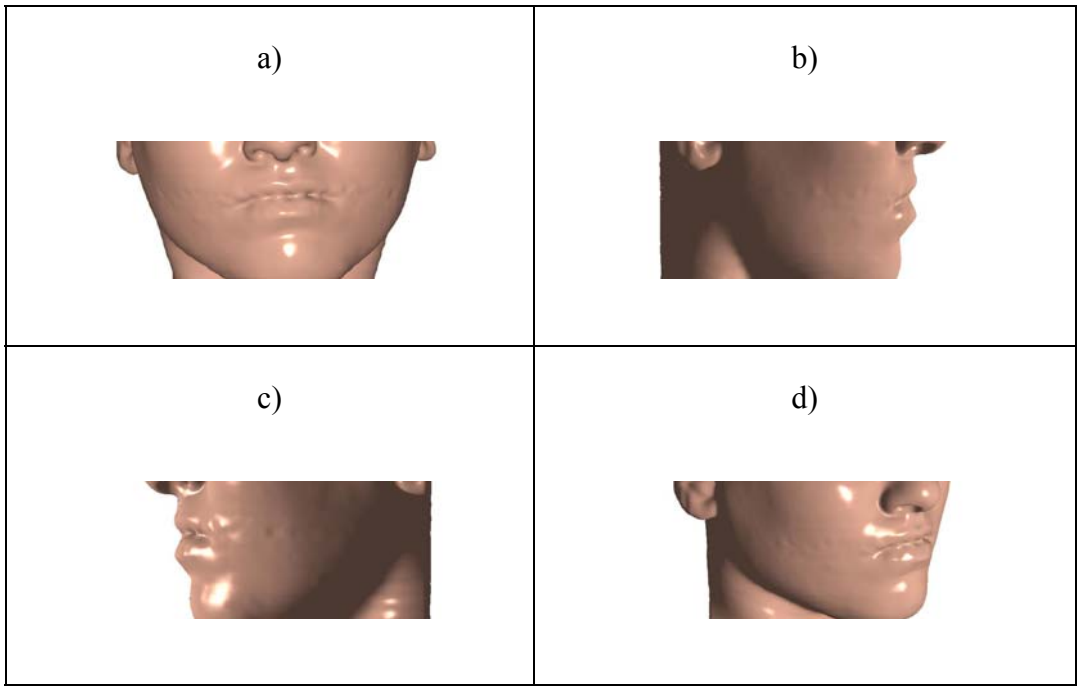

Figure 3.12 a) Front view b) Right view c) Left view and d) profile view of second patient

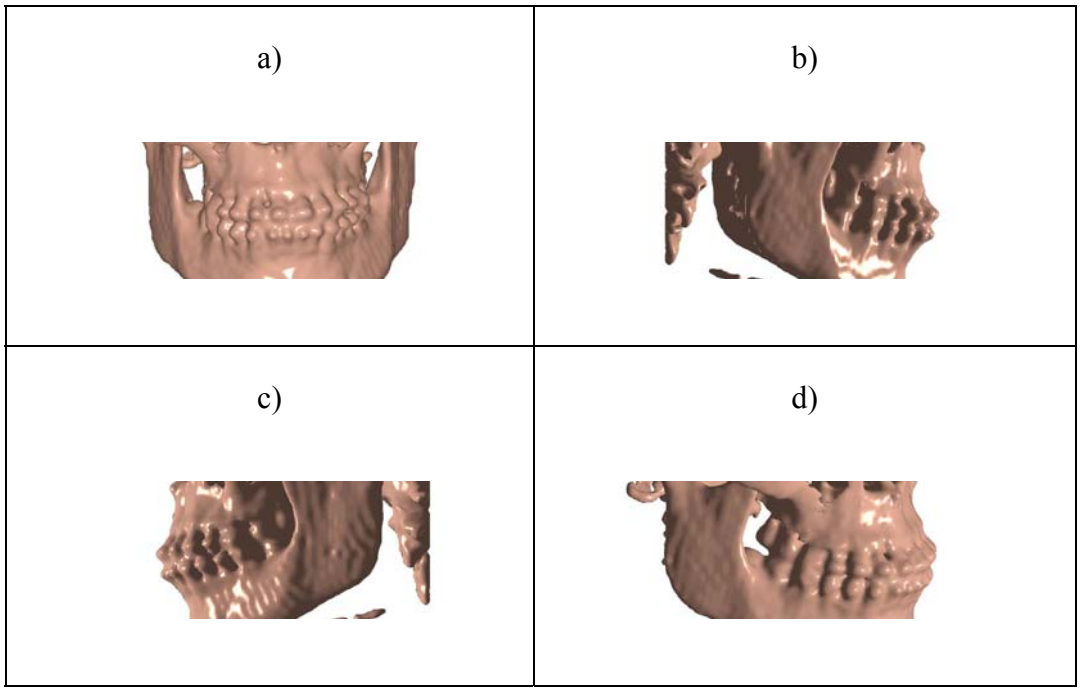

Figure 3.13 a) Front view b) Right view c) Left view and d) profile view of segmented bones of second patient

CHAPTER 4

4. AUTOMATIC REGISTRATION OF MEDICAL IMAGES AND DATASETS

4.1 Introduction

Registration is indispensable when we have to track changes of images taken at different times, from different viewpoints or by different sensors. Registration methods can be viewed as different combinations of choices for a feature space, a search space, a search strategy and a similarity metric. Feature space extracts valuable information from different datasets that will be used for matching. Search space consists of available transformations capable of registering these datasets. Search strategy decides on how to choose the next transformation in order to find the optimal transformation that aligns two different datasets. Similarity metric compares these transformations with each other and the registration continues until a satisfactory small metric is found. Major areas that use registration heavily are computer vision and pattern recognition, medical image analysis and remotely sensed data processing [19]. In the field of medical imaging, changes in the position of the patient over a period of time, position of the imaging machine, change in the viewpoint and changes within some portions of the image make the registration a necessary step in image analysis.

According to [38] medical image registration methods can be classified into several categories. These categories include registration according to dimensionality, nature of registration basis, nature of transformation, domain of transformation, interaction, optimization procedure, modalities involved, subject and object. These categories are further divided into subcategories and references are given for all of them in [38].

In our case, registration of two dimensional images and three dimensional datasets that belong to head are considered. CT scans of patient that are subjected to orthodontic treatment are taken before the start of treatment and at the end of treatment. Since a time has to pass during the treatment, the registration can be characterized as time-series registration. In addition to this, the datasets to be registered taken at different times have the same modality, so our registration purpose also falls into monomodal registration category. As described in 2.3.4, there are two different subcategories for the nature of registration basis. These include extrinsic and intrinsic registration. The CT images provided by the Orthodontics department of Hacettepe University are taken without using any frame that will make the registration easier, therefore the registration has to be intrinsic.

Before starting a registration task, type of the transformation involved in the images should be decided. An image coordinate transformation can be rigid, when only translations and rotations are involved. If parallel lines are mapped on parallel lines, transformation is called affine. If lines are mapped onto lines, it is called projective. If lines are mapped onto curves transformation is elastic [19]. Rigid registration is the simplest type among these registrations.

Registration of head CT scans taken at different times is a rigid registration problem [39] when some precautions are taken before and during the scanning. The patient should not move during the scanning process for an accurate registration. Gantry tilt of the detector should be set to zero. Otherwise the tilt should be corrected before or during registration which will cause substantial errors. Same conditions in the first scanning should be provided in the second scanning. Also as in the case of segmentation, registration should be automatic in order to prevent errors that belong to user interaction. Assuming all of these conditions are met, accuracy of registration is left to the algorithm.

Throughout this chapter of the thesis, automatic registration of two dimensional images and three dimensional datasets are investigated. Before

proceeding, brief information about rigid transformation and iterative closest point algorithm are given.

4.2 Rigid Transformation

An image coordinate system is called rigid, when only translations and rotations are allowed. Rigid transformations account for object and sensor movement in which objects in the images retain their original shape and size. A rigid-body transformation is composed of a combination of a rotation, a translation and a scale change. Two dimensional rigid transformations can be described by:

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \end{pmatrix} + s \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \tag{4.1}$$

This can be written as

$$\overline{p}_2 = \bar{t} + sR\overline{p_1} \ , \tag{4.2}$$

Where $\overline{p_1}$ and $\overline{p_2}$ are two coordinate vectors of the two images, $\bar{t}$ is the translation vector, s is the scale factor and R is the rotation matrix.

A rigid three dimensional transformation can be described using a single constant matrix equation. The matrix equation can be written in an open form as:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ 1 \end{pmatrix} = \left(\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} \tag{4.3}$$

where T is an arbitrary translation vector and R is a 3x3 rotation matrix defined by [11]:

$$R = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix}. \tag{4.4}$$

Using rotation ω about the *x* axis, rotation φ about the new *y* axis and rotation κ about the new *z* axis, the individual components of the rotation matrix can be written as [11]:

$$r_{xx} = \cos\phi\cos\kappa \tag{4.5}$$

$$r_{xy} = \sin\omega\sin\phi\cos\kappa + \cos\omega\sin\kappa \tag{4.6}$$

$$r_{xz} = -\cos\omega\sin\phi\cos\kappa + \sin\omega\sin\kappa \tag{4.7}$$

$$r_{yx} = -\cos\phi\sin\kappa \tag{4.8}$$

$$r_{yy} = -\sin\omega\sin\phi\sin\kappa + \cos\omega\cos\kappa \tag{4.9}$$

$$r_{yz} = \cos\omega\sin\phi\sin\kappa + \sin\omega\cos\kappa \tag{4.10}$$

$$r_{zx} = \sin\phi \tag{4.11}$$

$$r_{zy} = -\sin\omega\cos\phi \tag{4.12}$$

$$r_{zz} = \cos\omega\cos\phi \tag{4.13}$$

Solving for the individual angles that correspond to rotation is algorithmically cumbersome. Instead of using the angles, the rotation matrix can be described by quaternions.

A quaternion is a four-element vector,

$$q = (q_0, q_{1,} q_{2,} q_3) \tag{4.14}$$

having the property,

$$q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1. \tag{4.15}$$

Quaternions encode the rotation and has some properties that make them suitable while developing an algorithm.

In terms of quaternions, the rotation angle is defined by [13]:

$$R(q)=\begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2+q_2^2-q_1^2-q_3^2 & 2(q_2q_3-q_0q_2) \\ 2(q_1q_3+q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2+q_3^2-q_1^2-q_2^2 \end{bmatrix} \tag{4.16}$$

A more convenient way to write the overall registration state vector is:

$$\vec{q}=\left[\overrightarrow{q_R}\mid\overrightarrow{q_T}\right]^T \tag{4.17}$$

where

$$\overrightarrow{q_T}=\begin{bmatrix} q_4 & q_5 & q_6 \end{bmatrix}^T \tag{4.18}$$

is the translation vector.

Once the unit quaternions are found, conversion between quaternions and the rotation angles is easy using (4-5) through (4-12). The algorithmic advantages of the unit quaternions are used in the iterative closest point algorithm which is described in the following section.

4.3 Iterative Closest Point Algorithm (ICP)

Given two sets of partially overlapping range data and an initial estimate of their relative positions, iterative closest point algorithm is used to register the data sets by improving the position and orientation estimate. First introduced by Besl and McKay [40], iterative closest point is an essential step in model building, dimensional inspection, and numerous applications of range data processing.

Iterative closest point algorithm can be used for registering point sets, line segment sets, implicit curves, parametric curves, triangle sets, implicit surfaces

and parametric surfaces. Although main purpose for ICP is registration of three dimensional shapes, it covers the two dimensional registration.

At each step of iteration in ICP, correspondences are calculated between the two data sets, and using these, a transformation which minimizes the mean square error of the correspondences is calculated. If the mean square error falls below a threshold or the predefined number of iterations is exceeded, the algorithm stops.

Before going into details of the algorithm, simple definitions will be given. The Euclidian distance definition in a three dimensional space used in the algorithm is,

$$d(\overrightarrow{r_1}, \overrightarrow{r_2}) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{4.19}$$

where

$$\vec{r}_1 = (x_1, y_1, z_1) \text{ and } \vec{r}_2 = (x_2, y_2, z_2). \tag{4.20}$$

For two dimensional datasets, z axis components are not included in the equation.

Considering a data shape P and a model shape X which are to be aligned, each having N_p and N_x number of points. For all data points in P, distances from the model X are calculated. Among these distances, minimum distance between the point and the model is taken as the distance metric. This distance metric is denoted by,

$$d(\vec{p}, X) = \min_{\vec{x} \in X} \left\| \vec{x} - \vec{p} \right\| \tag{4.21}$$

Letting Y denote the resulting set of closest points and C denote the closest point operator:

$$Y = C(P, X)\,. \tag{4.22}$$

Given the resultant corresponding point set Y, the least squares registration is computed using,

$$(\vec{q}, d) = Q(P, Y)\,, \tag{4.23}$$

where Q is the least squares quaternion operator defined in [40].

Using the correspondent data points and model points, cross covariance of the sets P and X are calculated, which is then used to form a symmetric 4x4 matrix. Eigenvalues of this symmetric 4x4 matrix gives the unit quaternions. After finding the quaternions, optimal translation vector is calculated. The data shape points are updated via $P = \vec{q}(P)$ where $\vec{q}$ is the overall registration state vector.

If we have measured data point set P with N_p points and the model data point set with N_x number of points, the implementation of the iterative closest point algorithm is as follows:

Initialization:

$P_0 = P$

$\overrightarrow{q_0} = [1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T$

$k = 0$

Iteration:

Compute the closest points: $Y_k = C(P_k, X)$.

Compute the registration : $(\overrightarrow{q_k}, d_k) = Q(P_0, Y_k)$

Apply the registration : $P_{k+1} = \overrightarrow{q_k}(P_0)$

Terminate the iteration if $d_{k-1} - d_k < \tau$ where τ is a preset threshold.

4.4 Two dimensional automatic medical image registration

Two dimensional medical image registration is necessary when we need to register two dimensional images of a body part taken at different times. These images can be taken by conventional X-ray machines or CT machines. However two dimensional registration of CT slices taken at different times is not possible, because of the change in the coordinate system. They need to be registered in a three dimensional space. Assuming the same viewpoint is provided in a two dimensional time-series image, registration consists of finding the rotation angles, scale change and translation vector.

Automatic registration of two dimensional medical images requires finding correspondent points in images that we want to register without any user interaction Following the point extraction, extracted points should be registered using a registration algorithm.

[41] describes a method for automatically finding correspondent points in medical images. The algorithm is divided into six steps:

Step 1: A similarity measure is defined that will be used in the next step of the algorithm. Only requirement for the similarity measure is that, similarity values must lie between 0 and 1; 0 meaning identical subimages whereas 1 means the largest dissimilarity between images.

Step 2: For every pixel *(x*,y) in the reference image *I* the similarities *Si*m((*x*,y),(*x*+*k*,*y*+l)) between a circular area of radius *r* centered at *(x*,y) and a circular area of equal size centered at *(x*+*k*,*y*+l); such that $k^2 + l^2 < R^2$, is calculated. Sim((x,y),(x+k,y+l)) is defined as:

$$Sim((x,y),(x+k,y+l)) = \frac{1}{(G-1)N} \sum_{\substack{\forall i,j; \\ i^2+j^2 \le r^2}} \left| I(x+i,y+j) - I(x+k+i,y+l+j) \right| \tag{4.24}$$

G being the number of gray levels, N is the number of pixels defined by a circular region r. This step of the algorithm maps a two dimensional image to another two dimensional image, which the authors call "Similarity Surface" centered at pixel (x,y). Shape of underlying similarity surface gives an indication of how the pixel at that coordinate is distinctive from other pixels. Multiplication of similarity values at only 16 sites shown in Figure 4.1 is a fast method to determine the distinctiveness of a pixel. Distinctiveness is defined as,

$$D(x,y) = \prod_{k=1}^{16} Sim((x,y),(x^k,y^k)), \tag{4.25}$$

where (x^k,y^k) are the pixels around the circular region having a radius of 3 which is shown in Figure 4.1.

Distinctiveness is also a two dimensional image which can be displayed for visualization purposes. Distinctiveness map of a CT slice is shown in Figure 4.2. MATLAB code used in obtaining the distinctiveness map is given in Appendix E.

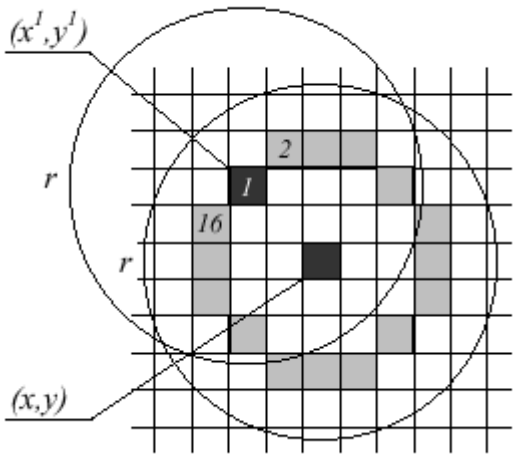

Figure 4.1 Digital circle with radius r=3 used for calculating the distinctiveness of a pixel

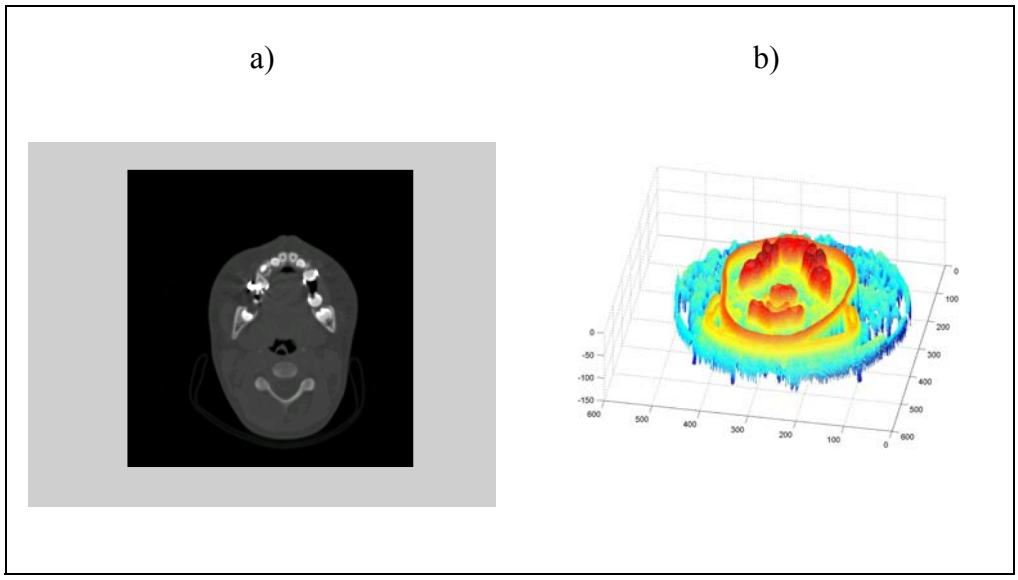

Figure 4.2 a) CT image and b) its three dimensional distinctiveness map

Step 3: A predefined number of pixels with highest distinctiveness are chosen and among these pixels satisfying a separation of d_{min} between them are selected such that first pixel has the highest distinctiveness in the image, following pixel having the highest distinctiveness and a separation d_{min} between already selected pixels is chosen. This process continues until the set of pixels achieve the predefined number.

Step 4: Step 4 is the pixel location correction step. At the end of step 3, we have two sets of pixels with highest distinctiveness and a separation of d_{min}.. Each candidate point in the second image is searched around the first image candidate points. Search area is defined by a circle with radius R. Corrected pixel location is found by choosing the pixel with smallest similarity value.

Step 5: In step 5**,** first, invalid point pairs are removed by setting a threshold for the similarity between the pairs. Second, an assumed transformation is applied to first image's pixels and the geometric misfit is calculated for each pixel. If this geometric misfit is greater than a threshold, these points are removed. This process is repeated as many times until the entire misfit is smaller than the threshold.

Step 6: Weights are assigned to point pairs using the distinctiveness, similarity and the misfit values. Authors advise to use these weighted point pairs while approximating the geometric transformation.

There are many variables included in the algorithm such as the circle radius size, thresholds, number of distinct points. Once these values are set for a given two dimensional registration task, following registrations of the same type of images will be automatic. Also this algorithm is said to be effective while registering monomodal two dimensional images with a linear functional relationship between their intensity values.

For the images we consider, type of transformation involved in images is assumed to be rigid. Also instead of applying predetermined transformation and calculating the misfit for each point pair, iterative closest algorithm can be applied in the fifth stage of the algorithm. ICP finds a transformation such that the overall sum of misfit is minimized. The subjective function that is to be minimized can be formed using the weights assigned in sixth step. MATLAB code of two dimensional iterative closest point is given in Appendix F.

The modified automatic registration algorithm proposed above is tested using a processed CT topogram image and the synthetic image obtained by rotating this image with a given angle. The topogram image has a white background. This background is subtracted from the image using simple morphological operations. Contrast of the output image is adjusted to make bone regions clear. This image is rotated by a given predefined angle. This rotation models the patients head position change from one image to another image. While forming the rotated image, bicubic interpolation is used. Background subtracted image and 17 degrees rotated synthetic image are shown in Figure 4.4. 40 points extracted on both images with highest distinctiveness using a minimum distance 15 are also shown in Figure 4.4. After examining the points carefully, it is seen that, some of the points in the first image are not mapped exactly to the second

image. These pixels have to be corrected using template matching. ICP algorithm is run using the extracted points and recovered angle is found as 15.49 degrees.

The angle is recovered with a slightly large error and the reason for this error mainly depends on the rounding effect of the interpolation process. This effect can be overcome by applying the reverse rotation using the angle found above and iterating the same steps until angle increment or misfit between features is below a threshold value. Figure 4.3 shows the resultant RMS distance error between features in two images versus iteration number graph.

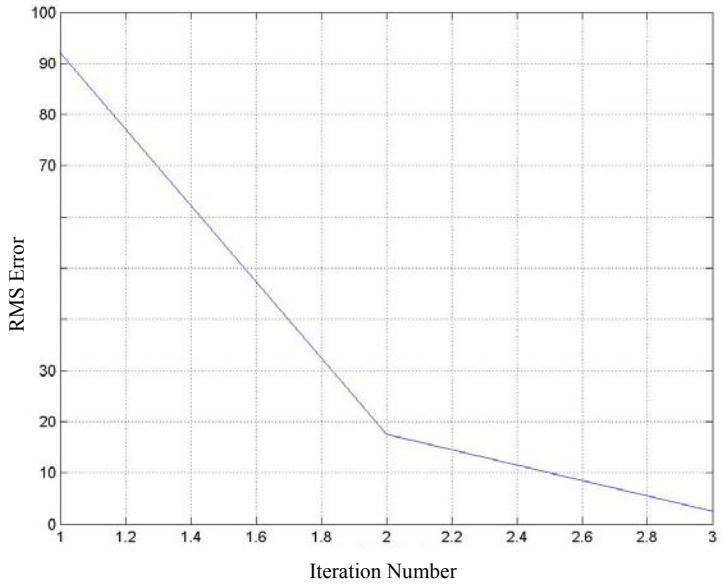

Figure 4.3 RMS Error vs. Iteration Number for 2-D Automatic Image Registration

It is seen that after 3 iterations RMS error drops down to 2.531 from 92.09 which allows us to recover the initial angle with less than %1 percent resolution. This algorithm is applied to the topogram images provided from the Orthodontics department shown in Figure 4.5. Area of interest is selected to confine the search area for the algorithm which is the head in our case. Figure 4.6 shows the result of the registration algorithm.

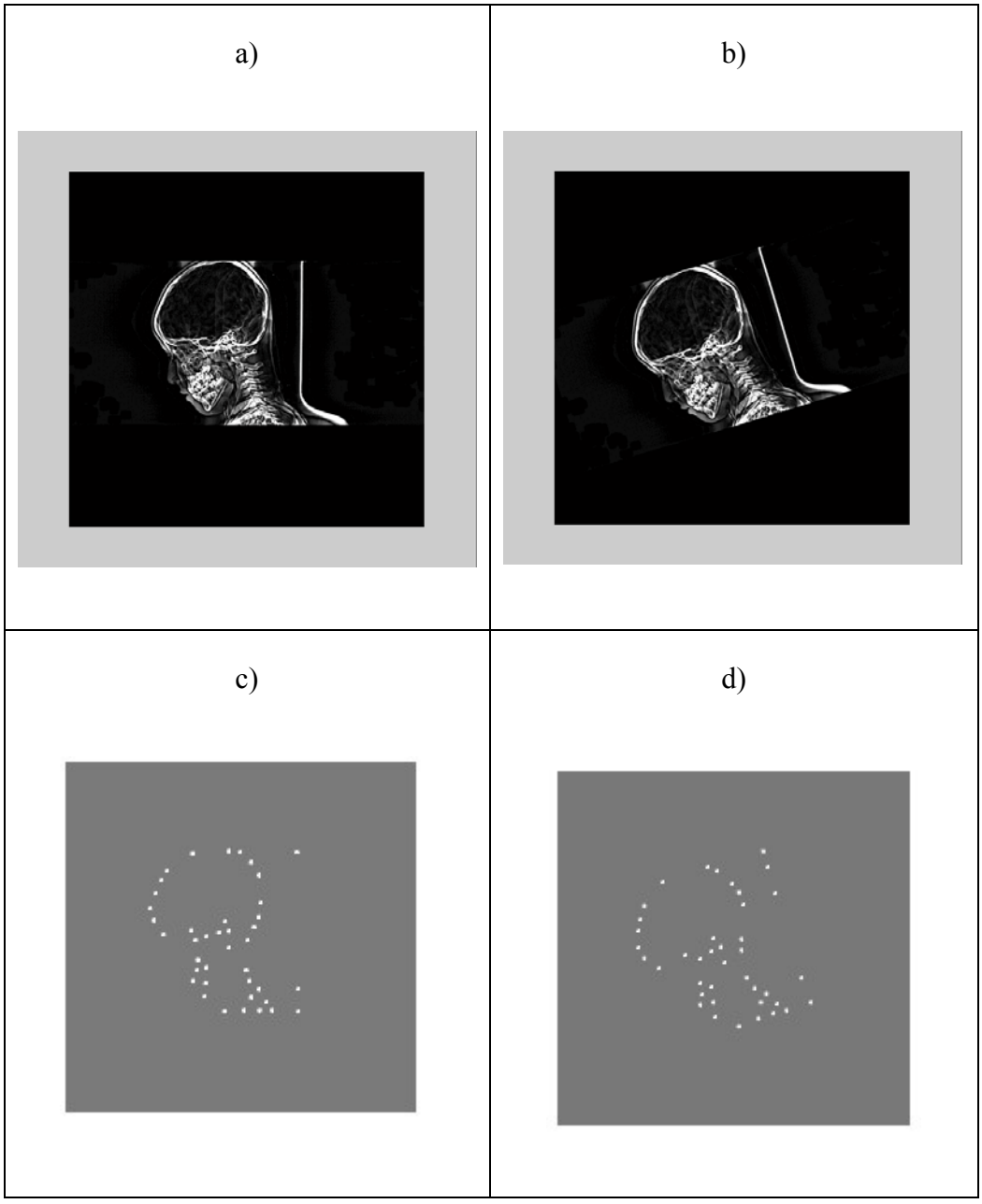

Figure 4.4 a) Original processed image b)17 degrees rotated image c) Extracted points from original image d) Extracted Points from rotated image

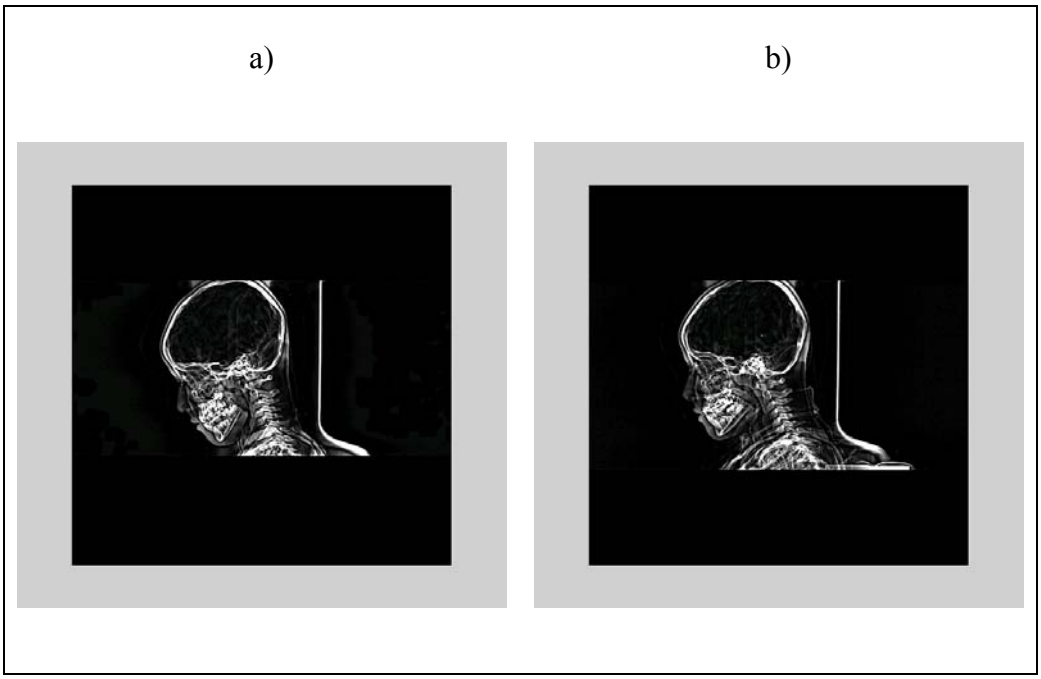

Figure 4.5 Real Images used in testing the automatic registration algorithm
a) Before treatment b) During treatment

The method for two dimensional automatic registration is as follows,

1. Number of points, the separation of points that will be used in registration process are determined. 40 points with minimum distance 15 can be the starting point and these parameters may be changed if needed for better performance.
2. Features are extracted from both images and the registration method described above is applied.
3. From the registered image and initial image, anatomical landmark points may be selected and using these points the amount of displacement at those points may be calculated.

4.5 Three dimensional automatic medical image registration

Registration of CT datasets formed by several slices taken at different time intervals is a three dimensional registration task. Slice positions and slice contents may change because of rotation, scaling and translation from one dataset to other.

Achieving the exact patient position may not be possible even when using a fixture.

Automatic registration of three dimensional datasets requires the extraction of features in a three dimensional space and finding the optimal transformation between them. Iterative closest point is used in some clinical cases and found to be efficient in terms of registration error [27], [28].

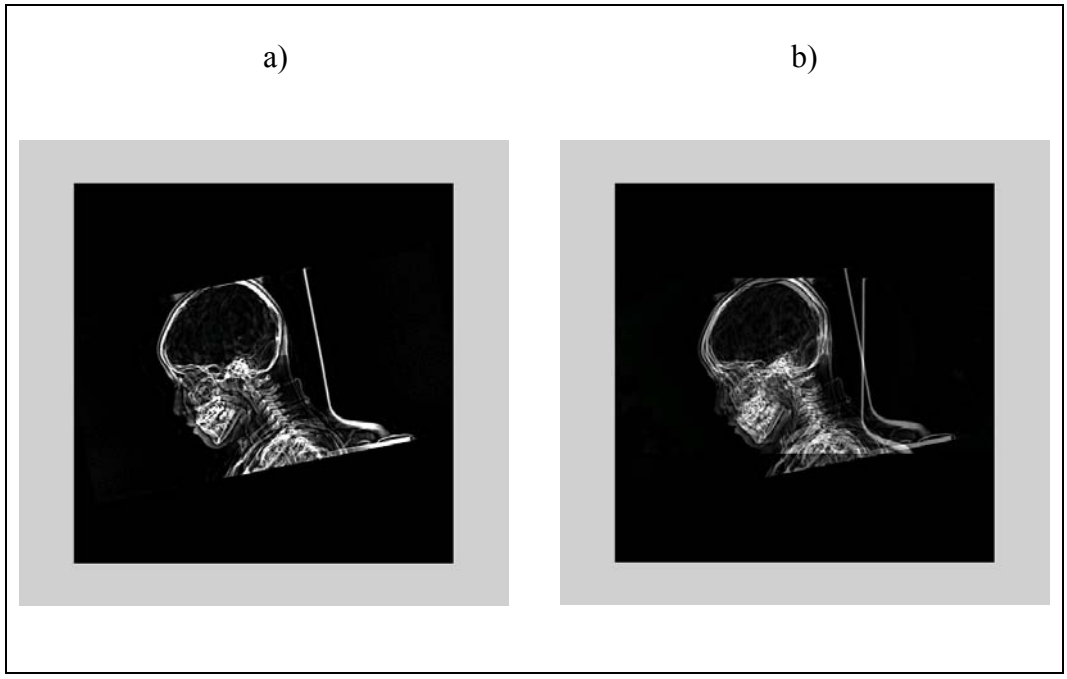

Figure 4.6 a) Registered Image and b) registered image superimposed on the first image

In our case, ICP algorithm is tested with a real dataset and a synthetic dataset formed by rotating this dataset using bilinear interpolation. MATLAB code used in rotation and bilinear interpolation is given in Appendix G. Rotation models the change of patient position or change in the viewpoint. Figure 4.7 shows the same numbered slice in the original and the rotated dataset with rotation angles -2.345 around x axis, 4.345 around y axis and 1.879 around z axis. It is clearly seen that rotation changes the image content and finding the back transformation is possible only when three dimensional dataset is considered as a whole.

Upon registering two dataset, candidate point pairs for registration are selected from the isosurface vertices. Isosurface is a combination of surfaces having a level that is equal to a given threshold. In this sense, isosurface level extraction is similar to global thresholding. Figure 4.8 shows the vertices of the patches found from the original and the rotated dataset. Pathes corresponding to original set consists of 177711 points and patches that correspond to rotated dataset consists of 165877 points. Threshold is intuitively selected as 180 such that most of the bone region is segmented out of the dataset. Vertices of the patches formed by isosurface extraction are reduced to five percent of its original value in one dataset to increase the speed of the search. This reduction eliminates most of the irrelevant point pairs. Since ICP algorithm considers a given number of points which is smaller than the whole number of data points, using relevant points on each iteration allows the algorithm to approach the final transformation more quickly and accurately. Figure 4.9 shows the reduced patch vertices in the rotated dataset that consist of 8455 points.

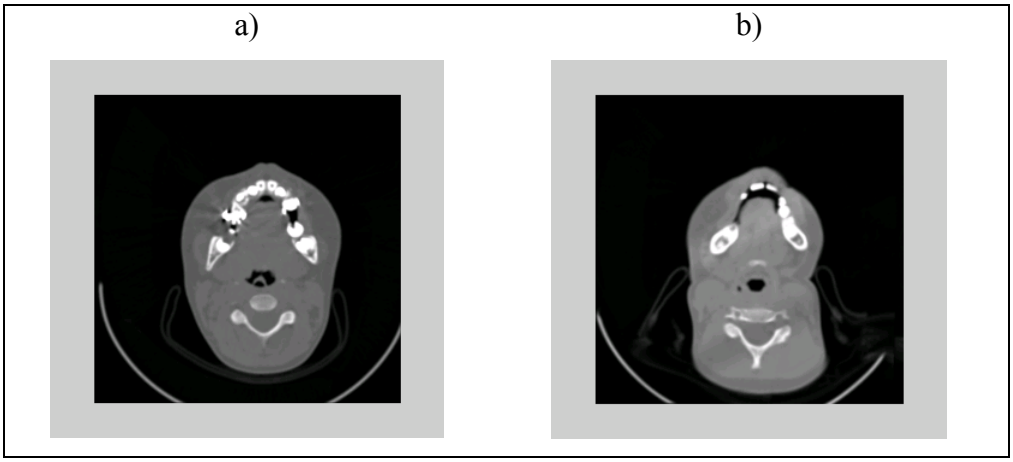

Figure 4.7 a) Slice in first dataset and b) same slice location in the rotated datasets

Patch vertices in first dataset and reduced patch vertices in the second set are given to the ICP algorithm. In each iteration, ICP algorithm rotates the reduced points such that overall misfit between the closest points in first set is reduced. ICP algorithm outputs the successively transformed dataset. Rotation angles are found by comparing the rotated set with the initial set given to ICP. Figure 4.10 shows

the RMS distance versus iteration number graph while ICP is running to register the original dataset with 4 degree rotated set in each axis. RMS distance is calculated by using the actual points and the successively rotated points and it is a point to point distance metric. It is seen that after a few iterations RMS error drops below 0.5. Initial angle given to the algorithm is recovered by comparing the successively rotated and translated set with the one given initially to ICP. Found angles for different initial rotation angles are given in Table 4.1**.** Three-dimensional iterative closest point algorithm implemented in MATLAB is given in Appendix H.

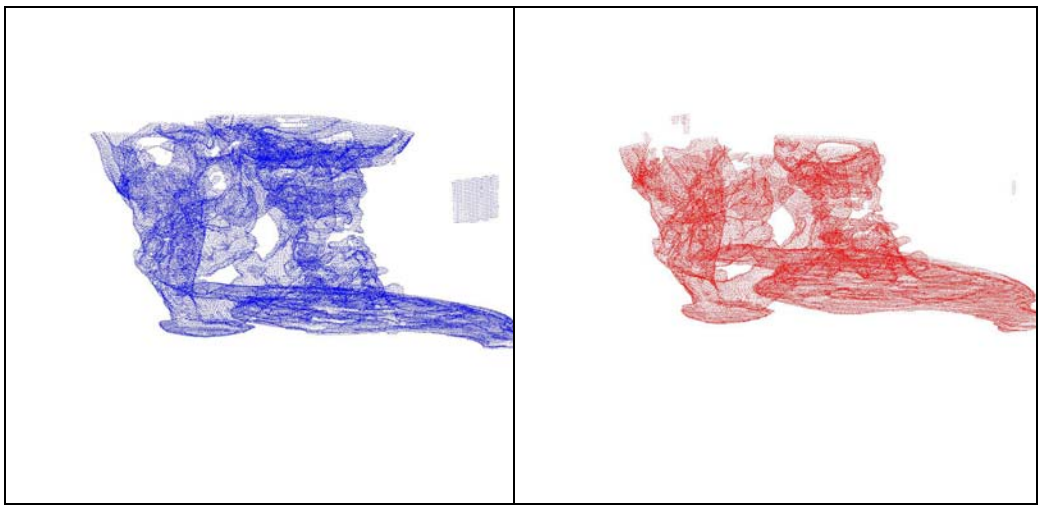

Figure 4.8 a) Original isosurface vertices and b) vertices in the rotated dataset

Figure 4.9 Reduced patch vertices of the rotated dataset

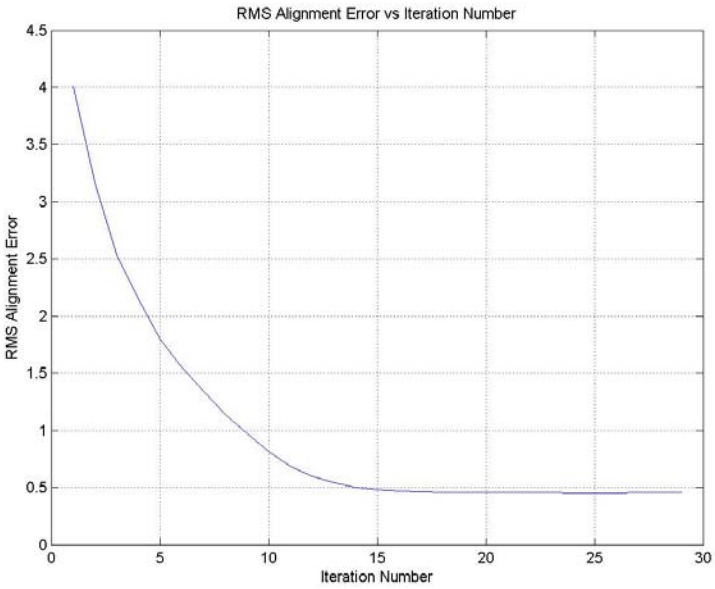

Figure 4.10 RMS error versus iteration number for 3-D automatic registration

Initial angle	**x**	**y**	**z**	**Found angle**	**x**	**y**	**z**
	1	1	1		1.0336	1.0070	0.9184
	2	2	2		1.9319	2.0804	1.9381
	3	3	3		2.9704	3.0547	2.9147

Table 4.1 Initial angles given and result of ICP algorithm

As seen from the table, angle is recovered with less than 10 percent error in all the rotations in Table 4.1. This amount of error depends on how good the corresponding points in the two datasets are selected. Also the distance metric used in ICP is important. Distance metric should be point to surface since selection of exact corresponding points in two datasets is not possible. Instead of using bone surface points that correspond to face vertices, for better results extremal points or crest lines defined in [21]-[22] can be used for extracting point pairs for head CT registration. It is shown in [42] that errors smaller than the resolution can be achieved using extremal points in the registration.

The steps for 3-D automatic registration are as follows.

1. Both head CT datasets taken at different times are segmented to extract the bone tissue. The discretization of both datasets should be made the same before this process therefore interpolation is needed.
2. Point features from both datasets are extracted especially anatomical landmark points that do not change with rigid rotation.
3. Points in one of the sets are reduced by sampling to speed up the search process. Sampling might be random to inhibit error localization. Also sampling may be uniform or it may be non-uniform.
4. ICP algorithm is applied to align one dataset to the other.
5. Using the aligned set and the initial set, the overall transformation matrix is found.
6. Transformation matrix is applied to one of the datasets for slice by slice comparison, visualization and measurement purposes.

CHAPTER 5

5. CONCLUSION

Medical imaging combined with powerful segmentation, registration and visualization algorithms allow medical treatment to be planned easily and accurately. Treatment planning consists of imaging body part at different time periods, segmentation of the part under investigation and registration of datasets gathered during this period. Comparison of these images and datasets provides the effectiveness of the treatment, the differences between planned and achieved results. The operator than decides on how to proceed in the next step of the treatment.

In this thesis, various segmentation algorithms are explored that can be used in the segmentation stage of treatment planning. Using the available algorithms, a hybrid algorithm is developed for bone tissue segmentation from head CT images. Although the algorithm is applied to head CT images, it can be used to segment bones from any body part. Segmentation is automated while leaving some of the parameters left for user control. This property of segmentation algorithm provides minimized errors dependent on user interaction, but on the other hand presents a means of control on the segmentation task.

Registration step of medical treatment planning is investigated for two dimensional and three dimensional images that involve rigid transformations. Possibility of automatic registration of two datasets taken at different times is shown. The accuracy of the results is discussed. However much registration algorithms discussed in this thesis are confined to rigid transformations, registrations of datasets involving non-rigid transformations is possible by

modifying the optimal transformation search stage. Like the segmentation counterpart, this registration algorithms are not restricted for head CT images; they can be used in any dataset that involve rigid registration.

In this thesis, visualizations of the segmented bone from head CT slices from two patients are provided to show the effectiveness of developed hybrid segmentation algorithm. Successive visualizations of datasets taken at different times will reveal the effectiveness of the planned treatment using this segmentation algorithm. Measurements can be taken that will also be helpful in guiding the treatment.

Developed and investigated algorithms throughout the thesis can be used to monitor the condition of patients and to constitute a planning strategy for individual patients. These algorithms may be tested with real datasets and automatic or manual measurements may be taken by comparing datasets taken at different time periods for a specific type of treatment in the future for determining the accuracy of the algorithms.

REFERENCES

[1] H.R.Winn, R.B. Stanley , “Intra-Operative CT in neosurgery and maxillofacial reconstruction”, MedicaMundi, Vol. 42., Issue1, pp 12-14, March 1998.

[2] Biography of Wilhelm Conrad Röntgen, Nobel e-museum Web Site, http://www.nobel.se/physics/laureates/1901/rontgen-bio.html

[3] Terry M. Peters, “Image-guided surgery: From X-rays to Virtual Reality”, Computer Methods in Biomechanics and Biomedical Engineering, Issue 4(1), pp 27-57, 2000.

[4] ME Falimirski , R. Gonzalez, A.Rodriguez , J. Wilberger , “The need for head computed tomography in patients sustaining loss of consciousness after mild head injury”, J Trauma., 55(1), pp 1-6, July, 2003.

[5] M.Fukui , T.Blodgett , C. Meltzer, “PET/CT imaging in recurrent head and neck cancer.”, Semin Ultrasound CT MR, 24(3), pp 157-163. , 2003

[6] Avinash C.Kak, Malcolm Slaney, “Principles of Computerized Imaging”, IEEE Press, ISBN 0-87942-198-3

[7] W.A. Kalender, R. Hebel, and J. Ebersberger, "Reduction of CT artifacts caused by metallic implants." Radiology, 164 (2), pp 576-7, August, 1987.

[8] G. Wang, D. L. Snyder, J. A. O'Sullivan, and M. W. Vannier. “Iterative deblurring for CT metal artifact reduction”, IEEE Trans. on Medical Imaging, Vol 15, pp 657-664, 1996.

[9] George J Grevera, Jayaram K. Udupa, “An Objective Comparison of 3-D Image Interpolation Methods”, IEEE Transactions on Medical Imaging, Vol 17. No. 4, pp 642-652, August, 1998.

[10] Thomas M, Lehmann, Claudia Gönner,Klaus Spitzer, “Survey: Interpolation Methods in Medical Image Processing”, IEEE Transactions on Medical Imaging, Vol 18. No.11, pp 1049-1075, November, 1999.

[11] Ramesh Jain, Rangachar Kasturi, Brian G.Schunck, “Machine Vision”, McGraw-Hill International Editions, Computer Science Series, ISBN 0-07-113407-7

[12] Kass, M., Witkin, A., Terzopoulos, D, “Snakes: Active contour models”, Int. Journal of Computer Vision, pp 321-331, 1987.

[13] L.D. Cohen, "On Active Contour Models and balloons", CVGIP: Image Understanding, Vol 53, pp 211, 1991

[14] Jae S.LIM, “Two-Dimensional Signal and Image Processing”, Prentice Hall Signal Processing Series, ISBN 0-13-935322-4

[15] Thomas Sebastian, Hüseyin Tek, Scott W. Wolfe, Joseph J. Crisco, and Benjamin Kimia, “Segmentation of carpal bones from 3-D CT images using skeletally coupled deformable models”. MICCAI, pp 1184-1194, 1998.

[16] A. Ossen, T. Zamzow, H. Oswald, E. Fleck, “Segmentation of Medical Images Using Neural-Network Classifiers”, NNESMED, pp. 427-432, 1994.

[17] B. Dogdas, D. Shattuck, and R. M. Leahy, “Segmentation of Skull in 3D Human MR Images Using Mathematical Morphology”, Proceedings of SPIE Medical Imaging Conference, vol. 4684, paper 180, pp. 1553-1562, 2002.

[18] X. Pennec, P. Cachier, N. Ayache, “Tracking brain deformations in time sequences of 3D US images”, Pattern Recognition Letters,24(4-5), pp 801-813, February, 2003.

[19] Lisa Gottesfeld Brown, “A survey of image registration techniques”, ACM Computing Surveys, 24(4), pp 325-376, December, 1992.

[20] M.Capek, R.Wegenkittl, A.König, “Multimodal Medical Volume Registration Based on Spherical Markers”, 9-th International Conference in Central Europe on Computer Graphics, Visualisation and Computer Vision’2001, Pilsen, February 5-9, 2001, Czech Republic

[21] J.P. Thrion, A. Gourdon, “The 3D Marching lines Algorithm and its Application to Crest Lines Extraction”, Raports de Recherce 1672, INRIA, 1992.

[22] J.P. Thrion, S. Benayoun, “Image Surface Extremal points, new feature points for Image Registration”, Raports de Recherce 2003, INRIA, 1993.

[23] G. Kagadis, K. Delibasis , G. Matsopoulos , N. Mouravliansky , P. Asvestas, G. Nikiforidis, “A comparative study of surface- and volume-based techniques for the automatic registration between CT and SPECT brain images”, Med Phys, 29(2), pp 201-213, February, 2002

[24] La Palombara PF, Fadda M, Martelli S, Marcacci M., “Minimally invasive 3D data registration in computer and robot assisted total knee arthroplasty”, Med Biol Eng Comput., 35(6), pp 600-610, November, 1997.

[25] Declerck J, Feldmar J, Goris ML, Betting F., “Automatic registration and alignment on a template of cardiac stress and rest reoriented SPECT images”, IEEE Trans Med Imaging,16(6), pp 727-737, December, 1997.

[26] C. A. Pelizzari, G. T. Y. Chen, D. R. Spelbring, R. R. Weichselbaum, , C. T. Chen, “Accurate three-dimensional registration of CT, PET, and/or MR images of the brain,” J. Comput. Assist. Tomogr., vol. 13, pp. 20–26, 1989.

[27] Florin Popescu, Marco Viceconti, Erika Grazi, Angelo Cappello, “A new method to compare planned and achieved position of an orthopedic implant”, Comput Methods Programs Biomed., 71(2), pp 117-127, June, 2003.

[28] Riccardo Lattanzi, Erika Grazi, Marco Viceconti, Angelo Cappello, Aldo Toni, “Accuracy and repeatability of cementless total hip replacement surgery in patients with deformed anatomies”, Med Inform Internet Med.,28(1), pp 59-71, March, 2003.

[29] Richard A.Robb, “Three Dimensional Visulization in Medicine and Biology”, Handbook of Medical Imaging: Processing and Analysis, ed Isaac N.Bankman, Academic Press, San Diego, CA, Chapter2, pp 685-712, 2000

[30] MD Seeman, Claussen CD, "Hybrid 3D visualization of the chest and virtual endoscopy of the tracheobronchial system: possibilities and limitations of clinical application.", Lung Cancer, 32(3), pp 237-46, June 2001.

[31] D.H. Boehm , C. Detter , M.B. Arnold , T. Deuse, H. "Robotically assisted coronary artery bypass surgery with the ZEUS telemanipulator system.", Reichenspurner Semin. Thorac. Cardiovasc. Surg., 15(2), pp112-20, April, 2003.

[32] P.M. Van Ooijen PM, R.J Van Geuns, B.J. Rensing, A.H. Bongaerts , P.J. de Feyter , M. Oudkerk, "Noninvasive coronary imaging using electron beam CT:

surface rendering versus volume rendering.", AJR Am J Roentgenol., 180(1), pp 223-236, January,2003.

[33] C.A. Pelizzari , G.T. Chen, "Volume visualization in radiation treatment planning.",Crit Rev Diagn Imaging, 41(6), pp 379-401, December, 2000.

[34] Digital Imaging and Communications in Medicine (DICOM) Standard, National Electrical Manufacturers Association, 2003.

[35] T.Moon, “The Expectation Maximization Algorithm”, IEEE Signal Processing Magazine, pp 47-60, November, 1996.

[36] Rolf Adams, Leanne Bischof, "Seeded region growing", IEEE Trans. on PAMI, Vol. 16, No. 6, pp. 641 -647, June, 1994.

[37] Andrew Mehnert, Paul Jackway, "An improved seeded region growing algorithm", Pattern Recognition Letters, Vol. 18, pp. 1065-1071, 1997.

[38] Maintz, J.B.A. & Viergever, M.A. (1998). “A survey of Medical Image Registration Methods”, Medical Image Analysis 2(1), pp. 1-36, 1998.

[39] J.V. Hajnal, D. J. Hawkes, D.G. Hill.. “Medical Image Registration. CRC Press, Baton Rouge, Florida, 2001 , ISBN 0-8493-0064-9.

[40] P.J.Besl, N.D. McKay, “A method for Registration of 3-D shapes”, IEEE Trans. Patt. Anal. Mach. Intelli. , Vol 14, No:2, pp 239-256, February,1992.

[41] Likar, B., Pernus, F.,“Automatic extraction of corresponding points for the registration of medical images”, Med. Phys. 26 (8), pp 1678-1686, August, 1999.

[42] X.Pennec, J.P. Thirion, “Validation of 3-D Registration Methods based on Points and Frames”, 5th International Conference on Computer Vision- Cambridge, MA, USA- June 20-23, 1995

APPENDICES

APPENDIX A

SEEDED REGION GROWING
Srg2d.m

```
function [out_im]=srg2d(im,seed_points);

j=1:8;
[row,col]=size(im);
size_im=size(im);
l_im=length(im);
label_image=zeros(row,col);
out_im=zeros(row,col);
im=double(im);
seed_label_matrix=assign_labels_to_seeds(seed_points); % assign seed levels to
                                                       %seeds
mean_of_regions=[];

for i=1:size(seed_points,1)  % assign initial means to regions
   mean_of_regions=[mean_of_regions;im(seed_points(i,1),seed_points(i,2))];
end

[row_seed,col_seed]=size(seed_points);
l_seed=row_seed;
ssl=[];

for i=1:l_seed
   neighbour_seed=neighbours(seed_points(i,1),seed_points(i,2),row,col);
   for j=1:length(neighbour_seed)
      if (neighbour_seed(j)>=1 & neighbour_seed(j)<=row*col)
         [delta,ind]=min(abs(ones(1,l_seed)'*im(neighbour_seed(j))-
         mean_of_regions));
         delta=[delta,ind];
         ssl=[ssl;neighbour_seed(j),delta];
      end
   end
   ssl=sortrows(ssl,[2]);
```

```
end

label_image=label_seeds(seed_label_matrix,row,col);                    % update label
% image

for t=1:length(mean_of_regions)
   n=find(im==mean_of_regions(t));
   label_image(n)=t;
end

while isempty(ssl)==0
   y=ssl(1,:);
   ssl=ssl(2:size(ssl,1),:);
   result=0;
   [x_coord,y_coord]=ind2sub(size(im),y(1));
   neigh=neighbours(x_coord,y_coord,row,col);
   labels=label_image(neigh);
   neigh=[neigh,labels];
   n=find(neigh(:,2)~=0 & neigh(:,2)~=-1);  % -1 is used for the boundary label so
                                             % do not look for boundary labels
   if size(n,1)>1
      if (sum(neigh(n,2))/size(n,1)==neigh(n(1),2))  % if all neighbours of y have
                                                     % the same label
         result=1;          % then assign y to this label
         label=neigh(n(1),2);
      end
   else
      [min_delta,label]=min(y(2:3));
      result=1;

   end
   if result==1   % if all the neighbours have the same label
      label_image(y(1))=label; % Set y to this label.
      n=label_image==label;
      l_n=sum(sum(n));
      % update mean of regions
      mean_of_regions(label)=sum(sum((n.*im)))/l_n;
      %update_ssl
      % find neighbours of the grow point
      [x_coord,y_coord]=ind2sub(size(im),y(1));
      neighbour_grow=neighbours(x_coord,y_coord,row,col);

[delta,mind]=min((abs(im(neighbour_grow)*ones(1,length(mean_of_regions))-
ones(1,length(neighbour_grow))'*mean_of_regions')'));
      delta=[delta' mind'];
      temp_ssl=[neighbour_grow,delta];
      [C,IA,IB]=union(ssl(:,1:2),temp_ssl(:,1:2),'rows');
```

```
        ssl=[ssl(IA,:);temp_ssl(IB,:)];
        labeled_pix_ind=find(label_image(ssl(:,1)));
        labeled_pix=ssl(labeled_pix_ind,:);
        ssl=setdiff(ssl,labeled_pix,'rows');
        ssl=sortrows(ssl,[2]);
        n=find(label_image==1);
        out_im(n)=mean_of_regions(1);
        n=find(label_image==2);
        out_im(n)=mean_of_regions(2);
        n=find(label_image==3);
        out_im(n)=mean_of_regions(3);
        imshow(uint8(out_im));
    else
        label_image(y(1))=-1;  %else flag y with boundary label
    end
end

for k=1:length(mean_of_regions)
    n=find(label_image==k);
    out_im(n)=mean_of_regions(k);
end

function neigh=neighbours(x_pixel,y_pixel,row,col);
          neigh=[x_pixel-1,y_pixel-1;x_pixel-1,y_pixel; x_pixel-
          1,y_pixel+1;x_pixel,y_pixel-1;x_pixel,y_pixel+1;x_pixel+1,y_pixel-
          1;x_pixel+1,y_pixel;x_pixel+1,y_pixel+1];
if neigh>1 & neigh<row & neigh<col
    neigh=sub2ind([row,col],neigh(:,1),neigh(:,2));
else
    j=1:8;
    n=find(neigh(j,1)>=1 & neigh(j,2)>=1 &  neigh(j,1)<=row & neigh(j,2)<=col);
    neigh=neigh(n,:);
end

function ssl=initial_neighbour_seed(seed_points,row_max,
col_max,mean_of_regions); %finds neighbours of ssl in image
global im;
[row_seed,col_seed]=size(seed_points);
l_seed=row_seed;
ssl=[];
for i=1:l_seed

neighbour_seed=neighbours(seed_points(i,1),seed_points(i,2),row_max,col_max);
    for j=1:length(neighbour_seed)
        if (neighbour_seed(j)>=1 & neighbour_seed(j)<=row_max*col_max)
          [delta,ind]=min(abs(ones(1,l_seed)'*im(neighbour_seed(j))-
          mean_of_regions));
```

```
            delta=[delta,ind];
            ssl=[ssl;neighbour_seed(j),delta];
        end
    end
    ssl=sortrows(ssl,[2]);
end

function seed_label_matrix=assign_labels_to_seeds(seed_points);
[row,col]=size(seed_points);
seed_label_matrix=seed_points;

for i=1:row
    seed_label_matrix(i,3)=i;
end

function label_image=label_seeds(seed_label_matrix,row,col);
label_image=zeros(row,col);

for j=1:size(seed_label_matrix,1)
    x_pix=seed_label_matrix(j,1);
    y_pix=seed_label_matrix(j,2);
    label_image(x_pix,y_pix)=seed_label_matrix(j,3);
end

function [result,label]=extract_neighbours(pix_coord,row_max,col_max);
global im;
global label_image;
result=0;
label=-1; % initialy y is boundary label
neigh=neighbours(pix_coord(1,1),pix_coord(1,2),row_max,col_max);
label_indices=sub2ind(size(im),neigh(:,1),neigh(:,2));
labels=label_image(label_indices);
neigh=[neigh,labels];
n=find(neigh(:,3)~=0 & neigh(:,3)~=-1);  % -1 is used for the boundary label so do
                                         % not look for boundary labels
if size(n,1)>1
    if (sum(neigh(n,3))/size(n,1)==neigh(n(1),3))  % if all neighbours of y have the
                                                   % same label
        result=1;                                  % then assign y to this label
        label=pix_coord(4);
    end
else
    [min_delta,label]=min(pix_coord(3:4));
    result=1;
    label=pix_coord(4);
end
```

APPENDIX B

SIMPLIFIED REGION GROWING
Region_growing.m

```
function [y]=region_growing(x,threshold,seed_level,block_size);
[row,col]=size(x);
global segmentedImage; % segmented image
global segmentedImage_assignment; % assigned segmented image pixels
global OutputImage;
global pixelCount;
global sumGrey;
global meanGrey;
global seedGrey;
OutputImage=zeros(row,col);
pixelCount=0;
segmentedImage=zeros(row,col);
segmentedImage_assignment=zeros(row,col);
for i=1:row
   for j=1:col
      if(x(i,j)>seed_level)
         OutputImage(i,j)=255;;
         segmentedImage_assignment(i,j)=1;
      elseif(abs(x(i,j)-seed_level)<threshold)
         if (segmentedImage_assignment(i,j)==0)
            pixelCount=0;
            sumGrey=0;
            meanGrey=0;
            seedGrey=seed_level;
            pixelprocessing(x,i,j,10,block_size);
         end
      end
   end
end
y=OutputImage;


function pixelprocessing(x,i,j,threshold,block_size);
        [row_x,col_x]=size(x);
        global greyValue;
        global noMoreNeighbours;
```

```
        global sumGrey;
        global pixelCount;
        global meanGrey;
        global segmentedImage_assignment;
        global OutputImage;
        global seedGrey;
        noMoreNeighbours=0;


while (noMoreNeighbours == 0)

   greyValue=x(i,j);
   sumGrey=sumGrey+greyValue;
   pixelCount=pixelCount+1;
   meanGrey=sumGrey/pixelCount;
   noMoreNeighbours = 0;

   heldI=-1;
   heldJ=-1;
   greyDifference=999;
   neighbour=[-1,0;1,1;0,1;1,1;1,0;1,-1;0,-1;-1,-1];

   for g=1:1:8

      connectedx=i+neighbour(g,1);  % x coordinate of the connected pixel
      connectedy=j+neighbour(g,2);  % y coordinate of the connected pixel
      if (i>=row_x-2 | i<=2) % the pixel is not in the range proceed to next one
         return ;
      end

      if (j>=col_x-2 | j<=2)  % the pixel is not in the range proceed to next one
         return ;
      end

      if (segmentedImage_assignment(connectedx,connectedy)==0)
         neighbourGreyValue= x(connectedx,connectedy);
         diffvalue=abs(meanGrey - neighbourGreyValue);
         if (diffvalue<threshold)
            if (diffvalue<greyDifference)
               heldI=connectedx;
               heldJ=connectedy;
               greyDifference=diffvalue;
            end
         end
       end
   end
```

```
    if (heldI~=-1 & heldJ~=-1)
        segmentedImage(heldI, heldJ) =255;
        OutputImage(heldI, heldJ) =255 ;
        segmentedImage_assignment(heldI,heldJ)=1;
        i=heldI;
        j=heldJ;

    else
        noMoreNeighbours = 1;
        segmentedImage(i, j) = 255 ;
        segmentedImage_assignment(i,j)=1;
        OutputImage(i,j)=255;
        return;
    end
end
```

APPENDIX C

HYBRID SEGMENTATION ALGORITM
Segmentation3.m

```
function [out_image,pixels]=segmentation3(x,ku_std,ku_mean,block_size);
[row,col]=size(x);
out_image=zeros(row,col);
b=ones(1,70);
[x2,thresh]=segmentation1(x,0.5);
edge_x2=edge_zerocross(x2,4,0);
edge_x2=imfill(edge_x2);
[X,num] = bwlabel(x2,8);
feat= imfeature(X,'Area','Centroid','BoundingBox','PixelList');
pixels=[];
for k=1:1:num
   bw_area=feat(k).Area;
   bw_centroid=feat(k).Centroid;
   bw_boundbox=ceil(feat(k).BoundingBox);
   bw_pixellist=feat(k).PixelList;
   bw_boundbox(3:4)=bw_boundbox(3:4);
   if bw_area>30  & min(bw_pixellist(:))~=1 & max(bw_pixellist(:))~=row
   temp_image=zeros(row,col);
   temp_rbg=zeros(row,col);
   [r,c] = find(X==k);
   rc = [r c];
   BW2 = bwselect(X,c,r,8);
   temp_image(BW2)=x(BW2);
   if bw_boundbox(2)+bw_boundbox(4)>row
   upper_limit_row=row;
   else
      upper_limit_row=bw_boundbox(2)+bw_boundbox(4);
   end

   if bw_boundbox(1)+bw_boundbox(3)>col
   upper_limit_col=col;
   else
    upper_limit_col=bw_boundbox(1)+bw_boundbox(3);
   end
```

```
cropped_x=x(bw_boundbox(2):upper_limit_row,bw_boundbox(1):upper_limit_col
);
   mean_x=mean(x(BW2)); % mean of subimage
   sigma_x=std2(temp_image); % standard deviation of  subimage

temp_rbg=region_growing(double(cropped_x),sigma_x*ku_std,mean_x*ku_mean
,block_size);

out_image(bw_boundbox(2):upper_limit_row,bw_boundbox(1):upper_limit_col)=
temp_rbg;
    end
end
```

APPENDIX D

3-D VISUALIZATION CODE
SHOW_CT.m

```
function show_ct(D,iso_level,filename);
close all;
figure(1);
[r,c,d]=size(D);
axis off;
axis tight;
axis image;
axis tight;
grid off;
Ds=smooth3(D);
if iso_level==500
   hiso = patch(isosurface(Ds),'FaceColor',[1,.75,.65],'EdgeColor','none');
else
   hiso = patch(isosurface(Ds,iso_level),'FaceColor',[1,.75,.65],'EdgeColor','none');
end

hcap = patch(isocaps(D),'FaceColor','interp','EdgeColor','none');
view(0,0);
lightangle(0,0);
set(gcf,'Renderer','zbuffer');
lighting phong;
isonormals(Ds,hiso)
set(hcap,'AmbientStrength',.6)
set(hiso,'SpecularColorReflectance',0,'SpecularExponent',50)
set(gcf,'PaperPositionMode','auto')   % Use screen size
view(0,0);
eval(['print -f1 -djpeg ', 'on.jpg']);
view(-90,0);
eval(['print -f1 -djpeg ', 'yan.jpg']);
view(90,0);
eval(['print -f1 -djpeg ', 'obur_yan.jpg']);
view(-44,8);
zoom(2);
eval(['print -f1 -djpeg ', 'profil.jpg'])
```

APPENDIX E

DISTINCIVENESS MAP
distinctive.c

```
#include "matrix.h"
#include "mex.h"
#include "math.h"
#define gray_level 255

double find_sim(double *x, double *y, int size, int num_circ_pixels)

{
int i,j,index1,index2;
double result;
result=0;
for (i=0;i<size;i++)
result=result+abs(x[i]-y[i]);
return (result/255)/num_circ_pixels;
}

int create_circle(double loc, int c_x, int circ_index, double circle_size)

{
int t,p,num_circ_pixels;
num_circ_pixels=0;
for(t=0;t<loc;t++)
for (p=0;p<loc;p++)
if((c_x-t-1)*(c_x-t-1)+(c_x-p-1)*(c_x-p-1)<=circle_size*circle_size+1)
{   circ_index=t+p*loc;
num_circ_pixels=num_circ_pixels+1;
}
return num_circ_pixels;

}


void im_windower (double *im, double *small_im, int origin_x ,int origin_y,
double radius, int im_col_size, int im_element_size)
{
```

```
int i,j,index,window_size,small_im_index;
window_size=2*radius+1;
small_im_index=0;

for (i=-radius;i<=radius;i++)
for (j=-radius;j<=radius;j++)

{
small_im_index=(j+radius)+(i+radius)*window_size;
if((i)*(i)+(j)*(j)<=radius*radius+1)
{
index=origin_x-1+j+(origin_y-1+i)*im_col_size;
if (index>0 & index<im_element_size)
small_im[small_im_index]=im[index];
else
small_im[small_im_index]=0;
}
else
small_im[small_im_index]=0;
}
}

void mexFunction( int nlhs, mxArray *plhs[],
int nrhs, const mxArray *prhs[] )
{
int m,k,l,t,row,col,num_elements,index1,index2,row_counter,col_counter;
double *im;
double *points,*radius;
int    *dims;
int    circ_index;
double *dist_map;
double d,circle_size,loc,num_circ_pixels;
double *circ_pixels,*small_im1,*small_im2;
int *circ_pixel_dims;
int circ_pixel_row;
double *small_im1_data;
double *small_im2_data;
double result;

num_circ_pixels=0;
im=mxGetData(prhs[0]);
radius=mxGetData(prhs[1]);
circ_pixels=mxGetData(prhs[2]);
num_elements=mxGetNumberOfElements(prhs[0]);
dims=mxGetDimensions(prhs[0]);
row=dims[0];
col=dims[1];
```

```
plhs[0]=mxCreateDoubleMatrix(row,col, mxREAL);
dist_map=mxGetData(plhs[0]);
circ_pixel_dims=mxGetDimensions(prhs[2]);
circ_pixel_row=circ_pixel_dims[0];
circle_size=radius[0];
loc=2*circle_size+1.0;
small_im1=mxCreateDoubleMatrix(loc,loc, mxREAL);
small_im2=mxCreateDoubleMatrix(loc,loc, mxREAL);
small_im1_data=mxGetData(small_im1);
small_im2_data=mxGetData(small_im2);
d=1.0;
num_circ_pixels=create_circle(loc,circle_size+1,circ_index,circle_size);

for (col_counter=0;col_counter<col;col_counter++)
for (row_counter=0;row_counter<row;row_counter++)
{
for (m=1;m<circ_pixel_row;m++)
{
k=circ_pixels[m-1];
l=circ_pixels[circ_pixel_row+m-1];
if (row_counter+k>0 & row_counter+k<row & col_counter+l>0 &
col_counter+l<col)
{
im_windower(im,small_im1_data,row_counter,col_counter,circle_size,col,num_el
ements);
im_windower(im,small_im2_data,row_counter+k,col_counter+l,circle_size,col,nu
m_elements);
result= find_sim(small_im1_data,small_im2_data,loc*loc,num_circ_pixels);
if (result !=0)
d=d*result;
else
{
d=0;
break;
}
}
}
if (row_counter+col_counter*col<num_elements)
dist_map[row_counter+col_counter*col]=d;
d=1.0;
}
}
```

APPENDIX F

2-D ITERATIVE CLOSEST POINT
ICP2.m

```
function [R,T] = ICP2(new_target,new_temp,max_iter,threshold);
error=100000;
iteration=0;
pixels1=new_target;
temp=new_temp;
first_pixels2=new_temp;
[r1,c1]=size(new_target);
[r2,c2]=size(new_temp);
while (abs(error)>threshold & iteration<max_iter)
   iteration=iteration+1
   k=dsearchn(pixels1,temp);
   target=pixels1(k,:);
   [r_target,c_target]=size(target);
   save points.mat target temp;
   q0=[0 0 0];
   [x,fval]=fminunc(@cost_function2d,q0);  % minimization step
   [R]=[cos(x(1)), -sin(x(1)) ;sin(x(1)), cos(x(1))];
   f_values(iteration+1)=fval
   error=abs(f_values(iteration+1)-f_values(iteration)); % update error
   temp=temp*R+ones(1,r2)'*[x(2) x(3)];
end
target=temp;
temp=first_pixels2;
save points.mat target temp;
x=fminunc(@cost_function2d,q0);
[R]=[cos(x(1)), -sin(x(1)) ;sin(x(1)), cos(x(1))];
T=[x(2) x(3)];

function f=cost_function2d(x)
load points.mat;
[row1,col1]=size(target);
[row2,col2]=size(temp);
min_row=min(row1,row2);
rxx=cos(x(1));
rxy=-sin(x(1));
```

```
ryx=sin(x(1));
ryy=cos(x(1));
R=[rxx rxy;ryx ryy];
f=1/min_row*sum(sum([target(1:min_row,:)-(temp(1:min_row,:)*R)-
ones(1,min_row)'*[x(2) x(3)]].^2));
```

APPENDIX G

3-D ROTATION USING BILINEAR INTERPOLATION
ICP3.m

```
#include "matrix.h"
#include "mex.h"
#include "math.h"
#define ndims 3

void mexFunction( int nlhs, mxArray *plhs[],
int nrhs, const mxArray *prhs[] )
{

const int dims[]={256,256,100};
long int size,k;
double new_x,new_y,new_z;
double a,b,c;
int n1_x,n2_x,n3_x,n4_x,n5_x,n6_x,n7_x,n8_x;
int n1_y,n2_y,n3_y,n4_y,n5_y,n6_y,n7_y,n8_y;
int n1_z,n2_z,n3_z,n4_z,n5_z,n6_z,n7_z,n8_z;
int n1,n2,n3,n4,n5,n6,n7,n8;
int x,y,z,i;

double *outdata;
double *pdata;
double *R;
double *T;

/* Check for proper number of input and output arguments */
if(nlhs > 1){
mexErrMsgTxt("Too many output arguments.");
}

/* Check data type of input argument  */

if(mxIsEmpty(prhs[0])) {
mexWarnMsgTxt("Input argument is empty\n");
}
R = mxGetData(prhs[1]);
T = mxGetData(prhs[2]);
```

```
plhs[0] = mxCreateNumericArray(ndims,dims,mxDOUBLE_CLASS,mxREAL);
outdata = mxGetData(plhs[0]);
pdata   = mxGetData(prhs[0]);
size    = mxGetNumberOfElements(plhs[0]);
for(i=0;i<size;i++)
{
z= (i/65536);
y= (i/256)%256;
x= (i-i/256*256);

new_x=(x*R[0]+y*R[1]+z*R[2])-T[0];
new_y=(x*R[3]+y*R[4]+z*R[5])-T[1];
new_z=(x*R[6]+y*R[7]+z*R[8])-T[2];

a=new_x-floor(new_x);
b=new_y-floor(new_y);
c=new_z-floor(new_z);

n1_x=floor(new_x);n1_y=floor(new_y);n1_z=floor(new_z);
n2_x=ceil(new_x);n2_y=floor(new_y);n2_z=floor(new_z);
n3_x=ceil(new_x);n3_y=ceil(new_y);n3_z=floor(new_z);
n4_x=floor(new_x);n4_y=ceil(new_y);n4_z=floor(new_z);
n5_x=floor(new_x);n5_y=floor(new_y);n5_z=ceil(new_z);
n6_x=ceil(new_x);n6_y=floor(new_y);n6_z=ceil(new_z);
n7_x=ceil(new_x);n7_y=ceil(new_y);n7_z=ceil(new_z);
n8_x=floor(new_x);n8_y=ceil(new_y);n8_z=ceil(new_z);

n1=(n1_x+(n1_y)*256+n1_z*256*256);
n2=(n2_x+(n2_y)*256+n2_z*256*256);
n3=(n3_x+(n3_y)*256+n3_z*256*256);
n4=(n4_x+(n4_y)*256+n4_z*256*256);
n5=(n5_x+(n5_y)*256+n5_z*256*256);
n6=(n6_x+(n6_y)*256+n6_z*256*256);
n7=(n7_x+(n7_y)*256+n7_z*256*256);
n8=(n8_x+(n8_y)*256+n8_z*256*256);

k=(floor(new_x)+floor(new_y)*256+floor(new_z)*256*256);

if (k>0 & k<size & n1>0 & n1<size & n2>0 & n2<size & n3>0 & n3<size & n4>0
& n4<size & n5>0 & n5<size & n6>0 & n6<size & n7>0 & n7<size & n8>0 &
n8<size)
{
if (new_x<0 | new_y<0 |new_z<0 | n1_x<0 | n1_y<0 | n1_z<0 | n2_x<0 | n2_y<0 |
n2_z<0 | n3_x<0 | n3_y<0 | n3_z<0 | n4_x<0 | n4_y<0 | n4_z<0 | n5_x<0 | n5_y<0
| n5_z<0 | n6_x<0 | n6_y<0 | n6_z<0 | n7_x<0 | n7_y<0 |n7_z<0 | n8_x<0 | n8_y<0
|n8_z<0)
outdata[i]=0;
```

```
else
{
outdata[i]=(1-a)*(1-b)*(1-c)*pdata[n1]+a*(1-b)*(1-c)*pdata[n2]+a*b*(1-
c)*pdata[n3]+(1-a)*b*(1-c)*pdata[n4]+ (1-a)*(1-b)*(c)*pdata[n5]+a*(1-
b)*(c)*pdata[n6]+a*b*(c)*pdata[n7]+(1-a)*b*(c)*pdata[n8];
}
}
}
}
```

APPENDIX H

3-D ITERATIVE CLOSEST POINT
ICP3.m

```
function [q,R,alldata_out,f] =
ICP3(model,alldata,sample_number,max_iter,threshold);
hold off
set(scatter(model, 'b.'), 'markersize', .001);
set(gcf, 'renderer', 'opengl')
hold on
axis off
h_alldata = scatter(alldata, 'r.');
set(h_alldata, 'markersize', .001);
axis vis3d
error=100000;
iteration=1;
xo=[0 0 0 0 0 0];
R = [1,0,0;0,1,0;0,0,1];
q = [1;0;0;0;0;0;0];
difference=100;
f=0;
all_data_length=length(alldata);
Tes=delaunayn(model);
while (abs(error)>threshold & iteration<max_iter & difference>1e-30)
   cov_px=zeros(3,3);
   cov_px1=zeros(3,3);
   iteration=iteration+1;        % find closest points
   pixels=round(rand(1,sample_number)*all_data_length);
   n=find(pixels==0);
   pixels(n)=1;
   data= alldata(pixels,:);
   D=av_distance(data);
   [K,dists]=dsearchn(model,Tes,data);
   mean_d=mean(dists);
   std_d=std(dists);
   median_d=median(dists);
   if mean_d<D
      dist_threshold=mean_d+3*std_d;
   elseif mean_d<3*D
      dist_threshold=mean_d+2*std_d;
```

```
elseif mean_d<6*D
   dist_threshold=mean_d+std_d;
else
   dist_threshold =median_d;
end
m=find(dists<dist_threshold);
sel_points=K(m);
temp=data(m,:);
target=model(sel_points,:);
lt=length(temp);
% calculation of the mean square distance function
   f(iteration)=1/lt*sum(dists)

if iteration>2
   difference=abs(f(iteration)-f(iteration-1));
end

mp=(sum(temp)/lt)';
mx=(sum(target)/lt)';
temp=temp';
target=target';

for i = 1 : lt;
   cov_px1 = cov_px1 + (temp(:,i)*transpose(target(:,i)) - mp*transpose(mx));
end

cov_px=cov_px1/lt;
% Calculation of the Q matrix
A = cov_px - transpose(cov_px);
D = [A(2,3); A(3,1); A(1,2)];
T = cov_px + transpose(cov_px) -
(cov_px(1,1)+cov_px(2,2)+cov_px(3))*[1,0,0;0,1,0;0,0,1];
Q = [cov_px(1,1)+cov_px(2,2)+cov_px(3,3), D(1), D(2), D(3);
   D(1), T(1,1), T(1,2), T(1,3);
   D(2), T(2,1), T(2,2), T(2,3);
   D(3), T(3,1), T(2,3), T(3,3)];
% Updating the quartenions to the eigenvalues corresponding to the largest
%eigenvalue

[eigvec,eigval] = eig(Q);
eigmax = max(eigval);
[eigmmax, index]=max(eigmax);

for i = 1:4
   q(i) = eigvec(i,index);
end
```

```
% Updating the rotation matrix as per the new quartenions

    rxx=q(1)^2+q(2)^2-q(3)^2-q(4)^2;
    rxy=2*[q(2)*q(3)-q(1)*q(4)];
    rxz=2*[q(2)*q(4)+q(1)*q(3)];
    ryx=2*[q(2)*q(3)+q(1)*q(4)];
    ryy=q(1)^2+q(3)^2-q(2)^2-q(4)^2;
    ryz=2*[q(3)*q(4)-q(1)*q(2)];
    rzx=2*[q(2)*q(4)-q(1)*q(3)];
    rzy=2*[q(3)*q(4)+q(1)*q(2)];
    rzz=q(1)^2+q(4)^2-q(2)^2-q(3)^2;
    R=[rxx rxy rxz;ryx ryy ryz;rzx rzy rzz];
    R=R/norm(R);
    % Updating the translation vector
    qT = mx - R*mp;

    for i = 5:7
        q(i) = qT(i-4);
    end
    alldata=alldata*inv(R)+ones(1,all_data_length)'*[q(5) q(6) q(7)];
    scatter(alldata, h_alldata);
    drawnow
end
data_out=data;
alldata_out=alldata;
```