2011

Heterogeneous Modeling of Medical Image Data Using B-Spline Functions

Olya Grove
University of South Florida, olyagrove@gmail.com

Follow this and additional works at: <https://digitalcommons.usf.edu/etd>

Part of the [American Studies Commons](#), [Bioimaging and Biomedical Optics Commons](#), and the [Computer Sciences Commons](#)

Scholar Commons Citation

Grove, Olya, "Heterogeneous Modeling of Medical Image Data Using B-Spline Functions" (2011). *USF Tampa Graduate Theses and Dissertations*.
<https://digitalcommons.usf.edu/etd/3130>

This Dissertation is brought to you for free and open access by the USF Graduate Theses and Dissertations at Digital Commons @ University of South Florida. It has been accepted for inclusion in USF Tampa Graduate Theses and Dissertations by an authorized administrator of Digital Commons @ University of South Florida. For more information, please contact digitalcommons@usf.edu.

Heterogeneous Modeling of Medical Image Data Using
B-Spline Functions

by

Olya Grove

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Major Professor: Les A. Piegl, Ph.D.
Daniel C. Simkins, Ph.D.
Dmitry Goldgof, Ph.D.
Sudeep Sarkar, Ph.D.
Don Cameron, Ph.D.

Date of Approval:
July 26, 2011

Keywords: BioCAD, Tissue density variations, B-Spline curves,
Anatomical contours, B-Spline surfaces

Copyright © 2011, Olya Grove

Acknowledgements

I would like to express my sincere and deep gratitude to my advisor Dr. Les Piegl, for his inspiration, advice and encouragement throughout my graduate studies. Dr Piegl's guidance and support has been quintessential to my growth as a student and a researcher. I would also like to thank my advisory committee members: Dr. Daniel C. Simkins, Dr. Dmitry Goldgof, Dr. Sudeep Sarkar, and Dr. Don Cameron for their participation, time and helpful suggestions.

I would like to take this opportunity to thank the faculty and staff at the CSE department for their support and assistance throughout my studies at USF. A special thank you to Catherine Burton for her support and willingness to help the students.

I would also like to give special acknowledgment to my research group partners who provided great and much needed feedback during our seminar discussions. Big thanks to my lab friend Khairan Rajab for helpful suggestions and motivation when needed.

I would like to thank all friends at USF and elsewhere for making my life easier at times and being understanding of my demanding schedule.

Finally, I would like to thank my dear husband Michael and my parents half a world away. It is with their unconditional love, support and understanding that I was able to complete this journey.

Table of Contents

List of Tables	iv
List of Figures	v
Abstract	xii
Chapter 1 - Introduction	1
1.1 Motivation	1
1.2 Prior Works	8
1.2.1 Evaluated vs. Unevaluated Models	10
1.2.2 Evaluated Models	11
1.2.2.1 Finite Element Based Representation Schemes	11
1.2.2.2 Voxel Models	14
1.2.3 Unevaluated Models	15
1.2.3.1 Feature Based Modeling	16
1.2.3.1.1 Single Feature	17
1.2.3.1.2 Multiple Features	19
1.2.3.2 Implicit Function Based Model	21
1.2.3.3 Set Based Shemes	25
1.2.3.4 B-Spline-Based Representations	25
1.2.3.5 Function Description Methods	28
1.2.4 Prior Works Concluding Remarks	29
1.3 Contributions	30
1.4 Organization of the Dissertation	32
Chapter 2 - B-Spline Modeling Fundamentals	34
2.1 Fundamental Curve and Surface Representations	34
2.1.1 Explicit Equations	35
2.1.2 Implicit Equations	36
2.1.3 Parametric Equations	37
2.2 Polynomial Curves	39
2.2.1 Bernstein Polynomials and Bezier Curves	40
2.2.2 B-Spline Basis Functions and B-Spline Curves	42
2.2.3 Basis Functions	43
2.2.4 B-Spline Curve	46
2.2.5 Knot Vector	49

2.2.6 Curve Degree.....	51
2.3 Biparametric Cubic Surface Patches	53
2.4 B-Spline Surfaces	54
2.5 Modeling with B-Spline Curves and Surfaces	56
2.5.1 Data Fitting.....	57
2.5.2 Global Curve Approximation	60
2.5.2.1 Least Squares Approximation	60
2.5.3 Surface Lofting.....	63
Chapter 3 - Modeling in Medical Applications.....	66
3.1 MRI/CT Fundamentals and Data Acquisition.....	67
3.1.1 X-Ray Computed Tomography	68
3.1.1.1 CT Numbers	71
3.1.1.2 CT Safety and Image Quality	71
3.1.2 Magnetic Resonance Imaging	74
3.1.2.1 Instrumentation and System Components.....	76
3.1.2.2 MRI Image Quality	78
3.2 Medical Image Processing Fundamentals	79
3.2.1 Manual Segmentation.....	80
3.2.2 Image Processing Basics	80
3.2.3 Image Processing Techniques	84
3.2.3.1 Image Smoothing	84
3.2.3.2 Image Registration	85
3.2.3.3 Image Segmentation.....	86
3.2.3.4 Boundary Detection.....	87
3.2.3.4.1 Active Contours-Snakes.....	89
3.2.3.4.2 Active Contours in Medical Image Analysis	91
3.2.4 Computational Anatomical Models.....	93
3.2.4.1 Voxel Based Models	95
Chapter 4 - Heterogeneous Modeling Using B-Splines.....	98
4.1 Contour Fitting with B-Spline Curves.....	98
4.1.1 Medical Data Preprocessing.....	100
4.1.1.1 Boundary Detection.....	103
4.1.1.2 Edge Based Segmentation	107
4.1.1.3 Canny Edge Detector Implementation	108
4.1.1.3.1 Smoothing.....	109
4.1.1.3.2 Calculating Gradient	109
4.1.1.3.3 Non-Maximum Suppression	110
4.1.1.3.4 Hysteresis Thresholding.....	111
4.1.1.4 Edge Following	111
4.1.1.5 Contour Cleaning	113
4.1.1.6 Point Data Segmentation for Curve Fitting.....	120
4.1.2 B-Spline Curve Fitting	123
4.1.3 B-Spline Curve Fitting Examples.....	127
4.2 Heterogeneous Fitting with a B-Spline Surface	136

4.2.1	Generating Point Cloud	138
4.2.1.1	Point Density	143
4.3	Point Cloud Approximation with a B-Spline Surface	151
4.3.1	Heterogeneous Cross Sectional Contour Fitting	156
4.3.2	Fuzzy Knot Vector Technology	163
4.3.3	Fuzzy Surface Approximation.....	169
4.3.4	Point Density Rendering	173
Chapter 5 - Conclusions and Future Work.....		182
References		188

List of Tables

Table 4.1. Data reduction capabilities.....	136
Table 4.2. Point density per pixel vs. pixel intensity	175
Table 4.3. Point density over [3x3] pixel window vs. pixel intensity	176

List of Figures

Figure 1.1. Classification of existing representation schemes [3]	9
Figure 1.2. Evaluated (a, b) and unevaluated (c) representation of a curve	10
Figure 1.3. Tetrahedron mesh node	13
Figure 1.4. Interior of a solid is represented with a collection of voxel elements	14
Figure 1.5. Exact distance field of an S-shaped 4 th -order B-Spline curve [38]	16
Figure 1.6. Representation of a general field function.....	18
Figure 1.7. Multiple-features representation scheme.....	19
Figure 1.8. Material functions represented explicitly with distance fields of features	20
Figure 1.9. Modeling geometry of region Ω using theory of R-functions [34].....	22
Figure 1.10. Set-based material model	25
Figure 2.1. Cubic Bezier curve	41
Figure 2.2. Third degree basis functions, uniform knots	45
Figure 2.3. Third degree basis functions, double internal knots	45
Figure 2.4. Third degree basis functions, full multiplicity in knots.....	46
Figure 2.5. Spline threaded between ducks	46
Figure 2.6. A B-Spline curve is specified in terms of a knot vector, a degree and a set of control points.....	48
Figure 2.7. Modifying B-Spline curve by moving control point has local effect on the shape	48

Figure 2.8. Multiple knots and their influence on the shape of the curve.....	50
Figure 2.9. B-Spline curves of different degrees defined over the same control points.....	51
Figure 2.10. The effect of control point manipulation on the local shape based on the degree of the curve	52
Figure 2.11. A control polygon and the resulting Bezier patch	53
Figure 2.12. Bicubic B-Spline surface consisting of 3x3 patches together with its control net.....	55
Figure 2.13. Example of interpolation and approximation.....	58
Figure 2.14. Impedance spectra dataset with 16370 points (a).....	63
Figure 2.15. Lofted surface.....	64
Figure 3.1. Patient table is continuously translated through source-detector plane while X-ray source tube is rotating and acquiring projection data.....	69
Figure 3.2. Schematic representation of the steps involved in 3D computed tomography	70
Figure 3.3. Nucleus visualized as a ball spinning about an axis.....	74
Figure 3.4. Block diagram of MR scanner components [47].....	76
Figure 3.5. MRI image slices	77
Figure 3.6. Automatic edge detection using MIMICS.....	82
Figure 3.7. Semi-automatic segmentation	83
Figure 3.8. Original CT scan (a) and smoothed version (b)	85
Figure 3.9. Edges produced by Canny edge detector for a high resolution cross sectional image from Visible Human Dataset	88
Figure 3.10. Original CT scan (a) processed by thresholding: $T=1200$ (b).....	89
Figure 3.11. Active contours applied to MRI data – an iterative procedure.....	90
Figure 3.12. Snakes models fail to capture fine details	92

Figure 3.13. Voxel representation of a cube in object space	95
Figure 3.14. Voxel representation of a CT slice	96
Figure 3.15. Voxel model built from Visible Human Dataset	97
Figure 4.1. Extracted and fitted contours from sequential cross-sectional CT images stacked (a) and lofted (b) using Rhino lofting capability.....	98
Figure 4.2. CT image showing great contrast between the bone and the surrounding tissue (a)	99
Figure 4.3. A CT image of fibula and tibia from the CT dataset	100
Figure 4.4. MRI of the pancreas – excellent soft tissue contrast	101
Figure 4.5. Cross sectional photograph from male cadaver dataset	102
Figure 4.6. Using commercial software for semi-automatic segmentation will require a significant amount of post-processing.....	103
Figure 4.7. DICOM image and corresponding intensity values	104
Figure 4.8. Global thresholding focuses on one region of interest	105
Figure 4.9. Applying global threshold to the original CT image (a) yields different results.....	106
Figure 4.10. Region of interest segmented using thresholding (a)	106
Figure 4.11. Applying Canny algorithm to MRI image of the lung (a) produces a binary edge map (b).....	108
Figure 4.12. Adjusting thresholds in hysteresis thresholding step results in fewer edges.....	111
Figure 4.13. Dangling pixels will disrupt the ordering of the pixels when using DFS approach	113
Figure 4.14. Canny edge map (a) and a contour approximated in the least square sense (b)	114
Figure 4.15. Original CT scan (a) followed by the edge map (b)	115
Figure 4.16. The points for inner contour form a jaggy data set (a) and the interpolating curve passes through every point (b)	116

Figure 4.17. B-Spline naïve Least-Squares approximation	117
Figure 4.18. Original dataset (a) approximated with a least-squares B-Spline curve (b)	117
Figure 4.19. Level of smoothing zero (a) and one (b)	119
Figure 4.20. Level of smoothing five (a)	119
Figure 4.21. Piecewise C^0 Bezier approximation	123
Figure 4.22. Segmentation examples	123
Figure 4.23. The main steps of the B-Spline boundary curve approximation algorithm	124
Figure 4.24. Curve fitting example 1	127
Figure 4.25. Curve fitting example 2	129
Figure 4.26. Curve fitting example 3	130
Figure 4.27. Curve fitting example 4	131
Figure 4.28. Curve fitting example 5	132
Figure 4.29. Curve fitting example 6	133
Figure 4.30. Curve fitting example 7	134
Figure 4.31. Border detected (a) and fitted with a B-Spline (b) automatically using MIMICS.....	135
Figure 4.32. Segment of fitted contour	135
Figure 4.33. B-Spline curve can be fitted to a dataset with discontinuities.....	136
Figure 4.34. Intensity values are captured and preserved through point density.....	137
Figure 4.35. Trabecular and cortical bone tissue are differentiable in the CT scan	139
Figure 4.36. The human eye will see halftone dots (a) as continuous tone imagery (b) from sufficient distance	141

Figure 4.37. Halftone masks replacing intensity value with a pattern of uniformly distributed dots.....	142
Figure 4.38. Original CT scan (a,c) and the corresponding halftone (b,d)	142
Figure 4.39. Neighborhood is defined as either a pixel or a voxel	144
Figure 4.40. Intensity value at each pixel is reproduced through a cluster of points associated with that pixel location	145
Figure 4.41. Original CT scan (a) and its edge map (b) combined to generate a density point cloud (c).....	146
Figure 4.42. MRI of pancreas (a) and corresponding density point cloud (b).....	147
Figure 4.43. Trabecular bone structure of bovine knee and corresponding density point cloud	148
Figure 4.44. Denser point cloud captures more tissue variation detail.....	149
Figure 4.45. Close-up of the same denser point cloud reveals presence of important anatomical detail	149
Figure 4.46. CT scan (a) and corresponding density point clouds with a varying degree of point assignment.....	150
Figure 4.47. Rectangular point cloud grid with rows containing variable number of data points.....	151
Figure 4.48. Point cloud data is organized using a rectangular grid and each row of data is approximated with a B-Spline curve.....	152
Figure 4.49. Lofting offers a lot of flexibility.....	153
Figure 4.50. General lofting through contours with various number of control points results in an overall high number of control points	154
Figure 4.52. Data structure arranging the dataset for cross sectional fitting	157
Figure 4.53. Point densities (b) are generated from the row of pixel intensities (a).....	158
Figure 4.54. Row of pixels (a) followed by the point densities (b) and the result of curve sampling (c).....	161
Figure 4.55. Cross sectional curve from a CT dataset (a) and curve sampled at uniformly spaced parameters (b)	161

Figure 4.56. Fitted B-Spline curve is evaluated by varying the number of sampling points.....	162
Figure 4.57. The “densest” row of data is chosen as the basis for master knot calculation	165
Figure 4.58. Fitted curve knot vector is updated with a knot from a master knot repository.....	166
Figure 4.59. The master knot gets updated with a knot not represented in knot repository.....	167
Figure 4.60. B-Spline curve with 6 control points.....	167
Figure 4.61. B-Spline curve with 7 control points.....	168
Figure 4.62. Fuzzy knot vector algorithm reuses internal knots and reduces the number of knots in the common knot vector	168
Figure 4.63. Conventional lofting vs. fuzzy surface fit	171
Figure 4.64. Example 1: the effect of increasing the number of control points	172
Figure 4.65. Example 2: the effect of increasing the number of control points	173
Figure 4.66. Original point cloud (a) and uniformly sampled surface (b).....	174
Figure 4.67. Sampled point cloud (a) and the histogram of point density distributions (b)	175
Figure 4.68. Original MRI images (a,c,e) and density maps produced by sampling fitted B-Spline surface function (b,d,f).....	177
Figure 4.69. MRI of the pancreas (a) and the density grayscale map of the B-Spline surface (b)	178
Figure 4.70. MRI of the brain tumor (a)	179
Figure 4.71. Visible Human Dataset, high resolution.....	179
Figure 4.72. Visible Human Dataset image converted to grayscale (a).....	180
Figure 4.73. Example 1: comparing the original image and the grayscale colormap.....	181

Figure 4.74. Example 2: comparing the original image and the grayscale colormap.	181
--	-----

Abstract

Ongoing developments in the field of medical imaging modalities have pushed the frontiers of modern medicine and biomedical engineering, prompting the need for new applications to improve diagnosis, treatment and prevention of diseases.

Biomedical data visualization and modeling rely predominately on manual processing and utilization of voxel and facet based homogeneous models. Biological structures are naturally heterogeneous and in order to accurately design and biomimic biological structures, properties such as chemical composition, size and shape of biological constituents need to be incorporated in the computational biological models.

Our proposed approach involves generating a density point cloud based on the intensity variations in a medical image slice, to capture tissue density variations through point cloud densities. The density point cloud is ordered and approximated with a set of cross-sectional least-squares B-Spline curves, based on which a skinned B-Spline surface is generated. The aim of this method is to capture and accurately represent density variations within the medical image data with a lofted surface function.

The fitted B-Spline surface is sampled at uniformly distributed parameters, and our preliminary results indicate that the bio-CAD model preserves the density variations of the original image based point cloud. The resultant surface can thus be visualized by mapping the density in the parametric domain into color in pixel domain. The B-Spline

function produced from each image slice can be used for medical visualization and heterogeneous tissue modeling. The process can be repeated for each slice in the medical dataset to produce heterogeneous B-Spline volumes.

The emphasis of this research is placed on accuracy and shape fidelity needed for medical operations.

Chapter 1 - Introduction

1.1 Motivation

Ongoing developments in the field of medical imaging modalities have pushed the frontiers of modern medicine and biomedical engineering, prompting the need for new applications to improve diagnosis, treatment and prevention of diseases. Currently, a wide range of medical acquisition modalities are available, such as Computed Tomography (CT) and magnetic resonance imaging (MRI), which are capable of generating accurate portrayals of body interior [5].

Development of medical image based 3D computational models that represent human anatomy and physiology have remained an active area of research [7]. Capable of depicting the body interior and the select organs accurately, these models are of great demand in applications such as dosimetry calculations in nuclear medicine, visualization and surgery planning, prosthesis design and manufacturing, and so on. Ideally, these models are not only able to visualize the internal structure of the body, but also to mimic as closely as possible the biological properties of the select organs and tissues [7].

The basic and most widely used format for generating 3D models is a volumetric voxel model [10]. Voxel models are currently considered to be the most faithful representation of the human anatomy [6]. These models are built from segmented anatomical images and are capable of producing rather realistic representations of the

inner structures. Volumetric models consists of voxels – the identically sized, tightly packed parallelepipeds that are formed by discretizing the object space with sets of planes that are parallel to each other [5]. Voxels are 3D analogs of pixels: each voxel contains a physical parameter value of the corresponding volume in the patient’s interior [12].

2D contiguous medical image slices are customarily stacked to produce 3D anatomical models of the human body. Regardless of the medical modality used, the output is an ordered sequence of image data slices with a gap between two consecutive images. Un-imaged space between the adjacent slices of patient data needs to be interpolated to fill in the space. If available, coronal, axial and sagittal slices can be combined to aid the interpolation between slices. During the interpolation, small structures, especially cable-like elements such as small blood vessels and neuronal dendrites [10] are not reconnected properly.

Voxel models suffer from a number of shortcomings. Since a voxel is the smallest element, volumetric voxel models have rough surface textures and suffer from aliasing, and structures smaller than a voxel – such as skin, small blood vessels and bone cartilage – cannot be adequately modeled. CT and MRI procedures, generate, on average, more than 2 million voxels per patient examination [5], which amounts to a large and cumbersome to process set of data.

Although voxel models produce rather realistic renderings of human anatomy and are suitable for rapid prototyping, these models lack geometric representation which makes them unsuitable for bioengineering design and analysis. In order to further advance biomedical engineering, anatomically accurate models suitable for applications in biophysics modeling, tissue and cell fabrication and printing, biomimetic modeling and

design are needed. Consequently, it is important to further convert image models into a CAD-based model [10].

Parametric mathematical models are the preferred representation in engineering design [11] due to a number of their properties such as accuracy, speed, projective and affine invariance, and ability to represent both free-form and conventional objects. NURBS models offer a myriad of advantages over voxel models. Resolution independent, these models can be easily resized and converted to voxel models. NURBS models are compact, concise and mathematically rigorous. Having been used extensively in engineering design, NURBS models enjoy a wide range of interrogation tools and lend themselves nicely for simulation and deformation analysis and manufacturing, which makes them extremely useful in biomedical research.

Due to ongoing developments in the field of medical imaging modalities and reverse engineering techniques, computer-aided technologies are being used extensively in biomedical engineering. Bio-CAD modeling is the field of modeling human anatomy in a CAD-based virtual environment [13] which has found applications in tissue engineering, part modeling and visualization, surgical simulation and planning, as well as manufacturing of various accessories and patient specific prosthetics.

An image-based bio-CAD model is constructed using medical image data produced by imaging modalities such as MRI and CT. A bio-CAD model is usually a boundary representation model – a solid model is defined by the bounding surfaces that enclose it [10]. Currently, there are two techniques used to generate an image-based bio-CAD model [11].

The first approach builds the surface from the point cloud data, using reverse engineering methods or commercially available reverse engineering software packages such as Geomagic [9]. A 3D voxel model generated from segmented image slices can be used to create point data. Point cloud data is triangulated to form a faceted model [9] and after further refinement, NURBS patches are fitted across the outer shape of the model. Alternatively, a 3D voxel model can be converted to a stereolithography (STL) model. The generated STL model can be further refined and imported in Geomagic for NURBS surface generation.

An alternative method to generating a 3D bio-CAD model from medical image data is to reconstruct a surface from serial parallel contours extracted from sequential medical images. Digital images are segmented into regions-of-interest (ROI) and edge points around the boundary are extracted and ordered. The contours are often further thinned to reduce the number of edge points and then closely fitted with closed parametric curves [8]. A set of cross sectional curves extracted from a sequence of evenly spaced planar image slices is then lofted to generate a surface model. The accuracy of the fitted surface model depends largely on the quality of extracted edge points and the B-Spline curves fitted to those contours.

In order to successfully reconstruct a surface model from contours, it is important to accurately identify and extract features of interest from medical images. Although medical image segmentation is an established and mature field, fully automatic segmentation of medical image data remains an unsolved problem [15]. Segmenting medical data remains a challenging task due to the size of the dataset, the complexity of

anatomical shapes, overlap of gray level values between neighboring tissues and organs, lack of consistently distinct boundaries and sampling artifacts and noise [6].

In order to loft through the fitted cross sectional parametric curves, a significant data reduction is required, which is either done manually or by using sub-sampling algorithms [8] which can inadvertently lead to loss of small detail. Consequent contours have to be properly aligned and if adjacent medical images contain multiple contours, a fitting algorithm should be able to determine correspondence between the contours. As a result, a fully automatic process for generating a 3D bio-CAD model from cross-sectional images remains difficult to implement and an effective solution has not yet been published in the literature [8].

Biological structures consist of a variety of tissues and are naturally heterogeneous. In order to adequately perform modeling and simulation, properties such as chemical composition, size and shade of biological tissues need to be incorporated into the computational model [7]. Introducing heterogeneity into the modeling of biological materials results in more accurate simulations and analysis, since mechanical and physiochemical properties are strongly related to the structural organization of the constituent biological materials [16]. Thus, in order to continue to advance our understanding of the human body, a bio-CAD model should not only capture the geometry of the organs of interest accurately, but also contain information about the constituent tissues.

Traditionally, existing CAD representations have been developed based on the premise of material homogeneity. Until recently, the primary objective of classical solid modeling focused on geometry information [1] and thus remained limited to

homogeneous material representation. Due to recent progress in manufacturing technologies, heterogeneous modeling has been gaining a lot of research interest. Introduction of material properties into the design domain has promised an array of advantages and wide range of applications in electrical, mechanical, optical, biomedical, thermal and other fields [3].

The most recent advances in heterogeneous modeling focus mainly on manufacturing and solid free-form fabrication [4] with the purpose of designing and manufacturing of solids with varying material properties. Modeling of tissue heterogeneities differs vastly from the design of heterogeneous engineering components [2]. Biological objects tend to have more complicated geometry than the mechanical components [11] and display a vast variation of material properties within the tissues. Over the course of natural development, these material properties have been formed within the tissues to serve complex biomechanical functions [1]. Biomedical applications currently focus not only on the macroscopic level such as the outer shape of the tissues and organs, but also on the microscopic structures of the tissues and, therefore, the traditional CAD techniques are inadequate for biomedical design [19].

Material properties of the tissues can be correlated with the grayscale values of medical images. Regardless of the type of imaging modality used, each pixel in the medical image contains a physical parameter value of the corresponding region in the patient's interior [20]. Thus, medical image data can be used not only as a means to differentiate between tissues, but also to reproduce the density variation within the tissue itself. Medical image data reveals a relatively smooth change of material density

throughout the image slice [15]. Currently, no underlying methods and exact mathematical relations describing these variations have been reported [21, 22].

Intensity variations in medical data have been correlated in the literature with the heterogeneity of tissues. For example, density variation within the tumor ROI in CT data can be correlated with the tumor histological type [99, 100]. Incorporating tumor heterogeneity based on intensity variations derived from medical image data can be used to better analyze tumor progression and proliferation and offer new therapeutic approaches. Similarly, intensity variations within bone tissue MRI and CT scans, often in combination with bone mass data produced by bone mineral densitometry (BMD) can be used for bone structure analysis. Incorporating this density information into a CAD model can be a useful tool to assess osteoporosis, bone strength and risk of future fracture [12].

In this dissertation, we will present a systematic way to construct an image based heterogeneous model using B-Spline functions. The heterogeneity of the patient's interior is captured through the point cloud density variations and modeled with a lofted surface function. The resultant surface accurately models continuous density variations of the medical data and can be visualized and analyzed in order to assess densities at any region of the surface.

Depending on the medical scanning techniques as the source of data, our modeling technique can be applied to data from various physical and biological sources at a range of scales such as nanoscale, microscale and macroscale, thus allowing incorporating a full range of biological properties into the model.

1.2 Prior Works

Heterogeneous solid modeling is an emerging area of research that has continued to steadily generate great popularity and research interest. As manufacturing, computing and imaging technologies continue to advance, it is becoming more appealing to incorporate material property information along with the geometry description in order to produce a complete representation of an object.

Although traditionally CAD has remained homogeneous, it is now obvious that introducing material modeling to CAD is crucial for design and fabrication. Heterogeneous modeling aims to incorporate material properties into the geometrical representation of the object. It is an interdisciplinary field that requires knowledge in CAD, CAM and other related fields [3] and has a wealth of applications. Heterogeneous models are enriched with knowledge required to perform adequate analysis, simulation and design and, as a result, are useful in biomedical, electrical, optical, geophysical, aeronautical, and an array of other fields.

Due to such a diverse list of applications, a range of heterogeneous modeling schemes that differ in their representation of geometry and materials and the mappings between the two have emerged. Since heterogeneous modeling is a highly interdisciplinary area spanning across CAD, CAM, and CAE fields, each of these domains produced distinct emphases and representations [3]. Thus, approaches generated by the CAD community focus primarily on CAD representation schemes for both geometries and the material heterogeneities, while CAE research is based on analytical and numerical solutions to simulate the heterogeneity and, consequently, the performances, of the models [3]. Given the wealth of approaches and proposed solutions, it would be

tedious to cover all the works existing. Instead, I will attempt to give a structural overview of the major approaches, with the main focus on the prominent prior works related to our approach.

In one of the first works to attempt to characterize the emerging schemes the authors subdivided proposed schemes into three categories: set theory, finite-element based representation and function description methods [23].

Figure 1.1: Classification of existing representation schemes [3].

Kou and Tan [3], in their survey on heterogeneous modeling, identified three major categories for classifying the existing representations (Figure 1.1): evaluated models, unevaluated and composite models. This classification is not rigorous and most certainly not unique, yet it offers a good perspective on the wealth of approaches that have emerged.

1.2.1 Evaluated vs. Unevaluated Models

Evaluated models illustrated in Figure 1.2(a) and Figure 1.2(b) utilize space discretization in order to represent material distribution [3]. Thus, voxel-based representation uses voxels – volumetric elements – to represent a heterogeneous object. Each voxel can have a material distribution associated with it. Voxel representation, both homogeneous and heterogeneous, has served the medical community well, but both representations suffer from the same drawbacks, such as loss of geometric information, voxelization, large memory overhead and aliasing.

Figure 1.2: Evaluated (a, b) and unevaluated (c) representation of a curve.

Unevaluated models (Figure 1.2(c)), on the other hand, combine exact geometry representation with rigorous function based material distribution [3]. As a result, these models are resolution independent and compact, and can offer efficient material interrogation to a specified precision [3].

1.2.2 Evaluated Models

The interiors of solids are discretized into smaller elements such as FEM meshes and voxels. Heterogeneity of the material is represented through intensive space decompositions, and at each element, material decomposition can either be homogeneous or heterogeneous. In order to represent material variation within the smallest element, Bernstein polynomials, B-Spline functions and tri-linear functions have been used as interpolating functions

1.2.2.1 Finite Element Based Representation Schemes

In finite element based representation schemes, the heterogeneous object's interior is discretized using a mesh. A collection of sub-volumes or polyhedrons is used to represent both the geometry and the material properties, with material parameters specified at the nodes of the mesh. Coefficients at the nodes can be used to determine the distribution of material within the solids. Node coefficients can be expressed as functions or be specified using scanning techniques [1]. Hence, material distributions inside a polyhedron are represented using a function through interpolation [3], which also allows mesh based representations to represent complex material distributions. Accuracy of the representation depends largely on the mesh density and the order of basis functions [1].

As a result, finer meshes are often used to represent highly complex regions. This presents a tradeoff between accuracy and efficiency.

Jackson et al [31] proposed a tetrahedral mesh based scheme to represent functionally graded materials (FGM). In this approach, the solid model is subdivided into topological cells and composition blending functions are associated with each sub-region. Each vertex in the proposed mesh maintains not only material properties, but also positional information as a function of the distance from the surface of the model.

Using a cell-tuple structure, a model M is decomposed into a set of cells $\{C\}$. In order to represent an FGM model using the cell-tuple structure, each cell maintains the material information as well as the geometric information. Material composition of the model is represented using a vector valued function $m(x)$ where each component m_i represents the volume fraction of the corresponding material present at a point x within the model.

Models are subdivided into tetrahedral meshes. Model space is subdivided into domains that are topologically simple enough to represent their shape and material function analytically. For each tetrahedron's domain c_k (Figure 1.3), the shape and composition is formulated in terms of control points $\{x_{k,i} || i| = n_g\}$ and control compositions $\{m_{k,i} || i| = n_m\}$ which are blended with barycentric Bernstein polynomials [31]:

$$[x_k(u), m_k(u)] = \left[\sum_{|i|=n_g} B_i^{n_g}(u) x_{k,i}, \sum_{|i|=n_m} B_i^{n_m}(u) m_{k,i} \right]$$

where n_g and n_m are the degrees of variation in shape and composition.

Figure 1.3: Tetrahedron mesh node. Each node of the tetrahedron maintains information about its position in space as well as material composition. Shape and material composition over the interior is evaluated as a linear interpolation.

The proposed representation is capable of representing objects with complex composition distribution but the process is computationally intensive and requires a lot of memory necessary to generate a large amount of meshes to represent the object.

In [4], a new representation scheme for modeling porosity and density of bio-materials is introduced. The proposed scheme for both design and representation is based on stochastic geometry, and internal shape parameters such as porosity are modeled as stochastic processes. The authors adapt an existing constructive solid geometry representation to incorporate internal properties through Boolean operations, enabling the design of heterogeneous scaffold models for tissue engineering. The authors argue that the exact geometrical and topological structure of tissues is not needed to generate tissue scaffolds for guided tissue engineering and regenerations, and, hence, do not incorporate any patient specific or tissue specific knowledge in the design.

1.2.2.2 Voxel Models

Voxel models represent a heterogeneous model with a collection of voxel elements [3]. Each voxel is a volume element which is the basic unit of CT or MRI reconstruction, representing a small cube in space.

Interiors of solids are discretized into uniform 3D cubical voxels (Figure 1.4). Voxel based representation can be generalized as [3]:

$$O = \{V_i\} = \{(x_i, y_i, z_i, m_i), 1 \leq i \leq n\}$$

where V_i is a voxel element from the collection of n voxels that constitute the heterogeneous object O , (x_i, y_i, z_i) represent the geometric location of voxel V_i and m_i represents the material composition over the volume interior.

Typically, a constant material property is assigned to each voxel. Heterogeneity inside each voxel can be also achieved through interpolated material distributions using Bernstein polynomials or tri-linear functions [46].

The material property at a point can be evaluated only by identifying the voxel within which it lies. Material properties can be specified in various ways: scanning techniques such as MRI and CT, and employing various functions and distances.

Figure 1.4: Interior of a solid is represented with a collection of voxel elements.

Voxel based schemes do not guarantee continuous material distribution and are not compatible with the known representation standards in CAD. However, various algorithms exist to construct a locally continuous approximation of material functions from voxel data.

Physical modeling attempts to create a computational copy of the actual object by incorporating the geometry and the physical properties into the object description. In order to adequately capture the essential information about the object, it is instrumental to describe the object in a mathematically rigorous fashion.

Voxel based representation lacks geometry description and is therefore incompatible with the vector based CAD solid modeling environment needed for biomedical analysis and simulation [28].

A methodology for efficient and reliable conversion of medical image data into CAD solid models remains in developmental stages [28].

1.2.3 Unevaluated Models

Unevaluated models utilize exact geometric data representations such as boundary representations and functional representations and (explicit, implicit, analytic) functions to represent material distributions rigorously [3]. Since these models do not rely on space decomposition, they are resolution independent and can be converted to evaluated models. Due to the mathematical description, these models are concise, compact and more suitable for engineering design, simulation and fabrication.

1.2.3.1 Feature Based Modeling

In [1], a material model of bone tissues is presented. The concept of representing geometric shapes is extended to model continuously varying material properties of the interior. The representation scheme is based on the concept of parameterizing the modeling space with distance fields of material features. A distance field for a given set S (geometry of material features) is defined by associating with every point P in the modeling space a value that is equal to the shortest distance from P to S .

In order to construct material functions in terms of intuitive parameters, as opposed to a *posteriori* imposed spatial discretizations, using distance from material features is proposed [38]. A distance function is defined [38] as a function $u: E^3 \rightarrow R$ that computes for any point p in space a real number equal to the Euclidean distance from p to S . Thus, for each point set S , the distance function defines a scalar distance field. Figure 1.5 shows the isolines of the Euclidean distance field for an S-shaped B-Spline curve of order 4.

Figure 1.5: Exact distance field of an S-shaped 4th-order B-Spline curve [38].

Distance fields are used extensively in material modeling to construct and represent models. Since the distance field is an intrinsic property of the feature geometry, it is required to be expressed explicitly [1]. Due to the computational expense of computing exact distance fields for material features and potential loss of differentiability at equidistant points [38], approximations of the distance fields are preferred over the exact distance fields. A number of approximation schemes for representing continuous distance fields, broadly classified into constructive, approximated, and potential function based categories [1]. Smooth approximations of the distance fields are capable of preserving most of the advantageous properties of the distance fields while scaling the computational cost and preserving differentiability.

1.2.3.1.1 Single Feature

The simplest problem of material modeling involves a single material feature - a closed subset S with known material properties [38].

Using single feature based representation, material properties are controlled, in this example, in terms of distance. Given a desired material function $F(u, x, y, z)$ for the feature S , the behavior of F as a function of distance u while keeping all other variables fixed, is examined. As a point p is moved a distance away from the boundary of the feature S , the value of $F(p)$ can be expressed using a Taylor series expansion [1, 38]:

$$F(u) = F_0(0) + uF_1(0) + m \sum_{k=2} \frac{1}{k!} F_k(0) u^k + u^{m+1} \Phi$$

The above formula is the generalization of the classical Taylor series, where the term $|x - x_0|$ is replaced with the value of the distance from a point p to the material feature [1]. This representation of a general field function is illustrated in Figure 1.6.

Figure 1.6: Representation of a general field function. Material function F is represented in the neighborhood of the material feature by a Taylor's series expansion in the powers of a distance field u (a). Material function $e^{-1.5u}$ is prescribed in terms of distance u to the S-curve [38].

Park et al [41] introduced an implicit procedural scheme where the reference feature is modeled with an implicit surface defined by a function f . The level sets of f are used to represent material distributions. Similarly, Liu et al [42] proposed a 'single based composition function' to represent material gradient in heterogeneous objects. Siu and Tan [26] propose the concept of "grading source", defined as "origin of grading", to model functionally graded material distributions. Their proposed material distribution is represented with analytic functions, using distance from the point to the source feature as a variable.

Since material distributions are defined in terms of the distance from the point to the control features, the material heterogeneity in these models [26, 41, 52, 38] are invariant under linear transformations. Single source models prove efficient in material

interrogation if the distances from the points to control features can be efficiently computed. However, these models do not provide sufficient coverage for modeling complex variations of biological materials.

1.2.3.1.2 Multiple Features

Realistically, a heterogeneous model will require definition of several distinct material features, $S_i, i = 1, \dots, n$. With the introduction of multiple materials arises the issue of combining individual material functions while preserving the values and derivatives specified on each material feature. Thus, given a set of material functions $\{P^i\}$, these individual functions must be combined into a single material function $P(P^1, P^2, \dots, P^n)$ using a distance-based representation scheme [38]. An example of the multiple-features representation scheme is illustrated in Figure 1.7.

Figure 1.7: Multiple-features representation scheme. A material function that combines material functions of individual features is constructed [38].

Combining individual material functions remains an open problem. Since the combined material function must, ideally, preserve the values and derivatives of the individual material features, P must interpolate the values and the derivatives at every feature S_i .

The most common interpolation technique utilizes convex combination of P^i :

$$P(p) = \sum_{i=1}^n P^i(p) W_i(p), \quad i = 1, \dots, n$$

where W_i is a weight function that controls the influence of the individual material function associated with feature S_i .

The most popular material blending function used for designing the weight function $W_i(p)$ is the inverse distance based function [1,28,40]. Per the definition of the function, the weight $W_i(p)$ is inversely proportional to some power of distance $u_i^k(p)$:

$$W_i(p) = \frac{\prod_{j=1, j \neq i}^n u_j^k(p)}{\sum_{j=1}^n \prod_{j=1, j \neq i}^n u_j^k(p)}$$

Figure 1.8: Material functions represented explicitly with distance fields of features. The material function $\bar{\rho}$ which represents density of bone tissue is constructed with a grid of 25x25 B-Spline basis functions [1].

In [1], CT scan data is used to obtain material density of the bone tissue. Density fractions of the bone tissue are constructed in distance canonical form using a material function R . In this approach, material functions are represented explicitly with distance fields of features (Figure 1.8). The material function approximates the discrete data from a CT scan, in a least-square sense, using a linear combination of basis functions $\sum_{i=1}^n C_i^R x^R$.

B-Splines are selected to be used as basis functions. The overall material model is thus constructed with a B-Spline basis as:

$$\min \left\{ \int_{\Omega} \left(R - \bar{\rho} \right)^2 d\Omega \right\}$$

where $\bar{\rho}$ is the material density of bone tissue, obtains from a CT scan and defined over a region Ω of the bone. The author points out that since material properties of the different bone tissues are related by various mathematical relations to material density, the concept of features can be used to assign material properties such as elastic modulus and ultimate strengths to each point within the bone model.

1.2.3.2 Implicit Function Based Model

Implicit function based models use functional representation to model both the geometry and the material distribution. The set of points in geometry space E^n is defined as a continuous real valued function of point coordinates $y = f(x_1, \dots, x_n)$, where $f(y) \geq 0$ defines the points that belong to the object and $f(y) < 0$ describes the points outside the object [3,34].

Since the algebraic functional representation has a limited range for representing various shapes, the existing algebraic functions can be thought of as “primitives” or “building blocks” to build more elaborate geometric shapes [34]. These analytic primitives can be combined with logical operations (such as “and” $\wedge$, “or” $\vee$) to form semi-algebraic sets.

Figure 1.9: Modeling geometry of region Ω using theory of R-functions [34].

For example, to define the boundary of the region (Figure 1.9(b)), one can choose the following primitives:

$$\Omega_1 = (r^2 - [(x - x_0)] + [(y - y_0)]) \geq 0$$

$$\Omega_2 = (x^2 - 1) \geq 0$$

$$\Omega_3 = (y - x) \geq 0$$

$$\Omega_4 = (y + x) \geq 0$$

Using the theory of R-functions [34], standard Boolean predicates can be expressed with corresponding real-time functions and implicitly define the object geometry [3]:

$$\Omega = \left\{ \left\{ \left[(y-x) \wedge (y+x) \right] \vee (x^2 - 1) \right\} \wedge \left\{ \left[(y-x) \wedge (y+x) \right] \wedge (r^2 - x^2 - y^2) \right\} \right\} \geq 0$$

The result of applying these functions is illustrated in Figure 1.9(a).

R-functions have found applications in many unexpected areas such as stability of motion, medical diagnostics, and chemical engineering [34].

Functional representation can be extended to model heterogeneous objects as well. Functional representation can be used as the basic model for both point set geometry and the material attributes. A large selection of primitives, operations and relations is available to model both the geometry and the attributes [34].

In [35], the authors introduce a constructive hypervolume model representing the geometry and the attributes independently by real-valued functions. Defining hypervolume object $o = (G, A_1, \dots, A_k)$, where G is a geometry point set and $\{A_k\}$ represents the collection of attributes of G , functional representation of the constructive hypervolume is expressed as [35]:

$$o = (G, A_1, \dots, A_k) : (F(X), S_1(X), \dots, S_k(X))$$

where F is a real-valued function representing point set G and is positive inside the point set and negative on the outside. S_i is a real-valued scalar function representing an attribute A_i . The authors proposed F-Rep constructive tree structure with primitives in the leaves and operations in the nodes of the tree. A tree structure corresponding to function F that defines the point set is constructed, and function S_i defining the attributes is evaluated by a tree traversal procedure.

In [36], authors presented a theoretical framework for a hybrid model of hypervolumes which combined a cellular representation and a constructive representation

using real-valued functions. The proposed framework allowed for independent representation of geometry and attributes, defining attributes as a property of space the attributes are associated with, rather than the geometric object itself. The hybrid model combined the advantages of functional representation and cellular representation, which according to [36] made the model compatible with existing representations such as voxel models and constructive volume geometry model.

An attribute A_i is modeled as a set of values N_i embedded into a space $\mathfrak{R}^{m_j}$. For each point from the modeling space $\Omega \subset E^n$, a mapping to the attribute value set N_j exists. Then, functional representation for an attribute A_j can be defined as [36]:

$$M_{A_{F_j}} = \left\{ (N_j, S_j) : A_j = \left\{ n_j \in N_j \mid n_j = S_j(X), X = (x_1, \dots, x_n) \right\} \in \Omega \right\}$$

where $S_j : \Omega \rightarrow N_j \subseteq \mathfrak{R}^{m_j}$ is a real-valued attribute function of point coordinates.

Functional representation used as the basic model for point set geometry and material distribution is efficient since point membership classification is done while performing the evaluation of real-valued functions [3]. Material interrogation is easy as attribute information can be determined by traversing a material attribute structure [36] or by directly evaluating explicit functions [3]. Functional representation can be used to model multi-material objects [35,36]; however, this representation needs to be enhanced or combined with other schemes to model more complex material distributions, such as those present in biological models.

1.2.3.3 Set Based Schemes

Set based schemes [24-26] employ subdivision of the interior of geometric solids into r-sets. Material properties are associated with each individual r-set in the subdivision. Since material and geometry domain are discretized into smaller elements, with varied material properties at each element, it is challenging to maintain smoothness across adjacent r-sets. Medical image data reveals a lot of complexity yet the biomaterials are varying smoothly across the interior of the biological entities.

The proposed heterogeneous solid model [24] consists of a finite number of material domains. Each material domain is modeled as an r-set and for each attribute an attribute function is defined on a particular decomposition to map the geometry model to the attribute model (Figure 1.10). Likewise, r-sets are used to represent the geometry of the solid model.

Figure 1.10: Set-based material model. An attribute function is defined for each material domain to map the geometry model to the corresponding attribute model [24].

1.2.3.4 B-Spline-Based Representations

NURBS models have been used almost exclusively in engineering design in fields such as visualization, analysis, manufacturing and prototyping. Due to a myriad of

desirable properties such as accuracy, speed, scalability and affine and projective invariance, NURBS models have found their way into bio-engineering.

A number of B-Spline methods have been proposed. Quin and Dutta [29] used B-Spline tensor product representation for heterogeneous turbine blade modeling. The existing B-Spline tensor representation was extended to represent material composition. Each point in parametric (u, v, w) point in parametric domain was coupled with a corresponding material composition point. Since the same set of points is used to represent geometry and the material composition, material properties coverage is limited and the modeling process is dependent upon a designer to specify the necessary material properties constraints.

Bhatt [27] proposed a B-Spline surface representation method to create slice models from CT scan data. Their approach makes use of the B-Spline hyperpatch tensor product representation proposed in [26]. CT images are manually processed and thresholded in order to identify the region of interest (ROI) in the scan. Using segmented CT data, for each voxel, the z coordinate is replaced by a material coordinate m , and the heterogeneous object that passes through both the geometry and the material composition points is constructed. The resultant model is suitable for analysis and fabrication and offers a significant data reduction. Similarly, Dutta and Qian [29] extend B-Spline methodology to introduce material properties. The concept of heterogeneous lofting is introduced to represent both the material composition of the heterogeneous object and its geometry.

Dutta and Qian [29] used a B-Spline representation to model both geometry and material composition of heterogeneous objects. The authors introduce the concept of

heterogeneous B-Spline curve and extend the idea to heterogeneous B-Spline tensor surfaces. Multi material objects are modeled interactively using heterogeneous lofting to interpolate through curves that combine both material features and geometry.

Martin and Cohen [30], on the other hand, proposed to separate geometric representation from attribute representation. In their approach, each attribute is represented as a trivariate NURBS volume and is independent of the geometry, which is also represented by NURBS trivariate volumes. While the material attributes and the geometry are separated, all volumes share the same parametric space [31]. In the paper, the authors described a new aggregate data type used to combine the geometry of the object with its attribute description.

Wu et al [93] proposed a NURBS-based volume model. An object is represented in a hybrid model including a NURBS representation and a voxel-based model. A voxel object can be generated through voxelization of its NURBS volume that is a generalization of NURBS representation of curves and surfaces.

An alternative approach to heterogeneous modeling using B-Spline representation scheme was proposed in [42]. Here, the authors combine functional representation (FRep) as the basic model for both object geometry and the attributes. FRep defined the object geometry directly through space partition, and a trivariate B-Spline primitive is used as a leaf in FRep constructive tree.

In [98], the authors attempt to model the heterogeneity of medical data by combining NURBS models to model geometry and voxel models to represent the interior or the models. A set of contours from sequential medical slices are generated and lofted

to produce a NURBS surface, and voxel data from the images is added into the data structure to fill in the interior of the model.

Despite the number of heterogeneous object modeling currently available, effective methodology to model complex heterogeneity has not surfaced [3]. Most of the proposed methods rely on the designer to control volume fractions of materials at each point. Regardless of whether the material composition is introduced as a separate dimension or coupled tightly with geometry, a huge number of degrees of freedom will be required in order to define a heterogeneous biomaterial. Biological structures generally exhibit complex material compositions, and while many of the proposed methods are capable of modeling simpler material distributions, they remain ineffective for heterogeneous biomodeling [3].

1.2.3.5 Function Description Methods

Under function description methods, the authors [23] include tensor product solids approach by Dutta [33] and constructive representation by Shin [34]. Under constructive representation scheme [34], heterogeneous solids are represented as compositions of simpler solids, extending the Constructive solid geometry (CGS) representation technique. The authors propose a corresponding data structure to keep track of the simpler models. The proposed constructive representation, according to the authors [34], is not capable of modeling arbitrary material distributions such as present in MRI and CT data.

1.2.4 Prior Works Concluding Remarks

Heterogeneous modeling has gained a lot of research interest due to continuous progress in all fields of science and the need for incorporating complex material properties of the models in order to further improve analysis, visualization, simulation and fabrication.

The above classification of the heterogeneous modeling approaches is a rough attempt to disambiguate, compare and contrast the research that is emerging across a number of fields. Evaluated models rely heavily on space subdivisions and thus can be suitable for representing irregular material distributions. Their accuracy remains tightly coupled with resolution, and memory requirements remain large. Voxel models are currently considered the state of the art in medical visualization and simulation due to good visualization properties and the relative ease of model construction. Unevaluated models, on the other hand, are often more compact and exact in data representation since they utilize exact geometric representations and rigorous functions to represent both the geometry and the material distributions [3]. As a result, these models are resolution independent, more concise and mathematically rigorous. However, generating unevaluated models capable of representing complex and irregular material distributions remains a challenging task.

A lot of the representation schemes available are capable of modeling objects with simple and regular material distributions, which is satisfactory for designing and manufacturing purposes of inanimate objects. However, modeling complex and irregular heterogeneities as demonstrated in biological materials remain a non-trivial undertaking and currently an unsolved task. While constructive approaches offer to solve the

complex material modeling problem by constructing the objects from simple primitives, each representing a different material, the resultant object remains a multi-material model [3] unable to capture and represent the unique and complex composition of biological materials. Biological materials possess too many degrees of freedom and such complex distributions when compared to man-made materials that fundamentally different representation schemes will be required to successfully model them. Oversimplifying these models and reducing them to multimaterial objects through, for example, individual attribute assignment, can result in a loss of important details and undermine the very essence of heterogeneous biological models and their applications.

As of today, no general and systematic approaches capable of modeling irregular heterogeneity has been reported [3]. Yet, modeling complex heterogeneous biological materials is becoming increasingly important across various fields. While visualization and geometrical models have greatly improved biomedical applications, in order to continue to advance a complete heterogeneous biological model that incorporates anatomy, physics and physiology is needed.

1.3 Contributions

Over the course of this research, we explore how to convert discrete medical data into a fitting continuous counterpart by using B-Spline function representation. Hence, the goal of this research is to investigate the use of B-Spline functions to as a basis for representing discrete medical data.

Medical imaging offers scientists and physicians an opportunity to explore the internal structures of patients, and as the result, the role of medical imaging has expanded beyond simple visualization and examination of anatomy. Despite the widespread use of medical imaging in areas such as surgical planning, intra-operative navigation and radiotherapy planning, the use of computer models suitable for analysis, simulation and exploring of medical data and the knowledge containing in that data remains limited [63].

Currently, no underlying continuous mathematical models are available for representing medical data [5]. A computational model depicts mathematically an organ or a tissue of the body, with the aim to not only to visualize, but also to reproduce the behavior of the various anatomical components. In order for these models to be useful, components such as chemical composition, shape, size, relative location and possible movement and deformation need to be successfully incorporated.

Thus, in this work, we researched and implemented the following:

- Computation of accurate contours from CT and MRI scans using B-spline curve approximation. Discrete pixel data representing organ boundaries was fitted with B-Spline curves, with the emphasis placed on shape fidelity, smoothness and accuracy.
- Fitting a continuous B-Spline surface to a density point cloud generated from discrete medical data. The resultant surface captured and preserved the essential tissue information using a concise mathematical form.

In the first part of the research, we investigated the problem of turning a discrete contour representation into a smooth B-Spline curve. Since extracted CT and MRI data is jaggy, not properly ordered and contains a lot of noise and represents shapes full of

intricate detail, traditional CAD models are not adequate to tackle the problem. The proposed algorithm takes advantage of the B-Spline-to-Bezier relationship in order to determine the right knot locations and the appropriate number of control points needed for fitting an optimal B-Spline curve.

Since a B-Spline curve defines the shape of the anatomical structures analytically, important properties such as organ volume and perimeter can be calculated.

In the second part of the research, we reduce the complexity of the medical image to a density point cloud, which essentially substitutes pixel intensities with density distribution of the point cloud. Thus, as a result, discrete representation is replaced with a smooth continuous surface. The resultant representation permits the representation of the surface at any resolution.

1.4 Organization of the Dissertation

In this dissertation we present the methodology we developed for applying B-Spline curve and surface representation scheme in order to model anatomical data contours and heterogeneity of the medical image data.

The remainder of the dissertation is organized as follows. Chapter 2 gives a brief overview of existing curve and surface representation schemes and focuses further on the B-Spline fundamentals. Data fitting with B-Spline curves and surfaces is discussed in more detail since these techniques are utilized heavily over the course of the research project. Chapter 3 focuses on the fundamentals of medical imaging and medical image

processing, and outlines challenges and open problems that exist. The connection between the tissue heterogeneity and pixel intensities is discussed.

Chapter 4 outlines the research done over the course of the research and writing of the dissertation. Relying on the fundamentals of B-Spline scheme representation and medical image basics, Chapter 4 discusses theoretical and design implementations of the following topics:

- Extracting and pre-processing of anatomic contours from medical image data
- Fitting of contour data with B-Spline curves utilizing the B-Spline-Bezier relationship in order to accurately capture the overall shape and the detail present in the anatomic boundary
- Generation of point cloud in order to reproduce density variations across medical image slice
- Fitting a B-Spline surface to the density point cloud. The resultant B-Spline surface is able to capture the density variations of the point cloud, while representing discrete point cloud set in a concise mathematical form.
- An important issue in surface fitting such as data explosion is addressed and solved by using ‘fuzzy knot control’ algorithm.
- Visualization and applications of the heterogeneous B-Spline surface are presented.

Chapter 5 concludes the dissertation with summary of the research and important outcomes and conclusions. Potential outcomes and directions for the research are outlined, with focus on further development of this research topic and future work.

Chapter 2 - B-Spline Modeling Fundamentals

The main focus of this chapter is on the basics of geometric modeling with B-Splines. Since the advent of major CAD systems, B-Splines and their extension, Non-uniform rational B-Splines (NURBS), have been an essential and indispensable standard in engineering design. Such appealing properties as accuracy, speed, ease of design manipulation, convex hull and projective invariance make B-Splines an excellent choice for biomedical modeling.

B-Spline surface and contour fitting allows representing a large amount of discrete medical data with a concise continuous form that is useful for later manipulation, visualization and analysis.

Flexibility, ability to realistically model anatomy, widespread use in industry, wealth of well researched computational algorithms all inherent to B-Spline representation scheme make B-Spline curves and surfaces an excellent basis for our heterogeneous modeling of medical data.

2.1 Fundamental Curve and Surface Representations

One of the main goals of geometric modeling is to represent real world objects through mathematical equations in order to simulate, visualize and analyze these objects

in a computer. These mathematical representations are not unique and various schemes, depending on the application, are used to represent the same object.

There are three main ways to represent curves and surfaces for geometric modeling – explicit equations, implicit equations and parametric functions. Each of these representations has its advantages and disadvantages, depending on the goal of the application.

2.1.1 Explicit Equations

The most basic definition of the curve in the xy plane, an explicit representation has the general form:

$$y = f(x)$$

where one value of y corresponds to each value of x . Thus, explicit functions generate y values from x values. Likewise, the explicit equation of the surface takes the general form:

$$z = f(x, y)$$

where just one value of z corresponds to each couple of values (x, y) .

This representation is simple to understand and to implement and is useful in an array of applications, such as plotting of graphs and functions. For example, an explicit relationship between amplitude, frequency, time and phase: $y = A \sin(\omega t + \varphi)$ is used extensively in mathematics, physics and electronics.

However, this representation cannot adequately represent multiple-valued and closed curves and constraints involving infinite derivatives. An explicit form of a curve cannot describe a curve in 3D space, and in general, the explicit representation is axis

dependent. As a result, explicit equations are of limited use to computer graphics or computer aided design.

2.1.2 Implicit Equations

The implicit representation of a curve lying in the xy plane has the general form:

$$f(x, y) = 0$$

which describes an *implicit* relationship between the x and y coordinates of the points lying on the curve. When compared with explicit representation, the implicit equations are capable of representing multi-valued functions and, as a result, a larger class of curves and surfaces.

By analogy to curves, an implicit equation of the surface takes the general form:

$$f(x, y, z) = 0$$

An example of an implicitly represented surface is the sphere of a unit radius centered at the origin, specified by the equation $f(x, y, z) = x^2 + y^2 + z^2 - 1 = 0$.

Using this representation scheme, problems such as determining if a given point is on the curve ($f(x, y) = 0$), or on either side of it ($f(x, y) > 0$ or $f(x, y) < 0$) are straightforward to solve. This form of representation has a variety of uses in computer graphics and CAD, usually for simpler, low degree cases. However, calculating the value of higher degree equations is computationally intensive since it requires solving non-linear systems of equations. Axis dependency remains an issue, making affine transformations difficult to perform.

2.1.3 Parametric Equations

A straight line in Euclidean space is a set of points p , that satisfy

$$p - p_0 = u(p_1 - p_0), \quad u \in (-\infty, +\infty), \quad p_0 \neq p_1$$

where p_0 and p_1 are arbitrary but distinct points on the line, and u is a parameter. As the parameter u takes all possible values within $(-\infty, +\infty)$, the point p traces the entire line.

Re-writing the equation after an algebraic manipulation yields the fundamental equation of linear interpolation:

$$p = (1-u)p_0 + up_1$$

It shows that

$$u = 0 \Rightarrow p = p_0$$

$$u = 1 \Rightarrow p = p_1$$

Furthermore, for $0 \leq u \leq 1$, a point p is at intermediate location between two end points, and the parametric equation of a line segment is:

$$p - p_0 = u(p_1 - p_0), \quad u \in [0, 1], \quad p_1 \neq p_0$$

The parametric equation of a curve lying in the xy plane has a general form:

$$x = x(t), \quad y = y(t), \quad a \leq t \leq b$$

Thus, in a parametric form, each of the coordinates of a point on the curve is represented separately as an explicit function of an independent parameter, making it coordinate system independent. Therefore, one can think of $C(t) = (x(t), y(t))$ as the path traced out by a particle as a function of time t in the time interval $[a, b]$.

The parametric representation of a curve is not unique. For example, the first quadrant of a circle of a unit radius centered at the origin can be defined by the parametric functions:

$$\begin{aligned}x(t) &= \cos t \\y(t) &= \sin t \quad 0 \leq t \leq \pi / 2\end{aligned}$$

The same quadrant of the circle can be represented alternatively as follows:

$$\begin{aligned}x(u) &= \frac{1-u^2}{1+u^2} \\y(u) &= \frac{2u}{1+u^2} \quad 0 \leq u \leq 1\end{aligned}$$

Parametric equations are axis independent, are capable of representing multiple-valued functions and infinite derivatives and have additional degrees of freedom when compared to either explicit or implicit representations. As opposed to explicit and implicit formulations, parametric equations are capable of representing a curve in 3D using a general form:

$$x = x(t), \quad y = y(t), \quad z = z(t) \quad a \leq t \leq b$$

Parametric representation for a surface requires two parameters, taking the following general form:

$$x = x(u, v), \quad y = y(u, v), \quad z = z(u, v) \quad a \leq u \leq b, c \leq v \leq d$$

Parametric form is more intuitive for designing and representing shapes in a computer. Due to the appealing properties of parametric functions, such as Bezier and B-Spline functions, algorithms for manipulating curves and surfaces are numerically stable and are capable of describing freeform shapes. As a result, the parametric representation is used intensively in computer graphics and computer aided design.

2.2 Polynomial Curves

Representing curves and surfaces parametrically has a number of advantages with respect to other existing representations. For geometric modeling purposes, it is imperative to restrict the coordinate functions to a class of functions that can be easily, accurately and efficiently processed in a computer, while not storage intensive and computationally efficient for computations of points on the curves and surfaces and their derivatives. The most widely used class of functions possessing the desired properties are the polynomials.

A d -dimensional polynomial curve of degree n is a parameterized curve $u \rightarrow P(u)$ of the form:

$$P(u) = \sum_{i=0}^n p_i u^i = p_n u^n + \cdots + p_1 u, \quad p_n, p_{n-1}, \dots, p_1, p_0 \in \mathbb{R}^d$$

where $P(u)$ is the set of all polynomials of degree less than or equal to n :

$P_n = \{p \in P : \deg(p) \leq n\}$. Functions u^i form a basis for polynomial curves. Thus, the general form for parametric representation of a curve is:

$$P(u) = \sum_{i=0}^n p_i X_i(u), \quad p_i \in \mathbb{R}^d$$

Functions $X_i(u)$ are called the *basis* functions. A number of basis functions have been invented, such as Bernstein polynomials and B-spline basis functions which are both used in computer aided design due to their appealing properties. Thus, parametric curves and surfaces used in geometric modeling typically are either polynomial or rational, i.e., the components of the representing function are either polynomials in the u and v variables or are a quotient of two polynomials on these parameters.

2.2.1 Bernstein Polynomials and Bezier Curves

A Bezier curve of order k is a degree is $k-1$ polynomial defined by k control points. A Bezier curve is a smooth interpolation of these k points. For example, a linear Bezier curve (order 2) is a linear interpolation of its two control points p_0 and p_1 :

$$p_{0,1}(s) = (1-s)p_0 + sp_1, \quad s \in [0,1]$$

Bezier curve of order 3 is a linear interpolation between the linear interpolation of control points p_0 and p_1 and the linear interpolation of control points p_1 and p_2 . This recursive method of interpolating between interpolations is described by De Casteljau's algorithm, which can be generalized to any order k using the following recursive formula:

$$p_{i,j} = (1-s)p_{i,j-1} + sp_{i+1,j-1}$$

A Bezier curve of degree n can be expressed in terms of basis functions:

$$C(u) = \sum_{i=0}^n B_{i,n}(u)P_i, \quad 0 \leq u \leq 1$$

Bernstein polynomials, used as Bezier basis functions, are the set of polynomials defined as follows:

$$B_{i,n}(u) = \binom{n}{i} \left(\frac{u-a}{b-a} \right)^i \left(\frac{b-u}{b-a} \right)^{n-i} = \frac{n!}{i!(n-i)!} \left(\frac{u-a}{b-a} \right)^i \left(\frac{b-u}{b-a} \right)^{n-i}, \quad u \in [a,b]$$

This also appears to be the form of the binomial distribution, utilized intensively in probability and statistics. The basis has a number of appealing properties such as partition of unity:

$$\sum_{i=0}^j b_{i,j}(s) = 1, \quad b_{i,j} \geq 0, \quad s \in [0,1]$$

The choice of basis functions determines the key geometric characteristics of the curve and surface representation scheme. Therefore, Bezier curves have a number of very useful properties for computer graphics and CAD [75]:

- Convex hull property: the curves are contained in the convex hulls of their defining control points, $\{b_i\}$
- Affine invariance: curves can be translated, scaled and rotated by applying the transformations to the control points
- Variation diminishing property: if a straight line intersects the curve in c number of points and the control polygon in p number of points, then it will always hold that $c = p - 2j$ where j is zero or a positive integer. The practical interpretation is that a curve which obeys this property will not “wiggle” more than the control polygon
- Endpoint interpolation: $C(0) = b_0$ and $C(1) = b_1$

Figure 2.1 illustrates a simple cubic Bezier curve exhibiting properties such as convex hull and interpolating through endpoints P_0 and P_3 .

Figure 2.1: Cubic Bezier curve.

For a cubic Bezier curve, there are four basis functions:

$$\begin{aligned}B_{0,3} &= (1-u)^3 \\B_{1,3} &= 3u(1-u)^2 \\B_{2,3} &= 3u^2(1-u) \\B_{3,3} &= u^3\end{aligned}$$

Intuitively, these blending functions define where the curve is positioned as a ‘blend’ of the four control points. For example, at $C(0)$ all the blending functions except $B_{0,3}$ are zero and thus only P_0 can affect the curve and the curve in fact interpolates the endpoint. At the position $u = 0.5$, all the control points have certain amount of influence on the shape of the curve.

2.2.2 B-Spline Basis Functions and B-Spline Curves

A single polynomial curve of degree d has only $d+1$ degrees of freedom. Thus, in order to interpolate or approximate a large number of points or to gain a finer control of the curve’s shape, a higher degree polynomial is required. High degree polynomials often have undesirable oscillations and exhibit poor computational behavior due to numerical errors associated with solving higher order systems of equations.

To summarize, curves consisting of just one polynomial segment have a number of shortcomings and thus are inadequate as the standard for CAD design systems [75]:

- A high degree is required for the majority of the applications since interpolation of n points requires $(n-1)$ -degree. High degree curves are inefficient to compute and are unstable numerically
- Complex shapes cannot be represented with low degree curves

- Lack of local control makes these curves unsuitable and cumbersome for interactive shape design

The alternative approach, to avoid the above stated issues, is to use piecewise polynomials. A piecewise polynomial $C(u)$ consists of m n^{th} -degree segments, which in turn are polynomial functions of a lower degree. The segments, $C_i(u), 1 \leq i \leq m$, are constructed such that they are joined with some level of continuity, and the parameter values, $u_0 = 0 < u_1 < \dots < u_m = 1$, map into the endpoints of the m polynomial segments.

A piecewise polynomial $C(u)$ has the following form:

$$C(u) = \sum_{i=0}^n f_i(u) P_i$$

where P_i are control points and f_i are piecewise polynomial functions forming a basis for the vector space for the set of all piecewise polynomials [75]. Since basis functions determine continuity and the overall behavior of the curve, the desired basis functions should possess useful and appealing analytic properties such as local support, convex hull, and variation diminishing and affine transformation invariance.

2.2.3 Basis Functions

Given $U = \{u_0, \dots, u_m\}$ -a sequence of real non-decreasing numbers, where $u_i \leq u_{i+1}, i = 0, \dots, m-1$ are referred to as knots, and U is the knot vector, the i -th B-Spline basis function of degree p is defined as:

$$N_{i,0}(u) = \begin{cases} 0 & \text{if } u_i \leq u < u_{i+1} \\ 1 & \text{otherwise} \end{cases}$$

$$N_{i,p} = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u_i}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

This recursive definition is known as the Cox-deBoor algorithm [94]. Figure 2.2 illustrates an example of third degree basis functions defined over uniform knots.

A B-Spline curve of order k is a piecewise k order Bezier curve. The key difference between the Bezier blending functions and B-Spline basis functions is that the number of control points for a B-Spline curve can be increased without increasing the degree of the curve [94].

B-Spline basis functions have nice geometric properties which define the desirable geometric characteristics of B-Spline curves and surfaces [75]:

- Local support property: $N_{i,0}(u) = 0$ if u is outside the interval $[u_i, u_{i+p+1})$
- In any given knot span $[u_j, u_{j+1})$, at most $p+1$ of $N_{i,p}$ basis functions are nonzero
- Nonnegativity: $N_{i,p} \geq 0$ for all i, p and u
- Partition of unity: for an arbitrary knot span $[u_i, u_{i+1})$, $\sum_{j=i-p}^n N_{j,p}(u) = 1$ for all $u \in [u_i, u_{i+1})$
- All derivatives of $N_{i,p}(u)$ exist in the interior of a knot span, where it is a polynomial
- At a knot, $N_{i,p}(u)$ is $p-k$ times continuously differentiable, where k is the multiplicity of the knot.

Since $N_{i,p}(u)$ is $p-k$ times continuously differentiable, increasing the degree increases the continuity, and, likewise, increasing knot multiplicity decreases the continuity (Figure 2.3 and Figure 2.4)

Figure 2.2: Third degree basis functions, uniform knots.

Figure 2.3: Third degree basis functions, double internal knots.

Figure 2.4: Third degree basis functions, full multiplicity in knots.

2.2.4 B-Spline Curve

A spline was originally a draughtsman's aid – a thin metal or wooden strip used to draw curves through fixed points by adding metal weights (called “ducks”). The resulting spline was threaded between the ducks (Figure 2.5), which resulted in a smooth curve that minimized the internal strain energy of the splines.

Figure 2.5: Spline threaded between ducks.

The concept of splines was first introduced by Schoenberg in 1946 [101]. In the paper, Schoenberg laid the mathematical foundations for splines and demonstrated how

they could be used to interpolate equally spaced samples of a function [102]. Despite the early introduction, splines did not take off until the early 1960s when the scientific community realized that these functions could be used to model the physical process of drawing a smooth curve. The intense interest that was generated was soon followed by applications in fields such as numerical analysis, approximation theory and various other branches of applied mathematics [102]. And with the advent of digital computers, splines caught attention of engineers and had a tremendous impact on computer graphics and computer aided design [102].

Similar to a Bezier curve, a B-Spline curve (Figure 2.6) does not pass through the control points. In terms of blending functions, a B-Spline curve is specified by:

$$C(u) = \sum_{i=0}^n P_i N_{i,k}(u)$$

where P_i is a set of control points and $n+1$ is the total number of $(k-1)$ -degree B-Spline basis functions. Each point on the curve is thus a weighted combination of a number of control points in the proximity of that point. These control points form a control polygon enclosing the curve. Therefore, a B-Spline curve at any point can be expressed in terms of the control points and the value of the parameter, u . Additionally, the curve is the result of mapping elements from the knot sequence onto Cartesian space.

Figure 2.6: A B-Spline curve is specified in terms of a knot vector, a degree and a set of control points.

Since B-Spline basis functions offer local support, meaning that they are nonzero only on a limited number of subintervals and not the entire domain $[u_0, u_m]$, moving a control point P_i affects the shape of the B-Spline locally, only on the subintervals where the basis functions are nonzero (Figure 2.7).

Figure 2.7: Modifying B-Spline curve by moving control point has local effect on the shape.

2.2.5 Knot Vector

A knot vector is a list of parameter values, or knots, that specify the parameter intervals for the individual polynomial curves that make up a B-Spline. For example, if a cubic B-Spline is comprised of four Bezier curves with parameter intervals $[1,3]$, $[3,4]$, $[4,7]$, and $[7,8]$, the knot would be $[t_0, t_1, t_2, t_3, 1, 3, 4, 7, 8, t_9, t_{10}, t_{11}, t_{12}]$

A knot vector describes the parametric space, and thus, knots subdivide the domain of the curve, not the curve itself.

Basis functions are defined over a knot vector and the shapes of the basis functions are determined entirely by the relative spacing between the knots. Individual knots are not meaningful by themselves but rather the ratio, or the knot interval between two adjacent knots. Since a knot interval in a knot vector describes the parameter length of a B-Spline curve segment, these ratios of knot values influence the mapping of parameter space to curve space.

For certain applications, it is desirable to introduce a lower degree of continuity at some point within the domain of the curve function. Since a B-spline consists of a sequence of polynomial spans joined at breakpoints or knots, discontinuity can be introduced by forming a multiple knot. At a regular breakpoint with singular knot, the degree of smoothness at its maximal value is C^{d-2} continuity. At a double knot, continuity is reduced to C^{d-3} , and generally, continuity at a knot with multiplicity m is C^{d-m-1} . Thus, when the number of duplicate knots equals the degree of the curve, a condition called full multiplicity knot forces a kink or a sharp corner on the curve by interpolating the control point (Figure 2.8).

Figure 2.8: Multiple knots and their influence on the shape of the curve.

There are some basic rules governing how the knot vector is defined. The values in the knot vector must always be in ascending order. Values can be expressed as an array of integers, from zero to some maximum value, since the ratios of the differences between the adjacent knots are important. Typically, knot vectors are normalized to the range $[0,1]$. Thus, knot vectors $[0,0,0,1,2,2,2]$, $[100\ 100\ 100\ 200\ 300\ 300\ 300]$ and $[-0.5\ -0.5\ 0\ 0.5\ 0.5\ 0.5]$ all produce the same curve.

The number of knots in the knot vector is always equal to the number of control points plus the order of the curve. Thus, a cubic (order=4, degree=3) curve defined over 4 control points will have 8 knots in the knot vector.

Knot vectors can be put into two groups: clamped and unclamped, and within each group, they can further be subdivided into uniform and non-uniform vectors. A clamped knot vector is a knot vector with the first and the last knot values repeated with multiplicity that is equal to the degree of the curve plus 1, $d+1$. The multiplicity in the knot vector forces the end points of the B-Spline curve to interpolate the first and the last control points.

2.2.6 Curve Degree

Most commonly used curves are degree 1 (linear), 2 (quadratic), and 3 (cubic). A curve with degree 1 is simply a polyline. Quadratic curves are often used to represent arcs and sections of an ellipse and cubic curves are used as free from design tools.

Curve degree defines the smoothness of the curve - the higher the degree, the closer the curve follows the control polygon, as evident from the Figure 2.9

Figure 2.9: B-Spline curves of different degrees defined over the same control points.

The degree of the curve also determines how much effect moving a particular control point will have on the local shape of the B-Spline curve. If the degree is high, moving a single control point affects more of the curve than if the degree is low (Figure 2.10).

Figure 2.10: The effect of control point manipulation on the local shape based on the degree of the curve.

The degree of the curve determines the maximum smoothness at the curve segment joints, and thus, affects the overall degree of continuity of the curve.

Continuity refers to the smoothness of the curve at the breakpoints. The most useful kinds of discontinuities are:

- C^0 -positional discontinuity, the curve has an actual break in it
- C^1 -tangential discontinuity, change of tangent forms a kink in the curve
- C^2 -curvature discontinuity, change in the acceleration as the particle moves along the path

As stated before, continuity at a breakpoint with a single knot is given by a relation C^{d-2} , from which it is evident that continuity is directly related to the degree of the curve. A linear curve of degree 1 thus has positional continuity at each joint. Increasing the degree of the curve also increases computational complexity and introduces numerical instabilities. A quadratic curve of degree 2 is capable of providing tangential continuity at most, and the cubic degree 3 curve offers C^2 continuity, which is adequate for most modeling and manufacturing applications.

2.3 Biparametric Cubic Surface Patches

The concept of joining piecewise cubic curve segments to form a continuous curve can be generalized to bi-parametric cubic surface patches [94]. Similarly to the definition of a piecewise polynomial, a point on the surface patch is defined by a parametric function and a set of blending functions defined for each parameter. For example, a cubic Bezier patch is defined as [75]:

$$P(u, v) = \sum_{i=0}^3 \sum_{j=0}^3 P_{i,j} B_{i,3}(u) B_{j,3}(v)$$

Mathematically, three-dimensional surfaces can be thought of as the Cartesian product of two curves.

For the Bezier patch definition, all the properties of the Bezier curves – the advantages as well as the shortcomings – are extended into the surface domain.

Figure 2.11: A control polygon and the resulting Bezier patch.

Joining Bezier surface patches is a straightforward extension of the curve-joining constraints.

B-Spline surface patches are related to B-Spline curves in the same way that the Bezier curves and Bezier patches are related. In terms of B-Spline basis functions, a B-Spline surface patch is defined as [75]:

$$P(u, v) = \sum_{i=0}^n \sum_{j=0}^m P_{i,j} N_{i,k}(u) N_{j,l}(v)$$

The shortcomings of Bezier curves are even more prominent in surface domain. Non-localness of Bezier formulation imposes very tight constraints on the surface design process. A Bezier patch is represented with 16 vertices which makes it adequate for designing shapes with simple topology. B-Spline patches, on the other hand, have no limit imposed on the number of vertices defining the characteristic polyhedron, and because of that, far more complex shapes are possible.

2.4 B-Spline Surfaces

The tensor product B-Spline surface of degree p in the u direction and degree q in the v direction is defined as the piecewise polynomial surface [75]:

$$S(u, v) = \sum_{i=0}^n \sum_{j=0}^m P_{i,j} N_{i,p}(u) N_{j,q}(v) \quad 0 \leq u, v \leq 1$$

where $\{P_{i,j}\}$ form a bidirectional control net, and the $\{N_{i,p}(u)\}$ and $\{N_{j,q}(v)\}$ are the non-rational B-Spline basis functions defined on the knot vectors:

$$U = \left\{ \underbrace{0, \dots, 0}_{p+1}, u_{p+1}, \dots, u_{r-p-1}, \underbrace{1, \dots, 1}_{p+1} \right\}$$

$$V = \left\{ \underbrace{0, \dots, 0}_{q+1}, v_{q+1}, \dots, v_{s-q-1}, \underbrace{1, \dots, 1}_{q+1} \right\}$$

where $r = n + p + 1$ and $s = m + q + 1$.

Figure 2.12: Bicubic B-Spline surface consisting of 3x3 patches together with its control net.

A B-Spline surface is a network of patches all of which are cubic in each of the parametric directions exhibiting first- and second-order continuity (Figure 2.12). The surface is defined as a $n \times m$ net of control points which has both u and v dimension. The desirable geometric characteristics of B-Spline curves resulting from the properties of B-Spline basis functions also extend well into the B-Spline surface domain, thus making B-Spline surfaces a powerful and complex representation scheme:

- Corner point interpolation: e.g. $S(0,0) = P_{0,0}$
- Affine invariance
- Convex hull
- Local modification: moving $P_{i,j}$ affects the surface shape only on the rectangle

$$[u_i, u_{i+p+1}) \times [v_j, v_{j+q+1})$$

2.5 Modeling with B-Spline Curves and Surfaces

When selecting a representation scheme for objects, the mathematical power of the scheme and the intuitiveness of design are two fundamental aspects to be considered. B-Splines have become wide spread and have become *de facto* standards in engineering design, have been implemented in major CAD systems available nowadays. CAD systems have taken advantage of many desirable and rather extraordinary geometric properties that B-Spline representation scheme offers. For example, since B-Splines form a local basis, a curve or a surface is confined to a convex hull smaller than the one formed by the entire set of control points, since only those control points that are associated with nonzero coefficient functions can contribute to the definition of the curves and surfaces. Thus, the local convex hull consists of only a handful of control points. For a cubic curve, only four successive control points are involved in the definition of a particular polynomial span; likewise, a grid of 16 control points completely defines a bicubic patch of the cubic B-Spline surface [75]. Geometrically, this implies that moving a control point on either a B-Spline curve or surface will only cause a small perturbation, and beyond the affected area, the curve or the surface remain unchanged. This property makes B-Spline representation particularly attractive to designers using CAD tools.

As mentioned earlier in this chapter, B-Spline basis functions offer affine invariance as one of their properties. Since a B-Spline curve or surface are a geometric entity defined by a convex combination of a locally involved control points and basis functions, an affine transformation is applied to control points, which preserves the shape of the defining control polygon. Thus, a transformation is performed simply by

multiplying the control points $P_{i,j}$ with an appropriate transformation matrix M , which includes translation, rotation and scaling: $P_{i,j}^{transformed} = MP_{i,j}$.

Variation diminishing property of B-Splines is also a very appealing property for modeling since it guarantees that a B-Spline curve or surface will not exhibit more ‘wiggles’ than the control polygon itself.

The mathematical definition which might seem intimidating at first provides a lot of freedom and flexibility. Thus, a knot vector can be a sequence of numbers provided they are an ordered non-decreasing sequence of real values. A uniform knot vector with evenly spaced knots is the simplest way to define a B-Spline curve or surface. Basis functions defined over such a knot vector are translates of each other [95] and, for practical purposes, do not need to be recomputed and can be stored in a table in advance. Generally, a designer is not involved in manipulating the knot vector and most commercial CAD systems such as Rhino or Maya make knot vector available to the designer on demand only, thus abstracting away the unnecessary complexity.

Incorporating knowledge into B-Spline representation scheme is an area of active research, promising to further improve the robustness and reliability of CAD modeling [111,112].

2.5.1 Data Fitting

Curve and surface fitting is the process of representing large amounts of data with a concise mathematical form suitable for later manipulations [96].

Data fitting, in this context, is a construction of B-Spline curves and surfaces which fit an arbitrary set of discrete geometric data such as a point cloud of Euclidian

points or derivative vectors. Fitting techniques can be broadly classified into two categories: approximation and interpolation. As the name implies, interpolation scheme results in curves and surfaces that interpolate the measured data at each point. Hence, when an interpolating curve or surface is constructed, they pass through the data points precisely as seen in Figure 2.13(a). Interpolation is used when function values at the measured points are known to a high precision and it is important to preserve these values by a fitted function, if possible [96].

Figure 2.13: Example of interpolation and approximation. Interpolating B-Spline curve passes through the points (a) while approximating curve aims at ‘best fit’ to the given points (b).

Approximation techniques, on the other hand, aim at finding a desired best fit to data. During approximation, curves and surfaces do not necessarily pass through the given discrete data, but rather approximate it, often in a least squares sense as shown in Figure 2.13(b). Approximation is often a desirable fitting technique when the point set is large and/or contains a lot of computational noise. Interpolating such data sets can become cumbersome, computationally expensive and computationally unstable [96]. In such cases, approximation allows to capture and describe the overall shape of the data without having to pass through every single point – which might not be desirable when

abundant amount of noise is present. Among available approximation methods are Least Squares, Least Median of Squares, Maximum Likelihood and Spline smoothing [96].

Given an input of geometric data such as points or derivatives, the output of a B-Spline data fitting algorithm is usually comprised of control points and knots that describe either the curve or the surface sought. Constraints such as degree p of the curve, or (p, q) degrees for surfaces, are required. If a certain degree of continuity C^r is enforced, then the chosen degree p for curve fitting must satisfy $p \geq r + 1$. For approximation, the number of control points and fitting tolerance are usually required.

There are many approaches and minor tweaks to data fitting problem, with many of the algorithms taking a heuristic approach [74]. The central issue is that given discrete geometric data to be either approximated or interpolated, none unique solution is obtained.

Since medical data is notorious for noise and abundance of data, approximation of data is of particular interest in this research, specifically global curve approximation and surface lofting.

2.5.2 Global Curve Approximation

While the number of control points for interpolation is automatically determined based on the input degree and other possible constraints, it is not known in advance how many control points are required for approximation, to guarantee a fit within a desired tolerance. As the result, approximation methods are typically iterative. Two general approaches exist when approximating a set of points [74]:

- Start with a minimum number of control points:
 - Fit the approximating curve or surface
 - Evaluate deviation of the fitted curve or surface from the data
 - If deviation is acceptable, or within the specified tolerance, fit is complete
 - Otherwise, increase the number of control points and reiterate the process
- Start with a maximum number of control points:
 - Fit an approximating curve or surface
 - Evaluate deviation of the fitted curve or surface from the data
 - If deviation is acceptable, or within the specified tolerance, fit is complete.
 - Otherwise, decrease the number of control points and reiterate the process

2.5.2.1 Least Squares Approximation

Least squares fitting technique generally involves computing the error of the fitted curve or surface from the measured point set and minimizing the square of the error in some sense [96]. A number of approaches to solving least squares problem exist, including normal equations, a regression approach, etc. [96].

Given a degree $p \geq 1$, number of control points $n \geq p$ and a set of points

$Q_0, \dots, Q_m, m > n$, a degree p curve is sought:

$$C(u) = \sum_{i=0}^n N_{i,p}(u) P_i, \quad u \in [0, 1]$$

satisfying the following:

$$Q_0 = C(0) \text{ and } Q_m = C(1)$$

The remaining data points are approximated in the least squares sense:

$$\sum_{k=1}^m |Q_k - C(\bar{u}_k)|^2$$

so that the sum is a minimum with respect to $n+1$ variables. Parameters $\{\bar{u}_k\}$ are a pre-computed set of parameters.

Applying a standard technique of linear least squares fitting yields the system of $n-1$ equations in $n-1$ unknowns:

$$(N^T N) P = R$$

where N is the $(m-1) \times (n-1)$ matrix of scalars:

$$N = \begin{bmatrix} N_{i,p}(\bar{u}_1) & \cdots & N_{n-1,p}(\bar{u}_1) \\ \vdots & \ddots & \vdots \\ N_{i,p}(\bar{u}_{m-1}) & \cdots & N_{n-1,p}(\bar{u}_{m-1}) \end{bmatrix}$$

R the vector of $n-1$ points:

$$R = \begin{bmatrix} N_{i,p}(\bar{u}_1) R_1 + \cdots + N_{n-1,p}(\bar{u}_{m-1}) R_{m-1} \\ \vdots \quad \ddots \quad \vdots \\ N_{i,p}(\bar{u}_1) R_1 + \cdots + N_{n-1,p}(\bar{u}_{m-1}) R_{m-1} \end{bmatrix}$$

and

$$P = \begin{bmatrix} P_1 \\ \vdots \\ P_{n-1} \end{bmatrix}$$

The set up of these equations requires a knot vector $U = \{u_0, \dots, u_k\}$ and parameters $\{\bar{u}_k\}$. Parameters can be computed using one of the existing parameterization methods, such as uniform or chord-length, depending on the application.

In distance sense, the approximation and the capture of essential detail improves as the number of control points increases. Yet, as the number of control points approaches the number of data points, undesirable wiggles and shapes in the fitted curve can occur, especially if the data set is dense and exhibits a lot of noise.

In a separate research project, we approximated impedance spectra dataset with a B-Spline curve. The datasets collected from each experiment were very dense, each having 16730 data points (Figure 2.14(a)). Setting the number of control points to 3% of the dataset, which resulted in 5 control points, produced the curve that did not capture main characteristics of the dataset in favor of the more predominant features (Figure 2.14(b)). Increasing the number of control points to 85 control points (Figure 2.14(c)) enabled the B-Spline fit to capture the overall shape of the dataset more adequately. Setting the number of control points to 5729, or 35% of the dataset, produced an approximation with an excellent capture of the detail (Figure 2.14(d)). This example illustrated very well the behavior of the global approximation fit and the influence of the number of control points on the fitting process.

Figure 2.14: Impedance spectra dataset with 16370 points (a). The dataset is approximated with B-Spline curves with various number of control points. Number of control points is set to 5 (b), 85 (c) and 5729(d).

2.5.3 Surface Lofting

Lofting is one of the oldest methods which has existed since the early days of CAD [96]. Lofting derives its name from large architectural spaces, or lofts, where ship hull designing took place. This method deals with fitting a smooth patch through a set of parametric cross sectional curves.

Given a set of cross sectional curves $\{C_k(u)\}, k = 0, \dots, K$, lofting is thus a process of blending these curves to form a surface (Figure 2.15). Most commonly, lofted surface interpolated through the cross-sectional curves, but depending on the application, approximation can also be used.

Figure 2.15: Lofted surface.

When interpolation is used, the section curves become the isoperimetric curves on the fitted surface. Based on B-Splines, lofting can be defined as follows [97]:

$$\begin{aligned}
 C_k(u) &= S(u, \bar{v}_k) = \sum_{i=0}^m \left\{ \sum_{j=0}^n V_{i,j} M_{j,q}(\bar{v}_k) \right\} N_{i,p}(u) \\
 &= \sum_{i=0}^m B_{k,i} N_{i,p}(u) \quad k = 0, \dots, n
 \end{aligned}$$

where the $(m+1)$ points $B_{k,j}$ form the control polygon of the k^{th} B-Spline curve $C_k(u)$.

All the $C_k(u)$ curves are assumed to be the u parameter curves of a B-Spline surface $S(u, v)$, and so, defined over the same knot vector U and have the same degree p . The $(m-1) \times (n-1)$ control points are determined by applying B-Spline curve interpolation $(m+1)$ times.

Since curves are expected to have the same degree and knots, the curves are checked for compatibility before the lofting operation, and made compatible if needed. The compatibility process raises the number of control points. Thus, if k cross sectional curves have, on average, n control points each, the compatibility process for lofting interpolation can produce as many as $O(k^2 n)$ control points, instead of the expected

$O(kn)$ [89]. Approximation to the cross sections is an alternative approach, promising to alleviate the issues introduced by compatibility process in the course of interpolation.

Chapter 3 - Modeling in Medical Applications

Medical imaging field has been evolving rapidly with the advent of faster, more accurate medical imaging devices. Nowadays, CT and MRI scanners are used extensively in medical practice and generate an abundant amount of medical data for medical imaging community to explore. With the availability of medical data came the need to improve and automate medical image processing techniques.

Section 1 reviews some of the key imaging modalities, and highlights the advantages and disadvantages of each. Understanding the basic physics of each modality and the limitations is important in order to develop algorithms and modeling methods based on the data.

Section 2 briefly covers medical image processing fundamentals and the methods that are of particular interest to our research. Despite extensive research, the fundamental problems in medical image processing remain unsolved [47] and existing algorithms remain open to improvement in terms of accuracy and automation. Segmentation and boundary extraction, in particular, have consistently remained an ad hoc procedure, requiring significant user input and interaction for best results [47].

The idea of generating 3D models from a stack of CT images was first published in 1980 and since then has generated a wealth of interest across various research fields. Computational models are an invaluable tool for achieving scientific progress in

experimental, biomedical and clinical research. Section 3 briefly describes the motivation behind the computational anatomical models and focuses on the current state of the art, 3D anatomical voxel models.

3.1 MRI/CT Fundamentals and Data Acquisition

Wilhelm Roentgen discovered X-rays in 1895 and published the radiograph of Mrs. Roentgen's hand in 1898, which was the first time the world had seen the interior of the human body visualized non-invasively (i.e., without the use of surgery). The discovery immediately prompted the "X-Ray mania" all over Europe and USA [47] and today medical imaging modalities such as X-Ray computed tomography (CT) and Magnetic Resonance Imaging (MRI) are an essential and routine part of medical diagnosis and treatment [47].

Since the first radiograph at the end of 19th century, imaging technology has improved considerably. New modalities such as MRI (1980s) and Ultrasound (1960s) have been introduced and have continued to improve in terms of safety and image quality. The availability of medical imaging modalities has greatly improved our understanding of human anatomy disease progression and treatment processes. Information obtained from medical imaging data is currently used to diagnose a disease, improve and customize the treatment process and provide vital clues about the progression and response of the disease to the treatment. Serial images are regularly used to create a 3D view of the anatomy for educational and medical treatment purposes.

3.1.1 X-Ray Computed Tomography

CT was the first imaging modality to offer detailed view of the internal anatomy in three dimensions [49]. The foundation of CT image generation is the measurement of x-ray attenuation along a line between an x-ray source and an x-ray detector [47]. A 2D cross-sectional image is reconstructed following the collection x-ray attenuation coefficients along all lines within the cross-section.

Seven generations of basic CT scanners have been introduced, with each consequent model introducing faster acquisition of data and higher image quality. Helical CT was developed in 1980s to address the growing need for fast volumetric data acquisition [48]. Increasing the speed of image acquisition not only decreases the amount of ionizing radiation the patient is subject to, but also improves the image quality by eliminating and reducing image artifacts resulting from patients moving and breathing.

In a helical CT scanner, x-ray source and detectors rotate continuously. The patient table is set into forward motion sliding the patient through the source-detector plane (Figure 3.1). As the result, as the name of the scanner implies, the position of the source carves out a helix with respect to the patient [48].

Figure 3.1: Patient table is continuously translated through source-detector plane while X-ray source tube is rotating and acquiring projection data.

In computed tomography, a number of 2D radiographs are acquired by rotating the X-Ray tube around the patient's body. Full 3D image is built using a reconstruction algorithm. The reconstruction algorithm used depends on the geometry of the scanner – parallel-beam, fan-beam, helical-scan, or cone-beam [48], and *filtered backprojection* is the most commonly used one. Thus, CT methodology is essentially a 3D version of X-ray radiography [47] and as such, offered high contrast between bone and soft tissue and low contrast between soft tissues.

Figure 2.3 illustrates the relationship between 3D, 2D and 1D projections. 3D computed tomography involves reconstruction of an object from 2D projections taken from multiple views around the object. Each projection at a given angle is composed of numerous line integrals through the objects in the direction specified by that angle. 1D projections are used to reconstruct an object slice by slice. 3D objects are traditionally reconstructed as a stack of 2D slices.

In a CT system, the x-ray tube makes a short burst of x-rays that propagate through a cross-section of a patient. The detector detects the exit beam intensity which is integrated along a line between the x-ray source and each detector [48]. Thus, the integrated x-ray intensity at any given detector is given by [48]:

$$I_d = \int_0^{E_{\max}} S_0(E) E \exp \left[- \int_0^d \mu(s; E) ds \right] dE$$

where $S_0(E)$ is the x-ray spectrum and $\mu(s; E)$ is the linear attenuation coefficient along the line between the source and the detector. Since integration over energy is mathematically intractable for CT image reconstruction [47], the concept of effective energy, $\bar{E}$, is introduced. As the result, the basic projection measurement ς_d is defined as [48]:

$$\varsigma_d = \int_0^d \mu(s; \bar{E}) ds$$

which leads to the important observation that the basic measurement of a CT scanner is a line integral of the linear attenuation coefficient at the effective energy of the scanner.

Figure 3.2: Schematic representation of the steps involved in 3D computed tomography.

3.1.1.1 CT Numbers

Effective energy varies between different CT scanners and is dependent on the type of x-ray tube used. This in turn means that the same object will produce a different value of μ on different scanners. Thus, to compare medical data produced from different CT scanners, the numbers are calibrated and expressed as *CT numbers*.

A CT scanner measures and reconstructs the value of coefficient μ at each pixel. CT number of a material, given the effective energy of the source beam, is computed from the linear attenuation coefficient of the material and is calculated using the following relation:

$$CT\ number\ h = \left[\frac{\mu_m - \mu_w}{\mu_w} \right] * a$$

where μ_m and μ_w are linear attenuation coefficients for the material and water, respectively, and a is a scaling factor.

CT numbers are expressed in Hounsfield units (HU). Hence, HU scale is a linear transformation of original coefficient measurement where radiodensity of distilled water at standard pressure and temperature (STP) is defined as zero. Since $\mu = 0$ for air, it is $h = -1000$ HU for air. Typically, largest CT numbers are found in the bone, where $h \approx 1000$ HU.

3.1.1.2 CT Safety and Image Quality

Because CT technology is based on the use of multiple radiographs, ionizing radiation can have harmful effects on the human body and should be taken into account. The body of the patient absorbs x-rays and because of that only a reasonably safe dose

can be administrated. Since a CT detector measures the intensity of the x-ray pulse after the remainder of it exits the patient, lower x-ray doses decrease the number of detected protons and, as a result, limit the signal-to-noise ratio (SNR).

Arbitrary high resolution is not possible to achieve due to the finite nature of detectors and the sampling limitations [48]. The most commonly employed reconstruction algorithm is a filtered backprojection [48]:

$$f(x, y) = \int_0^\pi \left[\int_{-\infty}^{\infty} |\zeta| G(\zeta, \theta) e^{j2\pi\zeta l} d\zeta \right]_{l=x\cos\theta+y\sin\theta} d\theta$$

where the term $|\zeta|$ is known as the ramp filter. The ideal ramp filter is not realizable and an approximation filter $W(\zeta)|\zeta|$ with window function $W(\zeta)$ is used instead. Also, the finite size of detectors prevents the line integrals from being imaged exactly and the additional filter $S(\zeta)$ is used to locally integrate the underlying signal [48]. As a result, the filtered backprojection is transformed into the approximation, where $\hat{f}(x, y)$ is the blurred version of the original function.

$$\hat{f}(x, y) = \int_0^\pi \left[\int_{-\infty}^{\infty} G(\zeta, \theta) S(\zeta) W(\zeta) |\zeta| e^{j2\pi\zeta l} d\zeta \right]_{l=x\cos\theta+y\sin\theta} d\theta$$

Thus, high resolution is hard to achieve in both the number of projections and the number of ray-paths per single projection [48].

Images produced by a CT scanner can suffer from a number of artifacts. For example, if undersampled, the reconstructed projections will appear aliased. These aliasing artifacts will propagate throughout the image reconstruction process and will manifest themselves in the final image. The most apparent such aliasing effect is a *streak artifact* [48], often emanating from object boundaries at points of small curvature. Beam

hardening is a phenomenon caused by energy selective attenuation of x-rays in the human body resulting in low-energy photons to be more readily absorbed. Due to the phenomenon, propagating beam is intensified in high-energy photons and produces a halo effect, often apparent in head scans.

Partial volume averaging artifact (PVA) is a very common artifact in daily practice. PVA artifact arises when the voxels in the imaging matrix are larger than the structures being imaged. When it occurs, neighboring tissues from different densities are averaged and included in the same voxel. PVA artifact can be especially a problem when assessing small lesions or structures for diagnostic purposes. This artifact is common to both CT and MRI imaging. To remedy the problem, the spatial resolution can be increased which comes at the expense of time and SNR.

Other artifacts occasionally present in CT images include motion, x-ray scatter and electronic drift. X-ray scatter results in convolution blurring and subsequent blurring of the image. A complete scan takes anywhere between 1 and 10 seconds [48], breathing and heartbeat cycles cause motion and the final image artifacts associated with it.

Spatial positions of the internal structures are encoded based on the X-ray positions that hit the detector. Factors such as magnification caused by the diverging beam and overlaying structures can negatively affect the encoding of the correct spatial information, and as a result, radiologists must rely on prior knowledge in order to correctly read corrupted CT data [48].

While methods for preprocessing the data or post-processing the images to eliminate and correct the artifacts, radiologists generally know to expect these artifacts and interpret them accordingly [48].

3.1.2 Magnetic Resonance Imaging

Magnetic Resonance Imaging (MRI) relies on the phenomenon called nuclear magnetic resonance (NMR) to generate the images. Since nuclei are comprised of protons and neutrons, they possess positive charges. If the nucleus has either an odd atomic number or an odd mass number, the nucleus is NMR-active. An NMR-active nucleus has an angular momentum Φ and possesses a spin (Figure 3.3).

Figure 3.3: Nucleus visualized as a ball spinning about an axis.

Collections of identical nuclei form nuclear spin systems. Since certain atoms such as ^1H , ^{13}C , ^{19}F and ^{31}P are abundant in biological systems, their nuclear spin systems can provide an NMR signal strong enough to be detected and be distinguishable from the background noise [48]. MR imaging focuses on ^1H atoms since they are present throughout the body in high density and give out a very strong NMR signal.

Each spinning nucleus also possesses a microscopic magnetic field. The microscopic magnetic field has a magnetic moment vector μ defined as [48]:

$$\mu = \gamma\Phi$$

where γ is the *gyromagnetic ratio*.

A spin system within a volume of material becomes magnetized when an external static magnetic field B_0 is applied. Magnetization vector M is a sum of N_s of individual magnetic moments:

$$M = \sum_{n=1}^{N_s} \mu_n$$

The volume of material in the MRI system is a small volume of tissue – represented in imaging system as a voxel. Two main factors determine the value at each voxel – the tissue property and the scanner image protocol [48].

M is a function of time, and can be manipulated by using external radio-frequency excitations and magnetic fields. The magnetization vector $M(t)$ has two components: longitudinal and transverse. Longitudinal component is oriented along the axis defined by the static magnetic field and the transverse component – orthogonal to the direction of the static magnetic field.

In order to generate an observed signal in MRI, the following steps occur. Rapidly rotating transverse magnetization creates a radio frequency (RF) excitation within the sample. RF excitation induces a measurable amount of voltage signal, in the coil of wire located outside the sample.

Hence, the sample being imaged is briefly exposed to a burst of radio-frequency energy, which, in the presence of the magnetic field puts hydrogen nuclei in an elevated energy state [47]. As the molecules go through their microscopic tumbling, they shed the acquired energy and return to their equilibrium state. The process of the molecules returning to their original surroundings is referred to as relaxation. Relaxation rates are

different for different tissues, which allows to image and contrast the patient's interior by measuring the relaxation rates.

3.1.2.1 Instrumentation and System Components

There are five principal components that comprise an MRI scanner: (1) main magnet, (2) set of coils to provide a switchable spatial gradient in the magnetic field, (3) resonators for transmission and reception of radio-frequency pulses, (4) electronics for programming the process of transmission and receiving of signals, and (5) a console for viewing and storing images [47]. The schematics representation is illustrated in Figure 3.4.

Figure 3.4: Block diagram of MR scanner components [47].

In CT, the spatial positions of objects are acquired using the positions of X-rays that traverse through the body and hit the detector. In MRI, gradient coils are used to encode the spatial positions. The main function of the gradient coils is to generate a temporarily change in the magnitude B_0 of the main magnetic field. Gradient coils allow choosing the slices of the body for selective imaging, and more importantly, they provide the means to spatially encode pixels within the chosen image slice. Echoes coming from individual pixels can be decoded and turned into an image.

Figure 3.5: MRI image slices.

The patient is placed in the magnet and once the exam begins, the patient's hydrogen nuclei align in the direction of the external magnetic field. The magnetization strength for each volume element (voxel) is sampled by transmitting a radiofrequency (RF) pulse that carries a magnetization component in the plane transverse to the external magnetic field [51] (which can be thought of as a XY plane and z-axis representing the direction of the magnetic field). As the hydrogen nuclei rotate in the transverse plane, the resulting component of the spin magnetization generates a return RF signal, or 'echo'. This echo is sampled in order to generate the next line of voxel data and the sampling

process is repeated either 128 or 256 times until the image data is generated [51]. Figure 3.5 shows two examples of MRI image slices that are used in our working dataset.

3.1.2.2 MRI Image Quality

The MRI image data quality relies primarily on sampling process, noise and contrast. Contrast depends on the choice of pulse sequence that can be controlled and the inherent tissue parameters.

The data acquired from MR imaging pulse can be thought of as scans of Fourier transform [47] and thus the reconstruction algorithm relies on the inverse Fourier transform to produce the image.

An MRI image is essentially the sampling of the 2D Fourier space of the effective spin density in a selected plane [47]. The sampling of the baseband signal uses an analog-to-digital converter with a specified receiver bandwidth (RBW). RBW uses an antialiasing filter which cuts off the frequencies outside the bandwidth interval. This as a result causes signals coming outside that bandwidth interval to be lost.

The nature of the Fourier space sampling instead of the image space also causes the image wraparound in image space [47], and a radiologist interpreting the data has to be aware of that.

Statistical fluctuations of the signal cause noise in MRI. Artifacts such as geometric warping and ghost streaks can be present in the final image. Some of the geometric warping distortion can be corrected by image post processing software. A wrap around artifact is a common artifact that occurs when the field of view is smaller than the anatomical structures that are being imaged. As the result, the anatomical areas which are

not included in the field of view ‘fold over’ to the opposite end of the image. Simplest way to prevent the wrap around is to increase the field of view, often at the expense of loss of image resolution.

Ghost artifacts arise due to the changes in the object in caused by patient motion such as breathing or heart beat. As the result, ghost artifacts are the most common artifacts present in MRI images.

MRI offers a number of advantages over CT due to the excellent contrast to noise ratio (CNR) which allows for better distinguishing between various tissues as well as between the normal and abnormal and diseased structures [51]. MRI systems are capable of generating multiplanar images in sagittal, coronal and transverse planes which allows better 3D reconstruction and cross-referencing, and does not expose the patient to ionizing radiation. In recent years, MRI technology has been extended to perform more specialized techniques such as magnetic resonance angiography, diffusion-weighted imaging (DWI), diffusion tensor imaging (DTI) and functional imaging.

3.2 Medical Image Processing Fundamentals

With the advent of faster, more accurate and less invasive medical imaging modalities and the availability of medical data to researchers, the field of medical imaging processing has been rapidly evolving and processing in the past few decades.

Regardless of the medical imaging modality used, medical images can be viewed as geometrically arranged arrays of data samples that store parameters corresponding to various physical phenomena ranging from levels of hemoglobin and oxygen saturation to

linear attenuation coefficients of tissues. Although used extensively for visualization and diagnosis, this medical data is also essential for building mathematical models for further processing, analyzing and simulation.

3.2.1 Manual Segmentation

Reading and interpreting medical image data requires highly trained and experienced technicians and clinicians, who bring their prior knowledge and expertise into medical data analysis and diagnosis. Manual segmentation, however, is a laborious and time consuming process that is subject to error. Images from large datasets such as the Visible Human Project Dataset [113] are often segmented by different people, which lead to discrepancies. The reported discrepancies range from 5% for larger organs to 24% for more complicated organs such as esophagus, and even higher for small anatomical structures [50]. At the result, manual segmentation is prone to subjectivity and human fatigue, which inadvertently causes segmentation errors.

3.2.2 Image Processing Basics

Due to the vast amount of medical data, in order to build accurate mathematical models, a range of post processing techniques for image analysis and feature extraction are required. Medical images offer a wealth of information regarding the patient's interior, yet they suffer from a number of imperfections [50]:

- Low resolution both in the spatial and spectral domains
- High level of noise
- Low contrast

- Image artifacts

These imperfections can stem from the imaging modality itself (e.g. ultrasound is inherently noisy and implants with metal in them will cause artifacts in MRI) or be a result of a tradeoff between image quality and time to acquire the images [50].

With any choice of a medical image modality, the space is sampled and reconstructed mathematically, and as a result, accuracy of the reconstructed image is limited by accuracy of the reconstruction algorithm and the imaging modality resolution. A standard CT procedure generates more than 2 million voxels per patient examination [5], and other scanning techniques such as MRI and ultrasound produce very similar amount of data. Vast amount of data produced, up to 35 megabytes (MB) per patient [5] hinders rapid image formation. Hence, among the challenges posted to computer scientists involved in medical image based modeling and post-processing are the development of rapid and accurate manipulation techniques capable of producing models useful to a physician.

An image is a map $I: D \rightarrow C$, where for any point x in the domain D the mapping I associates a “color” $I(x)$ in a color space C . For the sake of brevity, the discussion will be restricted to the case of a two-dimensional grayscale image which can be thought of as a function from the domain $D = [0,1] \times [0,1] \in \mathbb{R}^2$ to the unit interval $C = [0,1]$.

Although medical image segmentation is a relatively established and mature research field, fully automatic segmentation of medical data remains an unsolved problem [50]. Due to inherent noise in the data, complexity of anatomical shapes, overlap

of gray level values between neighboring tissues and organs and a lack of distinct boundaries, segmentation is often performed manually.

Image segmentation and edge extraction for further 3D model construction is often done using a commercial software package (e.g. MIMICS), which involves manually thresholding and editing each slice individually, as the result of automatic edge detection often provides a poor result (Figure 3.6).

Figure 3.6: Automatic edge detection using MIMICS.

The optimum medical image based application should be robust in the face of medical data imperfections, fast, simple for a medical specialist to use, as automatic as possible and, most importantly, accurate [47].

Due to the vast amount of data that needs to be processed, when processing medical image data for disease diagnosis, accuracy is often preferred over speed. The requirement for accurate medical image processing stems from the need to formulate a diagnosis. Accuracy is the main concern since the medical data displayed must provide visual information about the patient that is trustworthy. A physician forms their diagnosis based upon the discrepancy from the norm [5] and hence inaccurate data and models built

based on it are not tolerated. When building models based on medical image data, there are two main components to the accuracy: data collection and data post-processing. Data collection is the field a radiologist is concerned with and reliable data acquisition depends on the variables such as choice of appropriate medical image modality, a choice of a 2D image reconstruction algorithm and patient's dosage. While computer scientists have little control over the data they receive for post processing, their research can greatly improve the medical data processing and enhance the accuracy of the obtained medical data through a wealth of medical image processing and reconstruction techniques.

Figure 3.7: Semi-automatic segmentation.

Currently, semi-automatic segmentation is most commonly employed – these techniques rely on a limited manual input from the user, thus allowing processing a large number of contiguous images needed for generating a 3D model at a reasonable time and with an accepted degree of accuracy. Figure 3.7 illustrates an example of semi-automatic segmentation that was applied to an image slice from Visible Human Dataset.

3.2.3 Image Processing Techniques

A number of semi-automatic image processing tasks can be performed to augment and enhance the work of medical practitioners. Smoothing allows improving and simplifying an image through reducing noise and keeping the important features. Segmentation is one of the pivotal medical image processing tasks and much effort has been spent by the research community to improve semi-automatic segmentation and to achieve automatic segmentation methods. Segmentation is the process of isolating anatomical structures of interest for quantitative shape analysis, visualization and subsequent model building [47]. Semi-automatic segmentation relies on either image thresholding or edge detection algorithms and involves a certain amount of manual post- and preprocessing.

3.2.3.1 Image Smoothing

During the smoothing, an image is simplified while preserving important information. The main goal of image smoothing is to reduce noise without losing key features. Thus, the image is pre-processed with the aim of simplifying the subsequent analysis [47].

If $I : D \rightarrow C$ is a given image which contains a certain amount of noise, then the most straightforward approach to removing noise is to approximate I by a modified function S [47]. Thus, the image function I is replaced by the convolution $S^\sigma = G_\sigma * I$ where $G_\sigma = \left(2\pi\sigma^2\right)^{n/2} e^{-\|x\|^2/2\sigma^2}$ is a Gaussian kernel.

Gaussian smoothing will smear out the noise artifacts in the image on scales of order σ and smaller. As the side effect of smearing out the noise fluctuations, Gaussian

smoothing blurs edges in the images, which can make extracting boundaries and key features in the image more difficult (Figure 3.8). A number of edge preserving smoothing methods exist [49] that allow preventing or diminishing the smearing of the edges while blurring out the noise.

Figure 3.8: Original CT scan (a) and smoothed version (b).

3.2.3.2 Image Registration

Image registration is an important technique in medical image processing. It is a process of bringing multiple images together in order to determine the spatial correspondence amongst them and to align them spatially.

In the context of medical imaging, image registration is extremely useful in bringing images from different modalities together. Data from different patient positions, or different time frames, can be compared in order to get a better diagnosis or to monitor the progression of the disease. A good example of medical image registration is the combination of pre-surgery and intra-surgery images [49]. Real time images acquired during the surgery are often low resolution images due to time constraints. Image

registration helps relate the pre-op and the intra-surgery images in order to better guide the surgeon throughout the operation.

Registration has been studied and covered extensively in literature. Numerous approaches have been proposed, with underlying techniques ranging from statistics to computational fluid dynamics. [47].

3.2.3.3 Image Segmentation

As medical imaging continues to play a prominent role in diagnosis and treatment of disease [63], scientists involved in medical image processing and analysis have been concentrating their effort on the challenging problem of computer-assisted extraction of clinically useful information about the anatomical structures imaged using modalities such as CT, MR, and PET.

Segmentation is a process of organizing an unstructured visual representation into a structured version of it. Raw digital data has no structural descriptions beyond intensity values and has to be partitioned into homogeneous regions that are semantically meaningful for a particular application. In the context of medical imaging, the partitions have to be anatomically meaningful [47].

Segmenting structures from medical images and subsequent reconstructing of a compact geometric representation from the segmented data remains a difficult task due to the sheer size of the datasets and the inherent complexity and variability of the anatomical shapes of interest [63].

3.2.3.4 Boundary Detection

Detection and description of boundaries in digital images is one of the fundamental tasks in medical image processing. Due to the presence of noise and image artifacts, boundaries cannot be detected accurately using only intensity and contrast information [60].

The shortcomings typical of sampled digital data, such as sampling artifacts, noise and spatial aliasing can result in indistinct and often disconnected boundaries [63]. The challenge in boundary detection is to identify boundary elements as belonging to the same structure and then to extract these elements forming a coherent and consistent model of that boundary structure.

Low level image processing methods, although fast and easy to implement, usually rely on the local information and make incorrect assumptions during the local neighborhood interrogation process, resulting in inaccurate and infeasible object boundaries. Subsequent cleaning and interpretation of these extracted boundaries, as the result, will require considerable amounts of expert intervention, which is costly and time consuming. Edge detectors can fail in the vicinity of the boundary where boundary transition is not well defined or noise is present. In the presence of noise along the boundaries, an edge detector can find multiple false positive edges where only one boundary is expected, as seen in Figure 3.9.

Figure 3.9: Edges produced by Canny edge detector for a high resolution cross sectional image from Visible Human Dataset.

Low level image processing methods that rely on local edge information and intensity cannot be used directly for object boundary detection since they cannot automatically detect boundaries and will not produce a connected and topologically correct boundary structure [48]. Yes, these techniques are often instrumental as the first step towards semi-automatic edge detection and can be used as a guidance step for manual segmentation and approaches such as active contours.

Segmentation of anatomical structures is the crucial task of medical image analysis and the essential step of most medical processing tasks such as registration, tracking and labeling [60]. Medical image analysis applications require that anatomical raw data from the original image be reduced to compact analytical representations of the shapes [60]. For example, segmentation of the heart image data is crucial for extracting

such vital diagnostic information as ejection-fraction ratio, heart output, and wall thickening.

Currently, clinical segmentation relies heavily on manual slice editing. A skilled technician manually traces the region of interest in each slice – a labor-intensive and time-consuming process. Accurate and consistent results are difficult to achieve due to reasons such as operator bias and fatigue.

Figure 3.10: Original CT scan (a) processed by thresholding: $T=1200$ (b).

Segmentation involving traditional low-level image processing techniques such as region growing, edge detection and thresholding (Figure 3.10) require considerable amounts of expert guidance and user input. Yet, automating these models remains a difficult task due to the complexity of shapes present in medical image data and the variability within the sets of patient data [60].

3.2.3.4.1 Active Contours – Snakes

Active contours, or ‘snakes’, are frequently employed to detect boundaries in medical image datasets [53-60]. Since the user input required for active contours is minimal, snakes are considered to be the first step towards automated boundary detection.

Snakes combine the bottom-up constraints derived from image data together with top-down, *a priori* knowledge derived from the image such as location, shape and size of the structures [63].

Figure 3.11 illustrates the initial and the final steps of the boundary detection process with snakes.

Figure 3.11: Active contours applied to MRI data—an iterative procedure.

Snakes or active contours method is currently the most widely used approach for segmentation and boundary extraction in medical images [61, 63]. Energy minimizing active contour models were introduced by Kass et al [11], and since then have been actively employed in medical image analysis. The snakes method has gained popularity due to the ability to deform dynamically in order to adapt to the shape of interest in the image.

First introduced by Witkin, Kass and Terzopolous [61], the active contour approach for automatic boundary detection has since been investigated by a number of researchers [53-65]. Given an image $I : D \in \mathbb{R}^2 \rightarrow C$, the initial parametrized curve

$\Gamma:[0,1] \rightarrow D$ is conditional on a steepest descent flow for an energy functional of the form [47]:

$$E(\Gamma) = \int_0^1 \left\{ \frac{1}{2} w_1(p) |\Gamma_{pp}|^2 + \frac{1}{2} w_2(p) |\Gamma_p|^2 + W(\Gamma(p)) \right\} dp$$

The first two terms control the smoothness of the active contour Γ . The active contour explores the image information through the potential function $W:D \rightarrow \Re$ which is usually constructed to be large near the edges and small elsewhere. One example of the W function could be [48]:

$$W(x) = \frac{1}{1 + \|\nabla G_\sigma * I(x)\|^2}$$

Thus, minimizing the overall energy E will result in attracting the active contour Γ to the edges in the image.

Active contours use not only the local gradient information but also employ the long-range spatial distribution of the gradient [60]. The long-range spatial distribution is incorporated by combining continuity and curvature constraints with local gradient strength [60].

3.2.3.4.2 Active Contours in Medical Image Analysis

Originally introduced in the field of computer vision and computer graphics, active contours since then have been used extensively in medical image analysis. Their ability to perform semi-automatic segmentation, tracking and visualization of anatomical structures has been explored by the medical image research community.

The main advantage of snakes over traditional image segmentation methods is that both spatial and image information is explored through energy minimization. Thus,

these models lock onto edges and other features of interest and can provide a unified solution to image processing tasks such as boundary detection, tracking and extraction. While this approach yields a smooth closed boundary contour, it requires an initial boundary input from the user. Because of numerically unstable computations, complexity, initialization and inability to capture fine details (Figure 3.12), this widely accepted approach is deemed unsuitable for automatic, highly accurate border extraction.

Although snakes offer a significant improvement over low-level image processing techniques towards robust boundary detection, they are not without limitations. First and foremost, snakes are interactive models and in order to use them for automatic and semi-automatic segmentation, they must be initialized as close to the boundary of interest as possible in order to accurately detect the boundary. A choice of appropriate forms of internal and external energies remains a central problem for snake implementation. For example, internal energy constraints can limit geometrical flexibility of the active contour and prevent the snakes from representing more complicated shapes [60].

Figure 3.12: Snakes models fail to capture fine details. An MRI scan of the lung (a), boundary detected by the snake (b), and close-up view showing loss of details on sections of the contour (c).

While the original snake definition provides an elegant method to simulate an elastic material which can dynamically attract towards image features of interest [58], the smoothness constraint has to be explicitly defined. In the literature, the following problems associated with the original definition have been recognized [58]:

- Slow convergence speed due to the optimization of the large number of constraint coefficients
- Difficulty of automatically determining weights associated with smoothness constraints
- Representation of the curve with a finite set of disconnected points
- Low accuracy and error associated with calculating high-order derivatives on the discrete curve in noisy environments

A wide range of approaches have been proposed to overcome the shortcomings of snakes and to further automate the deformable contour segmentation process. For a comprehensive survey on deformable models in medical image analysis, refer to [63].

3.2.4 Computational Anatomical Models

An increasingly important role of medical imaging in the diagnosis and treatment of disease has opened an array of challenging problems all centered on the computation of accurate geometric models of anatomic structures from medical images.

The concept of generating 3D models from a stack of CT images was first published in 1980 [68] and the idea of reproducing human anatomy with a computational model has gained a lot of research interest.

A computational model depicts mathematically an organ or a tissue of the body, with the aim to reproduce the behavior of the various components of the biological system under controlled conditions [7]. In order for the computational model to be useful, such important characteristics of the biological components as chemical composition, shape, size, relative location and possible movement, need to be incorporated.

Simulations have become an importable and indispensable tool for clinical studies and experimental methods in medical research. Simulations involving computational models (phantoms) are extremely useful when experimental studies are inadequate, clinical studies are too lengthy and expensive and potentially dangerous to human subjects and theoretical solutions are sought [66]. Building an accurate phantom based on exact human anatomy and physiological functions provides an indispensable research tool for researchers in the field of medical imaging devices, medical image processing, and computer aided tissue engineering, among many.

Currently, existing medical phantoms are divided into two general classes: pixel-based and geometry-based [66]. Pixel-based models are generally built based on patient data and are thus fixed to a particular anatomy and resolution. Introducing anatomical variations in order to extend the application of these models to a broader range of patients and conditions is difficult and often impossible. Since orange shapes are not analytically defined, calculations based on these models can only be approximated.

Geometry-based computational models, on the other hand, are based on geometric primitives. As a result, they allow anatomical variation and are resolution independent. Since shapes are analytically defined, important properties of the anatomy such as organ volume and circumference, can be accurately described and determined.

3.2.4.1 Voxel Based Models

Rapid advancements in medical imaging technology and the ever increasing use of medical imaging in medical practice have provided scientists with high-resolution cross-sectional digital images of human anatomy [50]. These digital images are essentially grids of pixels, typically represented by 512x512 pixels. When such pixel data is extended into the third dimension, pixels become cuboidal volume elements (Figure 3.13 and Figure 3.14) referred to as voxels [50]. Voxels are usually defined to contain a uniform medium, an associated coordinate address and an index that identified the voxel as belonging to a particular organ or tissue.

Figure 3.13: Voxel representation of a cube in object space.

The construction of voxel models requires the segmentation of medical data. The boundaries between various organs and tissues are identified and voxel belonging to the same structure are grouped together for visualization. Segmentation remains a time consuming and challenging problem as completely automatic methods for medical image segmentation have not been reported in literature [50].

Interpolation converts the sparsely spaced 2D slices formed by an imaging modality such as CT, MRI or ultrasound into a continuous 3D scene. Because medical image modalities leave un-imaged space between adjacent slices of patient data,

interpolation is used to fill in the space between the slices. Inter-slice density values are estimated by computing a linear average of opposing voxels in the original slices.

Figure 3.14: Voxel representation of a CT slice.

Voxel models are currently considered to be the most faithful representation of the human anatomy. They are built based on data from the actual anatomy, acquired either from live patients or cadavers. Easy to implement and construct once medical data is segmented, these models are very realistic and thus prove to be a great visualization tool and are well suited for rapid prototyping (Figure 3.15).

Figure 3.15: Voxel model built from Visible Human Dataset.

Voxel models, however, are not without shortcomings, such as aliasing and huge storage requirements. The construction of voxel models requires rather time consuming medical image segmentation, and structures smaller than a pixel cannot be adequately modeled.

The objective behind developing a computational model is to not only visualize the internal structure of the body, but also to mimic as closely as possible the biological properties of the region of interest in the body [14]. Voxel models lack geometric representation which makes them unsuitable for bio-engineering analysis and simulation.

Chapter 4 - Heterogeneous Modeling Using B-Spline

4.1 Contour Fitting with B-Spline Curves

In the first phase of the research, we tackle the following challenging problem. Given a cross sectional image of the human body, generate a B-Spline curve of the organ or region of interest that accurately approximates the boundary of that region of interest. This curve is an intermediate step used for turning the set of contiguous cross-sectional medical images into a lofted B-Spline model (Figure 4.1).

Figure 4.1: Extracted and fitted contours from sequential cross-sectional CT images stacked (a) and lofted (b) using Rhino lofting capability.

The problem appears rather easy to solve, given a large number of image processing and curve fitting techniques already available to the research community. However, automatic interpretation and segmentation of medical images remains a difficult and mostly unsolved problem [63]. Semiautomatic methodology is likely to

remain dominant in research and in medical practice [63], but even then, the preprocessed and segmented CT and MRI pixel data is jaggy and contains a lot of noise and dangling pixels (Figure 4.2). The shapes these pixel data represent contain a lot of intricate detail not present in traditional CAD and manufacturing data. Thus, the proposed method requires a number of different techniques both from the medical image processing as well as CAD. Proper scan data preprocessing is a vital step since input data cannot be overly noisy or be incorrectly ordered in order for the approximation step to produce a satisfactory B-Spline contour. In the sections below, we give the necessary details and discuss the steps involved in fitting pixel contours from cross sectional medical data with smooth B-Spline curves.

Figure 4.2: CT image showing great contrast between the bone and the surrounding tissue (a). Zooming in (b) exposes the jaggy pixel data and the small intricate details present throughout the contours of the regions of interest.

4.1.1 Medical Data Preprocessing

First step towards successfully reconstructing a 3D CAD model from a medical imaging dataset is to accurately extract features of interest from the digital medical images. Segmenting medical datasets remains a non-trivial task due to the size of the dataset, the complexity of anatomical shapes, sampling artifacts, and noise, which results in disconnected and inaccurate representations of the organ contours [14].

For this project, we used 3 distinct medical datasets: CT scans of the human fibula and tibia (Figure 4.3), MRI of the human brain and abdominal section, and Visible Human Project dataset [113].

Figure 4.3: A CT image of fibula and tibia from the CT dataset.

The CT dataset of the lower leg contains 753 images taken at 1.3 mm distance, each of the size of 512x512 pixels. The dataset is stored in DICOM format. Since CT

imaging offers excellent contrast for the skeleton without the use of contrast agent, this dataset proved extremely useful for contour generation and fitting part of the project.

MRI offers great soft tissue contrast which made MRI dataset suitable for heterogeneous point cloud generation and tissue density modeling (Figure 4.4).

Figure 4.4: MRI of the pancreas-excellent soft tissue contrast.

Visible Human Dataset [113] offers an unparalleled view of anatomy through the use of cross-sectional color photography in addition to CT and MRI imaging of the cadaver (Figure 4.5). Visible Human Dataset color images of the male cadaver have a very small voxel size – $0.33\text{mm} \times 0.33\text{mm} \times 1\text{mm}$ – and thus are capable of capturing and preserving unique anatomical details. Each slice of the original cross-sectional color photography is a 2048×1216 pixel 24-bit color image that is about 7.5 MB uncompressed, with the entire data set being about 14GB for the male cadaver.

Figure 4.5: Cross sectional photograph from male cadaver dataset.

Despite extensive research in the field of medical image segmentation, fully automatic segmentation of medical data remains an unsolved problem [6]. Due to inherent noise in the data, complexity of anatomical shapes, overlap of gray level values between neighboring tissues and organs and lack of distinct boundaries, segmentation is often performed manually. Manual segmentation is time consuming and requires a trained experienced radiologist to correctly identify and outline each tissue. Even with manual segmentation, the results of the segmentation can be rather inconsistent and vary greatly even among experienced radiologists. Discrepancies as large as 24% for difficult to segment organs such as esophagus have been reported [6].

Thus, semi-automatic segmentation remains the current state of the art – these techniques require a limited manual input from the user in order to perform segmentation. Since the input from the user is rather minimal, these techniques allow processing a large number of contiguous images needed for generating a 3D model at a reasonable time and with an accepted degree of accuracy.

Semi-automatic segmentation methods use either image thresholding or edge detection algorithms and require a certain amount of manual post- and preprocessing. A widely accepted approach by medical and science community to 3D model construction is segment medical data using a commercial software package (eg. MIMICS), which involves manually thresholding and editing each slice individually, as the result of automatic edge detection often provides a poor result (Figure 4.6).

Figure 4.6: Using commercial software for semi-automatic segmentation will require a significant amount of post-processing.

4.1.1.1 Boundary Detection

One of the central problems of medical image segmentation is that of determining of the boundaries of the segmented homogeneous regions that are anatomically meaningful. A typical example is identifying a tumor in an MRI or CT image and extracting the boundary of the segmented region in order to visualize and quantize the tumor site.

	147	148	149	150	151	152	153	154	155	156	157
206	0	0	0	0	1	0	0	0	5	0	0
207	0	0	0	5	1	0	0	0	5	0	0
208	0	0	0	1	0	0	0	5	0	0	0
209	11	1	16	9	1	0	0	8	0	0	0
210	8	3	12	8	1	5	16	24	0	0	48
211	6	0	0	0	1	4	16	14	0	0	198
212	12	0	0	0	1	7	7	0	0	111	459
213	0	0	0	0	5	10	0	0	45	317	736
214	0	0	0	0	6	0	0	0	189	577	947
215	0	0	2	24	23	1	0	77	412	812	1033
216	0	7	14	24	19	1	6	223	644	964	1041
217	0	6	3	6	0	1	74	427	847	1032	1024
218	0	8	8	1	0	13	229	655	1000	1075	1035
219	0	0	0	0	0	73	398	828	1062	1080	1033
220	14	6	4	0	10	218	627	976	1087	1074	1037
221	18	16	13	0	74	402	836	1066	1084	1062	1046
222	17	12	0	0	173	605	980	1084	1041	1018	1022
223	0	7	0	54	349	801	1063	1077	1003	982	997
224	0	0	0	151	553	945	1079	1054	984	957	969
225	6	0	31	302	752	1033	1062	1025	977	973	987
226	18	5	117	501	916	1064	1027	994	973	985	992
227	3	22	246	706	1033	1084	1030	1000	1007	1011	1002
228	0	60	389	848	1062	1058	1029	1027	1033	1024	997
229	0	156	566	958	1061	1027	1010	1029	1042	1045	1018
230	38	289	733	1036	1072	1029	1011	1032	1045	1054	1037
231	08	448	876	1068	1064	1012	985	1023	1049	1056	1029
232	92	602	983	1072	1027	980	979	1017	1043	1046	1018
233	14	762	1051	1065	977	942	977	1009	1031	1014	998
234	47	892	1099	1060	964	946	1005	1047	1048	1003	982
235	12	1016	1106	1021	931	937	1022	1058	1034	997	989
236	99	1090	1095	1003	930	954	1031	1057	1023	996	999
237	50	1100	1051	974	938	968	1008	1028	1019	1015	1019
238	28	1076	1016	973	960	964	962	974	995	1000	1010
239	68	1044	989	987	987	981	961	975	1000	1001	1015

Figure 4.7: DICOM image and corresponding intensity values. The boundary is not well defined.

Detecting and extracting a boundary from a medical image is a nontrivial problem to be done automatically. While there are discrepancies reported among trained

radiologists [6], the pixel intensities themselves are rather misleading – when looking at the intensity values, one can notice that the contour is inconsistent and not clearly defined, as seen in Figure 4.7.

Thresholding remains the simplest approach to segmentation and edge extraction. It is particularly useful for images containing solid objects against contrasting background [71], and thus performs well on CT data for bone segmentation (Figure 4.8) For our research, we originally used thresholding for CT datasets when extracting the contours of the bones, tibia and fibula.

Figure 4.8: Global thresholding focuses on one region of interest.

Since a single global threshold value often cannot describe the entire region of interest completely (Figure 4.9), various thresholding techniques such as hysteresis thresholding or adaptive thresholding were used.

Figure 4.9: Applying global threshold to the original CT image (a) yields different results. Global threshold set to 1800 (b) and 1900 (c).

Once the image has been thresholded, the boundary of the segmented region can be easily extracted from the binary image (Figure 4.10).

Figure 4.10: Region of interest segmented using thresholding (a). Boundaries extracted from the binary image (b)

An alternative approach to establishing the boundaries of the objects of interest in a medical image is to first identify the pixels belonging to the boundary based on chosen criteria, and then link these points to form an ordered connected boundary. The points labeled as belonging to the boundary are edge points, or edges. Edges are basic features

which carry useful information about object boundaries in an image. Due to the complexity of the image content, a widely accepted and precise mathematical definition of an edge has not been reported to date [70].

4.1.1.2 Edge-Based Segmentation

Edge-based segmentation [104-107] uses an edge detecting operator to segment regions of interest in the image. The result of the segmentation is an edge map, in which edges manifest locations of discontinuities in intensities, texture, etc.

Given an ideal image with a bright object O against a contrasting background, the physical object is represented by its projection on the image I . Since object is contrasted on the background, the variations of the intensity I are large on the boundary dO . Hence, it is rather intuitive to characterize the boundary dO as the locus of points where the norm of the gradient $|\nabla I|$ is large [48]. The approach was introduced in the 60's and the 70's by Roberts [72] and Sobel [73]. Canny improved on the original edge detection methods by adding a smoothing pre-processing step in order to reduce the influence of the noise on edges and a thinning post-processing step.

Edge points form an edge map of the original image. It is customary a binary image with edge points assigned intensity 1. The edge points indicate the position and overall shape of the boundaries but seldom form connected and closed contours. Thus, before the boundary can be extracted and processed, edge linking step is performed next.

Local edge detectors cannot be used directly for the detection of object boundaries since local edge points need to be arranged in topologically connected correct boundaries [47]. But edge detecting techniques are customary used as the first step towards

identifying the edge pixels and guiding semi-automatic boundary detection methods such as active contours.

4.1.1.3 Canny Edge Detector Implementation

In this research we used Canny edge detector [21, 103] as the first step towards identifying, extracting and fitting the contour of the anatomical region of interest (Figure 4.11).

Figure 4.11: Applying Canny algorithm to MRI image of the lung (a) produces a binary edge map (b).

The “classic Canny Edge Detector” algorithm runs in 5 separate consecutive steps:

- Smoothing
- Calculating the gradients
- Non-maximum suppression
- Hysteresis thresholding

4.1.1.3.1 Smoothing

It is inevitable that images contain a certain amount of noise. Since some of the noise might be mistaken for edges, in the first step of the algorithm, the noise is reduced through smoothing. We implemented Gaussian smoothing by applying a Gaussian filter. A convolution mask is usually chosen to be much smaller than the actual image, and the larger the window size of the convolution mask, the lower is the sensitivity of the edge detector to noise. We chose to use the following kernel of a Gaussian filter with a standard deviation $\sigma = 1.4$:

$$B = \frac{1}{159} \times \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 2 \\ 5 & 12 & 15 & 12 & 4 \\ 4 & 9 & 12 & 9 & 2 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

4.1.1.3.2 Calculating Gradient

Canny algorithm identifies pixels as edge points where the grayscale intensity of the image changes the most, by calculating the gradients in the image. We determine gradients at each pixel in the smoothed image by applying the Sobel operator kernels in the x - and the y -direction as follows:

$$K_{GX} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad K_{GY} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

The gradient magnitudes, also known as edge strengths, are determined as the Euclidean distance measure:

$$|G| = \sqrt{G_x^2 + G_y^2}$$

where G_x and G_y are the gradients in the x - and y -directions, respectively. Along with the magnitude of the gradient, the directions of the edges using the formula:

$$\theta = \arctan \left(\frac{|G_y|}{|G_x|} \right)$$

4.1.1.3.3 Non-Maximum Suppression

The ‘non-maximum suppression’ step keeps only the pixels on the edge with the highest gradient magnitude. The idea is that maximal magnitudes should occur right at the edge boundary. The following rules were followed for examining neighboring pixels in the 3x3 window of the pixel (x, y) :

- If $\theta'(x, y) = 0^\circ$, then the pixels $(x+1, y)$, (x, y) and $(x-1, y)$ are examined
- If $\theta'(x, y) = 45^\circ$, then the pixels $(x+1, y+1)$, (x, y) and $(x-1, y-1)$ are examined
- If $\theta'(x, y) = 90^\circ$, then the pixels $(x, y+1)$, (x, y) and $(x, y-1)$ are examined
- If $\theta'(x, y) = 135^\circ$, then the pixels $(x+1, y-1)$, (x, y) and $(x-1, y+1)$ are examined

Then, if the pixel (x, y) has the highest gradient magnitude from the cluster of the pixels examined, the pixel is kept as an edge pixel. Otherwise, the pixel not on the edge and should be classified as such. This step is used to produce one pixel wide edges.

4.1.1.3.4 Hysteresis Thresholding

Since an edge detector will produce edges that will be weak or unwanted edges, a filtering step is performed to get rid of the noise. Using one global threshold for eliminating pixels can remove valid pixels on a connected edge, introducing discontinuities. Instead, hysteresis thresholding is used: pixels with a gradient magnitude $D \leq t_{low}$ are discarded as noise, and pixels with $t_{low} \leq D < t_{high}$ are kept if they form a continuous edge chain with the pixels with high gradient magnitude, $D > t_{high}$.

Figure 4.12: Adjusting thresholds in hysteresis thresholding step results in fewer edges. The original CT scan (c) followed by the results of hysteresis thresholding (b,c).

4.1.1.4 Edge Following

In order to fit a B-Spline curve to the contour data, pixels along the boundary need to be extracted and ordered properly. Ordering of pixels is essential for a successful B-Spline fit.

Since we are working with medical data, there is a certain amount of prior knowledge that we can incorporate in the design of the edge following and linking algorithm:

- Pixel intensity corresponds to attributes of the subject, thus it is possible to calculate textures/information for segmentation
- Large amount of noise is present due to sensitivity of the measurement, artifacts caused by small movements
- Shape information such as closed contours and size since basic shape is known from anatomy

Following the detection of edges in the image, using the Canny edge detector, we employ an edge following algorithm to connect, order and extract arrays of boundary pixels. Our method is based on depth-first-search (DFS) algorithm. DFS is often used in image traversing, and has been applied to segmentation and path finding in medical images [108,109]. While DFS produces a spanning tree of all the vertices reached during a graph search, in our application the goal is to generate an orderly sequence of a closed boundary pixel positions. Just as in DFS, three categories are used to describe the vertices: undiscovered, discovered, and visited.

An edge map of the original image produced by an edge detector is used as input to the contour tracing algorithm. The algorithm creates a table of the same size as the input image and sets all cells to -1, to describe all pixels in the corresponding input image as *undiscovered*. Once an edge pixel is located (white pixel in a binary image, where 0/black denotes background and 255/white denotes an edge), it is marked as *discovered* and *current* and the corresponding table cell is changed to 0. The algorithm then proceeds to discover the neighbors of the *discovered* pixel, by recursively applying the algorithm. Eight neighbors are examined in a clockwise fashion: if only one neighbor is discovered, its status is changed from *undiscovered* to *discovered*. Once all neighbors are traversed,

current pixel x and y coordinates are added to the array of ordered boundary pixels. If two or more neighbors are discovered, the algorithm further investigates each member to identify dangling pixels and chooses one of the neighbors to proceed.

Although the Canny edge detection algorithm produces ridges that are one pixel wide due to the non-maximal suppression step, discontinuities are still possible and can negatively affect the ordering of the pixels needed for the fitting [9,32]. Therefore, while the Canny edge detection algorithm finds edges automatically, minimal post-processing might be required in order to get rid of remaining few dangling pixels (Figure 4.13).

Figure 4.13: Dangling pixels will disrupt the ordering of the pixels when using DFS approach.

4.1.1.5 Contour Cleaning

After the contour has been traced and cleaned we are ready to fit a NURBS curve to the pixel data. While it seems a matter of selecting one of the many fitting capabilities from existing libraries, several fitting attempts that have failed to produce acceptable results revealed that traditional CAD techniques are not appropriate to deal with image data, at least not without significant modifications.

Using existing design engineering approaches, one would fit a curve to a large dataset using one of the following approaches [76]:

- Start with an interpolation, i.e. the curve passes through all the data points, and then remove as many knots (control points) as allowed by the tolerance. This method did not work because the interpolating curve had too many wiggles that were not removable even with a high tolerance.
- Start with an approximation with some number of control points and increase the degree of freedom (knots and control points) until the tolerance requirement is met. This method did not work either because:
 - It was nearly impossible to guess the number of start degrees of freedom to get a reasonably good initial approximation
 - Intricate details could not be reproduced even if the number of control points were increased drastically
 - As the number of control points was increased, the curve approached an interpolating status and the unwanted wiggles started to appear

Figure 4.14: Canny edge map (a) and a contour approximated in the least square sense (b).

Even simple shapes with few intricate details as seen in Figure 4.14 present a challenge for global fitting methods.

In order to capture all the important detail, some methods attempt to interpolate the pixel data.

Figure 4.15: Original CT scan (a) followed by the edge map (b). Extracted contour pixels displayed in (c, d).

In Figure 4.15, Canny edge detector is applied to a CT scan (Figure 4.15(a)). The innermost contour is extracted and displayed in Figure 4.15(d).

Figure 4.16(a) shows pixel data from the inner contour that was extracted and ordered for contour fitting. When applying interpolation, the fitted B-Spline curve passes through every point and thus produces a curve that is not smooth and does not capture the shape of the internal anatomic structure faithfully, as seen in Figure 4.16(b).

Figure 4.16: The points for inner contour form a jaggy data set (a) and the interpolating curve passes through every point (b).

Interpolating through contours already containing a high number of control points will lead to exponential increase of overall control point count and subsequent numerical problems.

For simple contours devoid of intricate detail, existing automatic contour detection and fitting methods perform rather well. However, in presence of small yet crucial details which are abundant in medical datasets, these methods do not represent the desired shapes adequately, which can lead to incorrect overall presentation of anatomy as shown in Figure 4.17 and Figure 4.18. Faithful capture of the overall shape is of uttermost importance in medical applications such as custom implant design and calculation of volume and surface area of organs for medical dosimetry [6].

Figure 4.17: B-Spline naïve Least-Squares approximation. Pixel points are black and the fitted B-Spline curve is red. Selecting fewer control points (a): 3% of data points,(b): 5% of data points results in more profound loss of detail.

Figure 4.18: Original dataset (a) approximated with a least-squares B-Spline curve (b).

Having exhausted the existing CAD-based techniques in order to solve the problem, we approached the problem from a different angle: smoothing out the jagged pixel data and decomposing it into segments of similar complexity and shape. While the literature offers a few data smoothing methods [77, 78], they were not adequate to be applied in a CAD-based application.

We need a smoothing method that takes pixel data, i.e. the jaggy point set, and turns this data into a smooth point set. That is, the method must be able to alter the data set so that the interpolating curve becomes as smooth as the user wants it to be. Our method is based on a variation of Laplacian smoothing applied to a polygon. Given an ordered point set $Q_0, \dots, Q_m$, and the level of required smoothing lev , the algorithm below generates a new, smoother, set of points.

Algorithm 1: *Smooth_Pixel_Data* (Data points $Q[0], \dots, Q[m]$, degree of required smoothing lev)

```

for  $k = 1$  to  $lev$ 
  for  $i = 1$  to  $m = 1$ 
     $Q_i \leftarrow \frac{1}{4}Q_{i-1} + \frac{1}{2}Q_i + \frac{1}{4}Q_{i+1}$ 
  endfor
  if ( closed )
     $Q_0 \leftarrow \frac{1}{4}Q_{m-1} + \frac{1}{2}Q_m + \frac{1}{4}Q_1$ 
     $Q_m \leftarrow Q_0$ 
  end if
end for
return smoothed data set  $Q[0], \dots, Q[m]$ 

```

That is, the algorithm recursively computes a new position for each point until the required level is reached. Several examples are shown in Figure 4.19 and Figure 4.20 where the curve shown is the interpolating curve to the smoothed data points. It is evident that even one level of smoothing improves the data considerably. It is also interesting to note that as the level of smoothing increases, the new data set converges to a point (for closed data sets).

Figure 4.19: Level of smoothing zero (a) and one (b).

Figure 4.20: Level of smoothing five (a).
Convergence with levels 1, 5, 20, 100 and 500 (b).

An important question remains to be answered: when to stop smoothing, i.e. when is the data set smooth enough? There are two types of answers to this question: a technical and a practical one. The technical answer goes like this. For each smoothed point take a small neighborhood, e.g. 3-5 points on either side. Fit a curve, e.g. a circle, to this small data set and check the error. If the error is within a certain deviation, stop smoothing. While this is technically correct, it has two disadvantages: (1) it is

computationally expensive, and (2) it opens up new questions as to what are the right neighborhood and the acceptable deviation.

We opted to go with the practical approach: use a fixed level of smoothing based on the type of the data set. Medical data is fairly predictable within the organ type it represents so it is fairly easy to set up a knowledge base that gives the user the proper level of smoothing for successful curve fitting. In our examples all data was smoothed by 3-5 levels of smoothing.

4.1.1.6 Point Data Segmentation for Curve Fitting

The smoothed and ordered dataset $Q_0, \dots, Q_m$ obtained in the previous step is now used to assist in finding an optimal B-Spline curve that approximates the data set to within a user supplied tolerance and has relatively few number of control points. To approximate the given data set in the least-square sense requires the computation of parameters $t_0, \dots, t_m$ where the points are assumed, the choice of degree p , the highest index of control points n , and the knot vector U . The parameters are computed using the chord-length method [76], the degree will be chosen to be two or three, and the highest index and the knot vector will be computed in the section that follows.

Each B-spline curve can be thought of as the collection of Bezier segments that are automatically joined with C^{p-1} continuity, assuming that there are no multiple interior knots. This B-spline-to-Bezier relationship is what our method takes advantage of to determine the appropriate degrees of freedom (number of control points) and the right knot locations [110]. That is, the smoothed data set will be segmented using a set of Bezier curves so that on each segment the error is less than the given tolerance.

A Bezier curve is used frequently in computer graphics and CAD. Due to the blending used for Bezier curves and the lack of local control associated with it, this parametric curve is inadequate for fitting large datasets with intricate details. Increasing the degree of a Bezier curve could add flexibility to the shape capturing capabilities, but it would also increase significantly the processing effort for evaluation and introduce numerical noise in the computation. Instead, we can utilize numerically stable Bezier arcs to subdivide the dataset into smaller shapes and use this knowledge for B-Spline fitting.

Hence, least-squares Bezier fitting is applied to obtain a set of C^0 Bezier curves that approximate the data to within a tolerance. The Bezier least-squares fitting is a special case of B-spline fitting and the system to be solved simplifies to:

$$(B^T B)P = R$$

$$B = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ B_{0,p}(t_1) & B_{1,p}(t_1) & \cdots & B_{p,p}(t_1) \\ \vdots & \vdots & \cdots & \vdots \\ B_{0,p}(t_{m-1}) & B_{1,p}(t_{m-1}) & \cdots & B_{p,p}(t_{m-1}) \\ 0 & 0 & \cdots & 1 \end{bmatrix} \quad R = \begin{bmatrix} Q_0 \\ B_{1,p}(t_0)Q_0 + B_{1,p}(t_1)Q_1 + \cdots + B_{1,p}(t_m)Q_m \\ \vdots \\ B_{p-1,p}(t_0)Q_0 + B_{p-1,p}(t_1)Q_1 + \cdots + B_{p-1,p}(t_m)Q_m \\ Q_m \end{bmatrix}$$

where the parameters $t_0, \dots, t_m$ are computed using the chord-length method [76]. Now, given the Bezier fitting technique with a given degree, the point set is segmented using a combination of Bezier fitting and binary search type index splitting.

Algorithm 2: *Bezier_Arc_Splitting* (Data points $Q[0], \dots, Q[m]$, parameters $t[0], \dots, t[m]$, approximation tolerance eps)

```

start = 0 ; end = m
while end > start do
  left = start; right = end
  do
    n = end-start
    fit a Bezier curve to the points  $Q[start], \dots, Q[m]$ 
    success  $\leftarrow$  check if parametric error <  $eps$ 
    if success = yes then right = end else left = end
    end = (left + right)/2
  while left  $\neq$  end
  start = end; end = m
  save the parameter/index of the end point for the output parameter list
end while
return index set  $j_1, \dots, j_k$ 

```

That is, the algorithm accepts $Q[0], \dots, Q[m]$ and $t_0, \dots, t_m$ as input, uses the longest Bezier arc to locally approximate a subset of these points, and returns the index set $j_1, \dots, j_k$ so that $Q_{j_1}, \dots, Q_{j_k}$ are the junction points of the piecewise Bezier approximation. Several examples are shown in Figure 4.21 and Figure 4.22.

Local Bezier curves capture the shape characteristics of each subset so that each subset is simple enough to approximate the points by a single Bezier arc. We can utilize this information to determine the degrees of freedom required to pass a B-Spline curve to the data, as well as the locations of the knots that will allow faithfully reproducing all the local characteristics of the data.

Figure 4.21: Piecewise C^0 Bezier approximation. Approximation tolerance $\text{tol}=0.0008$ (a), $\text{tol}=0.002$ (b).

Figure 4.22: Segmentation examples. Level of smoothing $\text{lev}=5$, degree=3, tolerance=0.002

4.1.2 B-Spline Curve Fitting

B-spline curves can be fitted to the smoothed data or to the original data [79-84, 75, 85-87]. In many CAD applications curve interpolation followed by knot removal is an effective technique to approximate a large set of smooth point set (right side of the flow chart in Figure 4.23).

Figure 4.23: The main steps of the B-Spline boundary curve approximation algorithm.

Unfortunately, this technique does not work with noisy data so it is replaced with its smoothed version, which may introduce error. This leaves only one option: approximate the original data given the segmentation result of the smoothed data. That is, given $t_0, \dots, t_m$, the segmentation indexes $j_0, \dots, j_k$ (note that there are k Bezier segments), the highest index of control points will be set as $n = k * p$, where p is the degree of the curve. This is the minimum degree of freedom necessary to approximate the curve to within the given tolerance (as per the Bezier segmentation).

The knot vector is computed in two steps. First, a new set of parameters $s_0, \dots, s_n$ are computed from the original parameters and the junction indexes $j_0, \dots, j_k$ as follows:

for each $[t_{j_i}, t_{j_{i+1}}]$ compute

$$\begin{aligned} s_r &= t_{j_i} \\ s_a &\leftarrow (p-1) \text{ average of } t_j, t_{j_i+1}, \dots, t_{j_{i+1}-1} \\ s_{r+p} &= t_{j_{i+1}} \end{aligned}$$

where the k^{th} average of a set is defined as follow: cluster the set into k subsets and compute the average of each subset. If $k=1$ then this produces the usual average of a set. If $k=2$, the set is divided into two (equal) subsets and their averages are computed resulting in two averages. If $k=3$, one gets 3 averages, etc. The formula above computes exactly $p-1$ averages per segment resulting in $n+1$ new parameters $s_0, \dots, s_n$. These new parameters are then used to compute the knots based on the well-known formula of De Boor [76]:

$$\begin{aligned} u_0 &= \dots = u_p = 0 \\ u_{j+p} &= \frac{1}{p} \sum_{i=j}^{j+p-1} s_i \quad j=1, \dots, n-p \\ u_{n+1} &= \dots = u_{n+p+1} = 1 \end{aligned}$$

Now, we have the highest index n , the parameters $t_0, \dots, t_m$ and the knots $u_0, \dots, u_{n+p+1}$, a B-spline curve approximation is computed as formulated above, i.e. the matrix equation is solved for the missing control points. This approximation capability can be extended in a number of ways:

- Approximation with end derivatives specified [75]
- Approximation to closed data set [75]

- Approximation with floating end conditions, i.e. the end points of the curve do not coincide with the end points of the data set
- Approximation with end point constraint, i.e. the end points of the curve coincide with the end points of the data set Q_0 and Q_m

The next important step after the initial fitting is to check the error. Ideally this is done by projecting all points to the curve and computing the perpendicular distances [87]. Point projection is expensive (requires first and second derivatives) and is error prone and hence has not been implemented. A simpler method is to check the parametric error, i.e.

$$\Delta = \max_i |Q_i - C(t_i)| \quad Q_i \text{ is assumed at } t_i$$

If the error test is passed, the approximation has been completed. Otherwise, at certain areas the curve has not faithfully reproduced the shape and more degrees of freedom are needed to comply with the accuracy requirement. There are two options at this point: (1) introduce additional degrees of freedom locally, i.e. add knots to the segments that were not approximated well enough [84], and (2) do a global decomposition with a smaller tolerance. We elected to implement the second method, i.e. compute a new segmentation with a smaller tolerance. The logic behind this is that the piecewise C^0 Bezier segments capture the shape information quite well; however, going from the C^0 Bezier to the C^{p-1} B-Spline may require more freedom given the continuity of this curve. Also, the piecewise Bezier curve provides a shape dependent decomposition and a distribution of parameters that reflects the intricate details inherent in the data set. Practical experience shows that half to quarter of the given tolerance always produces a smooth B-Spline curve that is within the required accuracy and captures all desired details.

The last step in the fitting process is data purging, i.e. eliminating all unnecessary knots (control points). Details on implementing knot removal can be found in [76].

4.1.3 B-Spline Curve Fitting: Examples

The first example is shown in Figure 4.24. The boundary data is obtained from the bone CT shown in Figure 4.24(a). On the left fitting was obtained after one level of smoothing, whereas the right side shows a fit after five levels of smoothing. Both curves provide perfect coverage of the pixel domain (the left figure shows both the pixels and the curve superimposed). The data set has 638 points, level one smoothing resulted in a B-spline fit with 252 control points (60% data reduction), whereas level five produce only 146 control points (77% data reduction).

Figure 4.24: Curve fitting example 1. Original CT scan (a); edge detection (b); fitting with tolerance=0.001 and level=1. Contour points and fitted curve are shown in (c); level=5, curve only (d).

Figure 4.24 (Continued).

Figure 4.25 illustrates a fitting case with a large and a smaller tolerance used. The data was obtained from the lung MRI (Figure 4.25(a)). Note how the approximation misses the details due to the large tolerance. A tighter tolerance (an order of magnitude smaller) produced a perfect contour curve. The data contained 555 points, the 0.01 tolerance produced a B-Spline fit with 39 control points (92% reduction, although the curve is not acceptable), whereas 0.001 tolerance increased the number of control points to 193 (65% data reduction). Note that in the engineering practice a data set of 555 points requires about 10% or less control points because the data is smooth and contains no intricate details. In the medical practice this kind of data reduction is hardly achievable given the noise of the data and the very complicated shape it represents.

Figure 4.25: Curve fitting example 2. Original MRI image of the lung (a); edge detection (b); fitting with level=1, tolerance=0.01, contour points and fitted curve are shown (c); tolerance=0.001, curve only (d).

Figure 4.26 to Figure 4.30 demonstrate a variety of applications of the method on data sets obtained from the brain, head, kidney and bones. All examples used $lev = 3$ smoothing, tolerances 0.01 or 0.001 and the degree $p = 3$, and m, n denote the highest indexes of data points and control points, respectively.

Figure 4.26: Curve fitting example 3. Original brain MRI image (a); edge detection (b); fitting to a partial contour with level=3 , tolerance=0.001, contour points and fitted curve are shown (c); tolerance=0.001, curve only (d).

(a)

(b)

(c)

(d)

Figure 4.27: Curve fitting example 4. Original MRI scan (a); edge detection (b); fitting with level=3, tolerance=0.001, contour points and curve are shown (c); tolerance=0.001, curve only (d).

(a)

(b)

(c)

(d)

Figure 4.28: Curve fitting example 5. Original MRI scan (a); edge detection (b); fitting with level=3, tolerance =0.001, contour points and curve are shown (c); tolerance=0.001, curve only (d).

Figure 4.29: Curve fitting example 6. Original MRI image of the kneecap (a); edge detection (b); fitting with level = 3, tolerance = 0.01, contour points and curve are shown (c); tolerance = 0.001, points and curve (d); curve only (e).

Figure 4.30: Curve fitting example 7. An MRI scan with a more complicated contour: original image (a); edge detection (b); fitting with level = 3, tolerance = 0.001, contour points and curve are shown (c); tolerance = 0.001, curve only (d).

In presence of high contrast MIMICS fitting capability can perform automatic border extraction, as seen in Figure 4.31(a). The B-Spline that gets fitted automatically by MIMICS, however, falls victim to the jaggy and dense pixel data. Although the contour appears smooth and of desired shape, as shown in Figure 4.31(b), the final fitted curve has 2,172 control point and a kink (discontinuity). The original dataset has 1,319 points. Our proposed method fitted a smooth B-Spline curve with 156 control points (Figure 4.30(d)) to the same dataset. Such high number of control points is not acceptable in Bio-

CAD modeling: anatomical cross-sections or CT axial images are taken, on average, at 1 mm, which results in a large image dataset.

Figure 4.31: Border detected (a) and fitted with a B-Spline (b) automatically using MIMICS.

Although the contour generated by MIMICS appears smooth, when zooming into the section of the curve, it is obvious that the contour interpolates through the dataset, as seen in Figure 4.32(b). This fitting explains the high number of control points that the fitted contour has. Our curve, on the other hand, is smooth which can be seen by zooming in into the same section of the original contour, as seen in Figure 4.32(a).

Figure 4.32: Segment of a fitted contour. Proposed method (a) and MIMICS fitting capability (b).

Table 4.1: Data reduction capabilities

Figure	4.24	4.25	4.26	4.27	4.28	4.29	4.30
m	637	554	534	1159	1318	949	294
tol	10^{-3}	$10^{-2} / 10^{-3}$	10^{-3}	10^{-3}	10^{-3}	10^{-3}	$10^{-2} / 10^{-3}$
lev	1 / 5	1	3	3	3	3	3
n	251 / 145	38 / 192	123	81	177	113	21 / 78
%	60 / 77	92 / 65	76	93	86	88	92 / 73

Table 4.1 summarizes the data reduction (%) capabilities.

Data sets with small discontinuities in data contours can be fitted with a closed B-Spline curve, as illustrated in Figure 4.33.

Figure 4.33: B-Spline curve can be fitted to a dataset with discontinuities. Original boundary dataset(a) and the fitted B-Spline curve (b).

4.2 Heterogeneous Fitting with a B-Spline Surface

An increasingly important role of medical imaging in the diagnosis and treatment of disease has opened an array of challenging problems for research community centered

on generating of accurate geometric models of anatomic structures from medical image data. Such computational models should be able to depict an organ or a tissue mathematically with the aim to reproduce their biological characteristics [7].

Biological materials are naturally heterogeneous entities and yet, currently, little work has been done for constructing CAD representations of heterogeneous material properties of living tissues [1]. Presently, no underlying mathematical models that can be used to represent medical image data have been reported [7].

CT and MRI images create detailed visualizations of internal structures. A CT scan is a pixel map of linear attenuation coefficients of a material. Likewise, MRI generates a pixel map of tissue types based on the rates at which perturbed protons return to their equilibrium states.

Due to the direct relationship between material properties and the grayscale values, tissue densities can be estimated using pixel values from medical images [1]. Tissue density distributions can therefore be modeled based on discrete pixel values by generating a point cloud, as shown in Figure 4.34.

Figure 4.34: Intensity values are captured and preserved through point density.

In the second part of this research, we thus describe the implemented methodology for representing discrete medical image data with a continuous B-Spline surface which will faithfully capture and preserve tissue density variations present in the medical dataset.

The proposed approach will be carried out by the following steps: (1) generate a point cloud which will reproduce tissue density variations of (2) order point cloud data and fit cross sections with B-Spline curves that will encapsulate the heterogeneous distribution of the data and (3) loft a B-Spline surface through the cross sections, preserving the heterogeneity. In this section, these steps are described in detail and illustrated with proper examples.

4.2.1 Generating Point Cloud

During the past few decades, the use of imaging modalities has grown tremendously due to the advances in both computer technology and medical image systems [16]. The digital image quality of cross sectional imaging such as magnetic resonance imaging (MRI) and computed tomography (CT) has improved considerably and offer unparalleled view of the patient's interior.

Medical imaging modalities generate image volumes based on a set of specific physical parameters [16]. For example, a CT scan is a map of linear attenuation coefficients of a material, recorded as a pixel value. XR attenuation units generated during a scan characterize the relative density of the given substance. Proton density and mobility is used to generate an MRI image: depending on the material composition of the tissue, the rates of protons returning to equilibrium, once perturbed, differ. These

differences in return rates allow differentiating between neighboring organs and producing a pixel map of the interior.

Figure 4.35: Trabecular and cortical bone tissue are differentiable in the CT scan. Cortical tissue highlighted manually (right).

Regardless of the type of modality used, each pixel/voxel contains a physical parameter value of the corresponding volume in the patient's interior [16]. The data collected from a CT scanner is measured with a dimensionless unit, a CT number, and grayscale images from the CT data are generated by using linear relations that map CT numbers into grayscale space. Pixel intensities are calculated based on Hounsfield units (HU) where HU cover enough range for various tissues enough to produce separation and contrast between tissues and organs.

For example, higher CT numbers and thus, higher grayscale pixel values, indicate a higher material density. CT number of a material is closely related to the linear attenuation coefficient of the material and as such, characterizes the relative density of a substance. Medical image data can be used not only as a means to differentiate between tissues (Figure 4.35), but also to highlight the density variation within various tissues. For

bone tissue, CT numbers measure material density within the bone [1]. Similarly, density variations from medical image data can be used for tumor density analysis in order to gain better insight into the tumor progression and proliferation, and offer new therapeutic approaches.

Suryanto et al [3] reported on a lung cancer study linking tumor density measured from a CT scan with its histological type. The CT data for the lung cancer patient was correlated with the transthoracic fine needle aspiration. The study demonstrated that when using contrast CT, the mean tumor density increased in each histopathologic type of the tumor, indicating that different types of cancer exhibited different vascularization and were thus differentiable using CT imaging.

Kinoshita et al [4] investigated a correlation between cell density and diffusion tensor imaging (DTI) data for brain tumors, by comparing the preoperative MR data and postoperative biopsy results. The results of the study showed a strong correlation between the key values of the DTI imaging, fractional anisotropy and mean diffusivity, calculated directly from MR data, and the histological grading. Since the tumor is heterogeneous throughout [4], a density map and a model based on it can be used as a tool to aid the biopsy by targeting the location of the tumor with the highest cell density, thus providing a more accurate grading of the tumor.

CT and MRI image data can be viewed as a discrete map of materials, in arrays of pixels. Since pixel intensity value corresponds to the relative material density of the tissue, point cloud can be constructed based on the medical data, to capture the density variation in the modeled data. The idea is loosely based on the concept of halftoning – a technique for reproducing continuous grayscale and color images through patterns of dots.

Halftoning simulates continuous tone imagery through the use of equally spaced dots of varying size (Figure 4.36). A continuous tone image contains an infinite range of colors and the halftone process reduces visual reproductions to a binary image that is printed with only one color of ink.

Figure 4.36: The human eye will see halftone dots (a) as continuous tone imagery (b) from sufficient distance

The concept of halftoning can be extended to our application of capturing and reproducing density variations within a medical image slice. Thus, we can reduce the range of pixel intensities of medical image slice to patterns of dots.

The simplest case of halftoning we implemented consisted of 9 halftone masks, each covering a 3x3 pixel window. In this simple case, an average intensity over a 3x3 window is calculated and replaced by a mask of dots, depending on the average intensity I_{ave} . For example, if I_{ave} value falls in the range [225, 256), the pixel window is replaced with mask with 9 dots within the square (Figure 4 37).

Figure 4.37: Halftone masks replacing intensity value with a pattern of uniformly distributed dots.

Applying the halftone masks as shown in Figure 4.35 to the entire image produces the following halftones (Figure 4.38). From these examples, it is evident that simple halftone masks are capable of capturing and preserving the anatomical features present in the image.

Figure 4.38: Original CT scans (a,c) and the corresponding halftones (b,d).

Figure 4.38 (Continued).

4.2.1.1 Point Density

In mathematics a point density of a set is the ratio of the measure of the part of the set in the neighborhood to the measure of that neighborhood. Density of a set E that is measurable on the real line $\mathfrak{R}$ at a point x can be expressed at the ratio:

$$\frac{mes(E \cap \Delta)}{|\Delta|}$$

where Δ is any segment containing x and $|\Delta|$ is its length. One can introduce the density in n -dimensional space, where the lengths of the segments in $\mathfrak{R}$ are replaced by the volumes of the corresponding n -dimensional parallelepipeds.

Point density defines the concentration of point features within each predefined neighborhood, which in our case is a single pixel or a voxel (Figure 4.39).

Figure 4.39: Neighborhood is defined as either a pixel or a voxel. The number of points that fall within the neighborhood is totaled and divided by the area of the neighborhood.

It is important to emphasize that we do not seek to interpolate intensity values – instead, we wish to reproduce the intensity variation in the medical image through varying point density reconstructed at each pixel. A similar concept is used in cartography where a cartographic line usually takes the form of a discrete set of points identified by their position with respect to an arbitrary coordinate system [74]. As a result, the discrete set of points constitutes the core information about the line and the map that it is used to generate [74]. Similarly, important practical questions are tackled:

a) the size of a dataset that can be used to adequately and accurately represent the

information and b) the appropriate density distribution that will allow capturing all the essential detail at the level of detail required.

In a similar fashion, a point cloud can be generated based on the pixel intensities from either CT or MRI image. For each pixel or a set of pixels, a cluster of non-overlapping points is generated, to represent the value at that pixel. Since the range of intensities is finite and so is the number of internal organs of interest, it is relatively straightforward to reduce each color range to a fitting number of points to represent that color. Hence, the inherent contrast and heterogeneity present in the image slice is reduced to point cloud densities. Figure 4.40 illustrates the concept of generating a density point cloud based on the medical image: each pixel is represented with a point density within that pixel so that the features of interest are encapsulated in the point cloud.

Figure 4.40: The intensity value at each pixel is reproduced through a cluster of points associated with that pixel location.

Since depiction of the body interior with either MRI or CT technology is based on the concept of generating contrast between different organs depending on their biological

material properties thus making the absolute pixel intensities irrelevant, reproducing the same contrast with point density allows preserving the anatomical information of the medical image. Below is the algorithm that generates a point cloud for a medical image.

Algorithm 3: *Generate_Point_Cloud* (Image $I[n \times m]$, number of density point clusters k , number of points per cluster p)

```

compute  $p$  relative to  $k$ 
for  $i=0$  to  $n$ 
  for  $j=0$  to  $m$ 
    compute  $p_{I[i,j]}$ 
    for  $t=0$  to  $p_{I[i,j]}$ 
      generate random  $[x, y]$  within neighborhood  $I[x, y]$ 
    end for
  end for
end for
return Point Cloud

```

In order to reinforce the key features, points from the edge map that is generated by an edge detector can be added to the point cloud, as shown in Figure 4.41.

Figure 4.41: Original CT scan (a) and its edge map (b) combined to generate a density point cloud (c).

Figure 4.42 demonstrates the point cloud for an MRI image of pancreas region. MRI data provides excellent contrast for soft tissue and reproduces density variations

within the tissues and the neighboring organs through a range of intensities. To adequately capture the contrast across the slice, we generated 16 distinct classes of point densities, with higher intensities corresponding to higher point concentrations for that pixel.

Figure 4.42: MRI of pancreas (a) and corresponding density point cloud (b).

In Figure 4.43, an image of a cross section from the bovine knee was used. The image shows the trabecular bone structure within the joint. Interestingly, the intensity values do not value greatly across the image. The distribution of features in the image is more important in describing the functional quantities of the bone and the point cloud produced from the image (Figure 4.43(b)) captures the features and their distribution across the bone very well. The image is courtesy of Brian Richmond, PhD, who is using experimental data and image data to relate the trabecular bone structure with the functional loading of joints.

Figure 4.43: Trabecular bone structure of bovine knee and corresponding density point cloud.

Point density is adaptive and can be adjusted based on the application, the source of data, accuracy requirements and the computing capabilities. The relationship between the intensity value and the number of points generated per pixel neighborhood is usually linear and the coefficient of the relationship can be adjusted. Thus, it is possible to match the intensity value directly with the number of points per neighborhood, or reduce intensities to a number of point density cases. The denser the point cloud, the more detail is it capable to capture, but it also implies more data to process and fit. A denser point cloud is not as remarkable to display (Figure 4.44(b)) as it starts to appear to lose the overall appearance of the image but zooming into the denser point cloud illustrates that individual features of the anatomy are well preserved (Figure 4.45(b)).

Figure 4.44: Denser point cloud captures more tissue variation detail.

Figure 4.45: Close-up of the same denser point cloud reveals presence of important anatomical detail

Therefore, prior knowledge such as system capabilities and accuracy requirements can be incorporated into the point cloud generating algorithm to adjust the density range of the point cloud automatically, which will allow adequately differentiating between

finer variations in the image intensities. Figure 4.46 illustrates different point clouds generated from the same CT scan.

Figure 4.46: CT scan (a) and corresponding density point clouds with a varying degree of point assignment. Increasing point density per pixel (c) will allow capturing more fine detail and variation of material across the slice at the cost of increasing the dataset.

Similarly, a point cloud can be built from density data rather than or in combination with medical images. For example, bone mass data produced by bone mineral densitometry (BMD), combined with MRI or CT data of the bones, can be used to generate a heterogeneous model of a bone. Such a heterogeneous bone density model can be used as a tool to assess osteoporosis, bone strength and risk of future fracture [16].

4.3 Point Cloud Approximation with B-Spline Surface

The point cloud generated from an image captures, through point densities, the information about the anatomical features present in the original image. However, similarly to the image, point cloud is a discrete representation of the knowledge and thus suffers from the same drawbacks as voxel models.

Instead, we want to generate a continuous mathematical model that will capture the heterogeneity of the data using B-Splines. Due to the nature of B-Spline basis functions, the target representation scheme will guarantee smoothness and infinite precision and applicability to material features of arbitrary dimension. The goal is not to replace an existing visualization scheme, but rather to enhance it with an analytical counterpart, a smooth heterogeneous B-Spline surface.

Pixels in a medical image slice are arranged in a regular two-dimensional grid. The point cloud generated from an image hence has a rectangular topology as well, as illustrated in Figure 4.47.

Figure 4.47: Rectangular point cloud grid with rows containing variable number of data points.

Thus, giving this topology, we can organize the point cloud according to the original image topology and approximate the rows of point density data with a set of individual parallel B-Spline curves. Figure 4.47 shows the organization of the point cloud data using the rectangular grid.

Figure 4.48: Point cloud data is organized using a rectangular grid and each row of data is approximated with a B-Spline curve.

Once an array of parallel B-Spline curves is generated, we can loft these curves with a B-Spline surface. Surface lofting, also known as skinning, is a process of passing a smooth surface through a set of cross-sectional curves. Given a set of cross-sectional curves $C_k(u), k = 0, \dots, K$, a B-Spline surface is sought that either interpolates or approximates these curves at given parameter values, meaning that these curves form the iso-parametric curves of the surface.

Choosing v direction for lofting, the algorithm that computes a skinned surface follows [87]:

Algorithm 4: General_Surface_Lofting (Cross-sectional curves $C_k(u)$, number of control points n , degree of fitted surface in lofting direction q)

make input curves $C_k(u)$ compatible
 $U \leftarrow$ knot vector for C_k curve
 $v_0, \dots, v_k \leftarrow$ compute parameters for curve approximation
for $i = 0$ **to** n
 for $j = 0$ **to** k
 $Q \leftarrow i^{\text{th}}$ control point of C_j
 end for
 $\{P_{i,j}\}_{j=0}^k \leftarrow$ approximate $\{Q_j\}_{j=0}^k$ with a degree q curve
end for
 $V \leftarrow$ compute knot vector by averaging v parameters
return approximating surface $S(u, v)$

Lofting allows a lot of flexibility meaning that cross-sectional curves can be of any degree and can be defined over different knot vectors [87]. Figure 4.49 demonstrates an example of lofting a B-Spline surface through 3 cross sectional curves of different degrees and directions.

Figure 4.49: Lofting offers a lot of flexibility.

With flexibility comes the price of making the curves compatible, which requires that the curves all be of the same degree and defined over the same common knot vector.

The algorithm that makes curves compatible is outlined as follows [87]:

Algorithm 5: *Make_Curve_Compatible* (Cross-sectional curves $C_k(u)$)

make curve definitions compatible
scale knot vectors to a common interval
find the highest degree in $C_k(u)$
raise the degree of the lower degree curves if needed
merge individual knot vectors
refine cross-sectional curves
return compatible curves $C_k(u)$

The compatibility process results in the extremely high number of control points. For example, for k cross sectional curves with an average number of control points n , the total number of control points can range anywhere between $O(kn)$ and $O(k^2n)$ [89]. Given that an average medical image slice is 512 x 512 pixels, the number of points in the density point cloud that is generated from the image will quickly lead control point explosion. For example, in Figure 4.50, four cross sectional curves are lofted. During the compatibility process, the number of control points is increased for each contour to 42.

Figure 4.50: General lofting through contours with various number of control points results in an overall high number of control points.

Density point cloud generated from a medical image is arranged in a $n \times m$ rectangular grid fashion. However, the number of points and their distribution in each row differs since the point density for each pixel depends on the value of the intensity of that pixel.

Medical data heterogeneity is quite complex and varies throughout the image, and hence the traditional ‘point carpet’ topology cannot be obtained. Surface fitting to point cloud data has been investigated in CAD/CAM both for visualization and reverse engineering applications [91-93], yet many questions such as surface parameterization of cross sectional curves and the control point explosion remain an issue. The rows of data in point-cloud data often contain different numbers of data points and the distribution of the points in each row varies. Thus, in this part of the research, we focus on the following problem of heterogeneous B-Spline surface fitting. Given a point cloud dataset of the form:

$$Q_{i,j} \quad i = 0, \dots, n \quad j = 0, \dots, m_i$$

a degree (p, q) B-Spline surface is sought that approximates the point cloud data up to a user specified tolerance ε while preserving the heterogeneity of the data.

The fitting method produces a $C^{(p-1, q-1)}$ continuous surface that does not deviate from the user-specified tolerance following the steps [89]:

1. Fit curves to the rows of data points, with each curve capturing the distribution of the points along that curve, and independently of each other
2. Fit a surface to the cross-sectional curves produced in Step 1, within a fitting tolerance from any curve, while avoiding data explosion

Curves are fitted to rows of data independently of each other, with the stress on preserving the heterogeneity of the point set along the line. Since the number of points differs depending on the density of the tissue inferred from the pixel intensity, the data is arranged for fitting in a ragged array. Thus, the point cloud for each medical slice is organized in the array with dimensions $P[n][m[n]]$ where the array m keeps track of the size of each row of data in the ragged array P .

Data explosion is addressed through the proper control of the knot vector – fuzzy knot approach, covered in the following section.

4.3.1 Heterogeneous Cross Sectional Contour Fitting

Given the following desired point cloud arrangement

$$\begin{aligned} &P[0][0], \dots, P[0][m[n]] \\ &P[1][0], \dots, P[1][m[1]] \\ &\vdots \\ &P[n][0], \dots, P[n][m[n]] \end{aligned}$$

a hybrid data structure is created that arranges the point densities according to their common Y coordinate. Here, we follow the notation for image coordinate indexing, with (x_0, y_0) being the coordinate of the upper left corner.

Figure 4.52: Data structure arranging the dataset for cross sectional fitting.

Now, given the data points $Q_0, \dots, Q_m$ that fall within the cross section of the density points cloud, we are seeking a B-Spline curve that will approximate the data in the least-square sense.

In order to capture the heterogeneity of the point cloud data, the parameters for curve approximation need to be properly selected. Chordal length parameterization is used widely in geometric modeling application; however, for the B-Spline function to encapsulate the point distribution along the scan line, the parameters need to be uniformly spaced.

Uniform parameterization, on the other hand, is the simplest and least computationally expensive method for calculating the parameters. Using uniform parameterization, the difference between successive knots is the same, regardless of the actual length of the curve segment:

$$\begin{aligned}
t_0 &= 0 \\
t_i &= t_{i-1} + \frac{1}{k}, \quad i = 1, \dots, k-1 \\
t_k &= 1
\end{aligned}$$

This method is often deemed unsatisfactory for engineering applications [19]. However, we empirically established that when modeling density variations of medical data, uniform parameterization was the most fitting for capturing the density distributions of the data points along the line. This observation is theoretically sound since given a set of uniformly spread parameters, the B-Spline function encapsulates the point distribution along the scan line and maps the uniformly spaced parameters *onto* the points in the density clusters.

Figure 4.53 illustrates the concept of fitting to rows of intensity values. For each row of intensity values (Figure 4.53(a)), a row of point densities is generated, as part of overall point cloud (Figure 4.53(b)). Points from the row of densities (Figure 4.53(b)) are uniformly parameterized and approximated in a least squares sense with a B-Spline curve.

Figure 4.53: Point densities (b) are generated from the row of pixel intensities (a).

Fitting a B-Spline curve to approximate a cross section of the density point cloud requires a number of control points as an input. Since the density map produced from pixel data contains a large number of points, the choosing the right number of control points for approximation is a balancing act between accuracy and data reduction. Hence, we want to determine the number of control points n so that the curve approximates the density cross section within a given tolerance eps .

The adequate number of control points needed for approximation so that the curve does not deviate from the data points beyond the tolerance nearly impossible to guess; however it is needed as an input for approximation routine. One of the approaches is to interpolate first, following it up knot removal [95] produces fairly close estimate but the heterogeneity preservation of the fitted curve is negatively affected in the process. Instead, starting with initial guess approximation instead of interpolation produces better results [95] while the point density distribution is not affected.

The algorithm to compute an adequate number of control points n starts with the initial estimate, calculated as a user specified percentage of data points to be approximated. The percentage input can be adjusted by the user according to the complexity the data, or be chosen based on the prior knowledge such as the type of original scan (CT, MRI) and the grade of change in density based on the region of interest such as tissue type and location.

Approximation within the tolerance eps is evaluated by computing the maximum parametric error ε -the maximum distance between the data points and the curve evaluated at corresponding parameters: $\max |Q_i - C(u_i)|$. The algorithm to compute the number of control points as an iterative process:

- If the input estimate is inadequate , i.e. computed parametric error $\varepsilon > eps$, the number of control points is increased by calculating the increment $\Delta n = \frac{n}{2}$ and increasing the total index $n = \min\left(n + \Delta n, \frac{n+k+1}{2}\right)$
- Similarly, if the parametric error $\varepsilon < eps$, the number of control points can be increased without suffering the loss of accuracy. The search for a lower index n is performed in the left interval, $\left[n_{\min}, \frac{n}{2}\right]$

Having approximated the individual rows of data with cross sectional B-Splines, we want to make certain that the B-Spline function does indeed encapsulate the density distribution of points along the fitted line. A fitted curve is evaluated at uniform parameters and the sampling demonstrates that the B-Spline captures and faithfully preserves the point distribution.

Figure 4.54 demonstrates the concept. Using artificial pixel intensities data, we construct a row of point densities to reproduce the distribution of features in pixel domain. Lighter colors represent higher density of material in CT and MRI scans and thus we generate a denser cluster of points per pixels with higher intensity value. Likewise, lower intensity value represents lower material concentration and point assignment is adjusted accordingly. The density data (Figure 4.54(b)) is approximated with a B-Spline curve and then sampled back uniformly. To sample the curve, a B-Spline curve is evaluated at a given parameter by evaluating all non-vanishing basis functions and multiplying them by appropriate control points.

The result of the sampling is displayed in Figure 4.54(c); it is evident that when evaluated at evenly spaced intervals, B-Spline function reconstructs the distribution of the original point set.

Figure 4.54: Row of pixels (a) followed by the point densities (b) and the result of curve sampling (c).

Figure 4.55 contains the example of an individual curve from an actual CT dataset. The points (Figure 4.55(a)) are fitted and uniformly sampled (Figure 4.55(b)), with distribution closely resembling the original distribution.

Figure 4.55: Cross sectional curve from a CT dataset (a) and curve sampled at uniformly spaced parameters (b).

The original dataset in Figure 4.56(a) is approximated with a B-Spline curve. Once we fit the curve, we evaluated it at uniformly spaced parameters, but varying the number of sampling points.

Figure 4.56: Fitted B-Spline curve is evaluated by varying the number of sampling points. Original point set (a), followed by samplings of the fitted curve.

In Figure 4.56(b), the curve is evaluated at 10 equally spaced parameters. With just a sparse number of points, a B-Spline function attempts to redistribute the computed points according to the density distribution of the original dataset. Increasing the number of sampling points to 50 in Figure 4.56(c), 100 in Figure 4.56(d) and 300 in Figure 4.56(e) better illustrates the behavior of the B-Spline function since larger number of points is able to emphasize the point distribution. Thus, the B-Spline function reorganizes evenly spaced sampling parameters according to the encapsulated data distribution along the fitting line.

4.3.2 Fuzzy Knot-Vector Technology

CT or MRI procedure generates, on average, more than 2 million voxels per patient examination [9]. The amount of data needed to process is vast, as a result. While each scan line of the point cloud is fitted independently of each other, the high number of data points can lead to data explosion during the lofting procedure. Specifically, the process of making curves compatible for lofting can lead to an exponential increase of control points. Knots that located very close together still appear as distinct knots, which quickly lead to a huge database of control points and subsequent numerical problems.

When calculating the right knot vector for approximation, it is important to choose a knot vector that supports a numerically stable system of equations and also produces a high quality approximating curve [20]. At the same time, the position of each individual knot is flexible locally, i.e. the knot can be moved within a tolerance interval without affecting the underlying point distribution, the quality of the curve or the numerical stability of the system of equations.

Thus, we want to define an interval in which an individual internal knot can be moved without affecting the quality of the fitting. The idea is to use a knot vector U^p of degree p and bracket it using a knot vector of degree $p-1$, denoted as U^{p-1} .

We can define the interval of perturbation for each knot – a ‘fuzzy interval’ $[a, b]$ as follows:

$$\begin{aligned} u_{i-1}^{p-1} &< u < u_i^{p-1} \\ a &= (1 - per) * u_i^p + per * u_{i-1}^{p-1} \\ b &= (1 - per) * u_i^p + per * u_i^{p-1} \end{aligned}$$

where per is a variable that controls the amount of perturbation allowed per each knot. For example, $per = 100\%$ allows the knot to move within the entire ‘fuzzy’ region, while setting $per = 0\%$ prohibits perturbation.

The approximation algorithm accepts the data points and a knot vector as the input.

Algorithm 6 *Fuzzy_Curve_Fit* (row of point cloud data, $Q_{i,j}, i = 0, \dots, k$, Master knot $\hat{U} = \{\hat{u}_0, \dots, \hat{u}_m\}$, number of control points n , degree of approximating curve p , percentage of the interval usage allowed per)

$T = \{t_0, \dots, t_n\} \leftarrow$ uniformly parameterize the input data

$U^p = \{u_0^p, \dots, u_{n+p+1}^p\} \leftarrow$ knots for degree p

$U^{p-1} = \{u_0^{p-1}, \dots, u_{n+p}^{p-1}\} \leftarrow$ knots for degree $p-1$

$S = \{s_0, \dots, s_{n+p+1}\} \leftarrow$ allocate memory for approximating knots

for $i = p+1$ **to** n **do**

$a = (1 - per) * u_i^p + per * u_{i-1}^{p-1}$

$b = (1 - per) * u_i^p + per * u_i^{p-1}$

$\hat{u}_l \leftarrow$ input knot closest to u_i^p

if $(\hat{u}_l \in (a, b))$ $s_i = \hat{u}_l$ **else** $s_i = u_i^p$

end for

$C(u) \leftarrow$ approximate data $Q_i, i = 0, \dots, k$ with B-Spline using T and S

$\hat{U} \leftarrow \text{Merge}(\hat{U}, S)$

return approximating curve $C(u)$, updated Master knot $\hat{U}$

A *master* knot vector is the vector that will be passed to each curve approximation. It acts as a ‘bookkeeper’ of all knots encountered during the individual fittings. Even though individual cross sectional curves are fitted independently of each other, during the compatibility process required before the lofting routine all the knots are merged. Thus, the objective of the bookkeeping master knot is to reuse as many knots as possible between individual fittings.

Figure 4.57: The “densest” row of data is chosen as the basis for master knot calculation.

Ideally, the initial master knot vector should contain as many candidate internal knots that are available to be reused in individual fittings as possible. The row of scan line data with the highest rate of density variation is chosen as the base for the master knot calculation (Figure 4.57). To identify such a scan line density variation rate as well as the range of densities that are represented in that one scan line are evaluated, for each row of data, before the fittings are performed.

For each cross sectional curve, the algorithm computes two knot vectors, U^p and U^{p-1} , in order to bracket U^p knot into ‘fuzzy’ flexibility intervals. For each cross sectional curve, a check is performed against each knot: if the master knot vector contains a knot within the flexibility interval of the fitted knot, the knot from the master knot vector is selected. On the other hand, if the knot present in the fitted curve is not represented in the master knot vector, that knot is copied over to the master knot vector.

Below is a pictorial representation of the algorithm. The master knot is passed during each independent row approximation. For each row of data being approximated,

two starting knot vectors are set up, of degree p and $p-1$, to define knot flexibility intervals.

Figure 4.58: Fitted curve knot vector is updated with a knot from a master knot repository.

Thus, each internal knot in knot vector U^p is bracketed by the U^{p-1} knot vector. In Figure 4.58(a) U^p knot vector is displayed. To illustrate the algorithm, we choose one individual internal knot and display its interval as red brackets. As the algorithm reaches the knot, it checks the knot interval against the master knot and chooses the knot from it that is closest to the original knot from Figure 4.58(a). If finds a fitting knot, the algorithm replaces that knot with the knot from mater knot's repository. Figure 4.58(b) shows that the knot from the master knot replaces the original knot.

An alternative scenario is also illustrated, in Figure 4.59. For a particular internal knot in Figure 4.59(a), the bracket interval is checked against the master knot. Since the master knot does not contain a candidate knot in that region, the original knot is kept, as seen in Figure 4.59(b). The knot is also added to the master knot repository.

Hence, both the fitted knot vector and the master knot vector are examined and updated when necessary, during each curve approximation. As a result, the master knot

vector is passed and updated after each row approximation, acting as a bookkeeper of all common knots.

Figure 4.59: The master knot gets updated with a knot not represented in repository.

In Figure 4.60, 11 data points are approximated with a B-Spline curve. The resultant curve has 6 control points. The knot vector has 3 internal knots. Small bars display the knots for the knot vector U^p and larger dotted bars display the knots for knot vector U^{p-1} .

Figure 4.60: B-Spline curve with 6 control points. The knot vector U^p is bracketed by knots from the knot vector U^{p-1} .

Figure 4.61 illustrates a B-Spline curve approximating 13 data points and its control polygon and control points.

Figure 4.61: B-Spline curve with 7 control points. The knot vector U^p is bracketed by knots from the knot vector

The knot vector displayed in Figure 4.61, U^p is bracketed by using a knot vector of degree $p-1$.

Figure 4.62: Fuzzy knot vector algorithm reuses internal knots and reduces the number of knots in the common knot vector.

The knot vector with for the curve with a higher number of control points is selected as the master knot (Figure 4.62(b)) The knots for the second knot vector are bracketed, with blue brackets showing the width within the flexibility interval that can be

explored, based on the parameter, $per = 80\%$ (Figure 4.62(a)). After the knots from the second knot vector are compared against the master knot, no new knots are added to the common knot vector (Figure 4.62(c)). Thus, instead of having 7 internal knot vectors, the common knot vector only contains 4 knots.

The master knot algorithm allows re-using knots without having an impact on the density distribution, which results in a tremendous data reduction. Once all the cross sectional curves are fitted, they contain a large number of knots in common and the performing compatibility routine for surface lofting does not lead to data explosion. Employing the fuzzy knot approach allows, on average, 65-80% reuse of knots.

4.3.3 Fuzzy Surface Approximation

The set of cross sectional curves obtained using *Fuzzy_Curve_Fit* algorithm is fitted with the B-Spline surface. The objective of this step is to pass the B-Spline surface through a set of cross-sectional curves, while preserving the heterogeneity of the original point cloud dataset and continuing to deal with the issue of control point explosion.

Two tolerances, eps_u and eps_k , are used to control the fitting process. Tolerance eps_u is used to approximate columns of data in the v direction, and eps_k is used to perform a global knot removal of the final surface. Both tolerances can either be established empirically depending on the density of the point cloud and the complexity of the anatomical shapes present in the medical slices, as well as precision required by the application. The user has the control, however, to change and adjust the tolerances according to the precision and complexity requirements.

Algorithm7 *Fuzzy_Surface_Fit* (tolerance for final knot removal eps_k , tolerance for v direction approximation eps_u , required degrees for surface fit (p, q) , set of n cross-sectional curves $curU$, control point counts for n cross-sectional curves $m[n]$, percentage of the interval usage allowed, for v direction approximation per)

$\hat{V} \leftarrow \text{NULL}$, knot vector for v direction approximation
for $i = 0$ **to** n
 for $j = 0$ **to** $m[i]$
 $R_j \leftarrow j^{th}$ control point of $curU[j]$
 end for
 $curV[j] \leftarrow$ approximate $R_i, i = 0, \dots, n$ using *Fuzzy_Curve_Fit*, passing $\hat{V}$ and eps_u
end for
 $[V, m] \leftarrow$ make $CurV[v], j = 0, \dots, n$ compatible
 $S(u, v) \leftarrow$ remove all knots using eps_k
return approximating surface $S(u, v)$

The algorithm presented thus approximates a point cloud of density data generated from medical images with a $C^{(p-1, q-1)}$ degree (p, q) B-Spline surface. The method approximates the rows of density data independently of one another, calculating an optimum number of control points for the fitting within the user specified tolerance. The independent approximation of rows of point cloud data eliminates wiggles in the surface fitting [95] while uniform parameterization allows the curve fitting to capture the original distribution and the density of the points.

Figure 4.63: Conventional lofting vs. fuzzy surface fit

The fuzzy knot vector method produces a very slow, logarithmic-like, growth of control points as the function of tolerance, while traditional lofting methods would require an order of magnitude more control points in both directions. In Figure 4.63 the same set of curves is lofts using different approaches. Since the curves have various number of control points and knots, during the conventional lofting process the curves are made compatible. Thus, each contour now had 42 control points after the compatibility process. The fuzzy surface fitting allows reusing the knots and as a result each cross section has 19 control points after the compatibility process.

Lofting through cross sectional curves that encapsulated the density distribution continues to preserve the density variations of the original dataset.

Figure 4.64: Example 1: the effect of increasing the number of control points.

In Figure 4.65, control points from the fitted surfaces are visualized. In Figure 4.64(a) the number of control points is defined by user to be 2% of the total number of data points. In Figure 4.64(b), the number is set to 5% and in Figure 4.64(c) – to 15%. It is obvious that with increasing the number of control points, the capture of the detail improves. Interestingly, setting the number of control points to 15% of the data point count allows capturing the major detail rather well while purging 85% of the data.

Figure 4.65 illustrates the same behavior for a different dataset.

Figure 4.65: Example 2: the effect of increasing the number of control points.

4.3.4 Point Density Rendering

Once the heterogeneity of the point dataset is captured with a B-Spline surface function, it is possible to interrogate the surface for the information that the surface has encapsulated. One of the simplest yet useful tools of interrogation is reverse sampling. Here, we evaluate the surface at uniformly spaced parameters and reverse engineer the original point cloud back.

Figure 4.66: Original point cloud (a) and uniformly sampled surface (b).

This uniform sampling is the reverse process of the fitting process – generated point cloud produced can be visualized automatically to produce either a grayscale or a color map of the B-Spline surface.

In Figure 4.66(a) the original point cloud is compared against the sampled surface (Figure 4.66(b)). The original point cloud is dense since it is generated from MRI data and contains a lot of anatomical detail we want to preserve. Although the surface is sampled at uniformly spaced parameters, the surface function redistributes the parameters during the evaluation process according to the density distribution is captured during the fitting process.

When sampling the surface, it is apparent that the B-Spline function is preserving the original distribution of the points faithfully. Having obtained the point cloud back, it is possible to visualize the captured densities by assigning color based on the densities that fall within the pixel equivalent.

Assigning colors back can be done through either examining a histogram of density distributions (Figure 4.67) or by assigning the intensities based on an absolute count of points that fall within the pixel region.

Figure 4.67: Sampled point cloud (a) and the histogram of point density distributions (b).

Table 4.2 compares the value of the intensity of the original image at given location $[x, y]$ with the number of points that fall within the pixel region during sampling.

Table 4.2: Point density per pixel vs. pixel intensity

$[x, y]$	$I[x, y]$	x2 density	x3 density	x4 density	x5 density
222,195	255	10	15	15	25
284,251	58	3	4	4	7
177,218	92	7	7	7	11
299,272	88	4	5	5	8
313,341	59	3	4	4	7
217,374	71	4	5	6	8
248,246	131	5	8	9	15
242,248	155	6	9	10	13

In Table 4.3, the same process is repeated over a larger window. An average intensity over the [3x3] window is calculated, with $[x, y]$ being the center point. The point density is then measured as the total number of points that fall within that [3x3] window.

Table 4.3: Point density over [3x3] pixel window vs. pixel intensity

$[x, y]$	$I[x, y]$	x2 density	x3 density	x4 density	x5 density
222,195	255	86	130	171	215
284,251	60.89	263	38	52	66
177,218	90.67	41	60	80	99
299,272	84.33	31	47	62	73
313,341	58.44	25	38	52	65
217,374	70.56	29	46	62	78
248,246	126.44	42	64	83	105
242,248	150	49	72	97	1213

The columns 3 through 6 contain results of sampling with consistently increasing the number of parameters at which the surface is evaluated, by a constant factor. As the number of samples increases, the number of points that fall within a specific region increases linearly and rather consistently. It is important to note there is a linear constant that relates point density and the value of intensity, $I(x, y) = \alpha * PD$

This not only confirms that B-Spline surface function captures and encapsulates the intensity variations throughout the image faithfully but also provides a way to render the point cloud. Thus, after the surface has been uniformly sampled, the color can be assigned based on individual point densities and a ratio variable, which can either be

adjusted by the user or deduced based on the density of the sampling. As the rate of the sampling increases, the number of points that fall within the individual pixel approaches the intensity value.

Figure 4.68: Original MRI images (a, c, e) and density maps produced by sampling fitted B-Spline surface function (b,d,f).

Figure 4.68 displays the results of point density rendering for MRI data of the brain. The left column contains the original medical image and the right column contains the rendered version of the fitted B-Spline surface.

Figure 4.69: MRI of the pancreas (a) and the density grayscale map of the B-Spline surface (b).

Figure 4.69(a) shows the original MRI image of the pancreas region, which is full of intricate detail and intensity variation throughout the image slice. B-Spline surface is sampled and rendered automatically. Figure 4.69(b) displays rendered point densities of the B-Spline surface which captures all the essential detail present in the original image.

Similarly, color can be assigned based on the point densities by using an HSI (Hue Saturation Intensity) color model. Since we are relating point densities and the intensity with a ratio, using an RGB model does not allow directly relating the densities and the assigned color. Using HSI color model, we determine intensity component while keeping hue and saturation constant as a desired level. Figure 4.70(c) shows an example of rendering point densities from a fitted B-Spline surface using HSI color model.

Figure 4.70: MRI of the brain tumor (a). Surface point densities rendered in grayscale (b) and in color (c).

Figure 4.71 shows an image from Visible Human Dataset. The image is an actual photograph of a cross section of a male cadaver. This is a large (9.5MB) high resolution image (2048x1216 pixels) containing a lot of detail. Some of the background is cropped in order to be able to fit the image on the page.

Figure 4.71: Visible Human Dataset, high resolution.

In order to generate a point cloud, the RGB image was converted to grayscale (Figure 4.72(a)). Figure 4.72(b) contains the grayscale rendering of the surface point densities, which demonstrates how well the B-Spline surface preserves the density variation of the point cloud therefore the intensity values across the original image slice.

Figure 4.72: Visible Human Dataset image converted to grayscale (a). Fitted surface is uniformly sampled to produce a grayscale map (b) and a color map.

In order to compare original images against the rendered point cloud we subtracted one image from another. Figures 4.73(a) and 4.74(a) show the original brain MRI. Figures 4.73(a) and 4.74(b) display the grayscale rendering of the B-Spline surface point densities where colors are assigned automatically by normalizing on the ratio and the distribution of the point densities across the sampled point cloud. In Figures 4.73(c) and 4.74(c), the rendered image is subtracted from the original image.

Figure 4.73: Example 1: comparing the original image and the grayscale colormap. Original MRI image (a) subtracted from grayscale colormap (b) to produce image difference (c).

The above results are encouraging and display a constant intensity value throughout the image indicating that the offset between the rendered intensities and the original intensities is consistent and constant throughout the image. This, in fact, demonstrates that B-Spline surface captures and encapsulates the intensity distribution of the original image, as intended.

Figure 4.74: Example 2: comparing the original image and the grayscale colormap. Original MRI image (a) subtracted from grayscale colormap (b) to produce image difference (c).

Chapter 5 - Conclusions and Future Work

In this dissertation we investigated the use of B-Spline functions as an analytical basis for a representation scheme of live materials. We outlined the theoretical foundation for a heterogeneous CAD model and developed the algorithms required to convert discrete image data into a continuous mathematical model.

With the advent and ongoing developments in medical imaging, medicine has entered into a new era of diagnosis, prevention and management of diseases. As medical imaging continues to play a crucial role in medical practice, the medical community is pushing for new applications that will offer solutions beyond data display and visualization.

Although medical images offer unparalleled view of human anatomy and have proven to be an indispensable tool for visualization purposes, the discrete nature of pixel data limits the applicability of pixel and voxel models in such crucial areas of research as analysis, modeling and simulation. CAD field on the other hand has served the engineering community for decades, providing modeling solutions and analysis tools. CAD has relied on the extensive body of research and knowledge involving various representation schemes and their applications. Bio-CAD is a new and exciting interdisciplinary field that promises to combine the innovations in medical imaging and well tested and proven reliable tools from engineering design.

Bio-CAD is centered on the goal of building a 3D heterogeneous analytical model of human anatomy that will incorporate not only the geometry of the individual patient's body and internal organs but also the material information about the constituent tissues. A 3D model of the patient will allow the medical practitioner to store all the vital information about their patient electronically and can be used as a tool for health monitoring and treatment. The mathematical nature of this model will make it useful for analyzing the progression of diseases and the response to treatments, monitoring and comparing vital organ and tissue statistics since information such as volume and surface area of the individual components can be computed.

Converting the patient specific scan data into a 3D heterogeneous CAD model is an exciting step towards improving and enhancing medical practice. Applications for Bio-CAD models include but are not limited to surgical simulation and virtual body exploring, simulation and surgery planning. Due to the wealth of engineering tools in both CAD and CAM fields, applications such as manufacturing and design of patient specific prosthetic devices and extracellular matrix design for tissue regeneration applications can also be possible.

In this work, we laid down the groundwork for building accurate heterogeneous models from patient specific data. Using B-Spline function representation, we outlined the methodology for representing topologically irregular and complex biological entities in a precise and continuous mathematical form. The mathematical model we were able to implement is an important step towards the ultimate goal of creating a complete 3D heterogeneous model of human anatomy from medical image scan data.

Building CAD models from discrete medical image data remains a challenge due to the nature of biological data. Medical image data is abundant in intricate yet important detail and since image representation is discrete, automatic interpretation of the images continues to be a difficult and often unsolved task. For example, detecting and describing boundaries of internal organs and regions of interest in medical data is a much needed step for creating a CAD model; yet, it is a fundamental research problem in medical image processing. In the first half of this research work we thus focused on combining techniques from medical image processing and CAD for extracting boundaries and representing them accurately and analytically with B-Spline curves.

The process of generating B-Spline curves involved such important pre-processing steps as detecting the boundary, ordering the dataset and smoothing it using a variation of Laplacian smoothing. We investigated the Bezier-B-Spline relationship in order to capture the intricate details and the exact shape of the original contour. The mathematical relationship between these curve representation schemes allowed pre-processing the parameters for B-Spline fitting thus guiding the B-Spline fitting automatically. The results demonstrated the efficiency and the high fidelity of the fitted curves.

In the second part of the research we focused on the concept of data heterogeneity. Since biological materials are naturally heterogeneous it is imperative to incorporate the inherent heterogeneity into the representation scheme. Currently CAD models are built on the premise of homogeneity with the main focus on geometry of the modeled objects. Although heterogeneous modeling is currently an active research area, modeling biological entities remains an unsolved problem due to the degrees of freedom

and the complex distributions of materials that they possess. The existing CAD solutions for modeling heterogeneous models tend to either oversimplify the material distributions or model them as multi-material objects. We proposed an alternative approach of capturing varying heterogeneity present in medical image data with B-Spline functions.

In Chapter 3 we discussed in great detail the fundamentals of medical imaging and image generation and investigated the relationship between pixel intensity values and material densities of the materials. Based on that, we reproduced the contrast and features, expressed in terms of intensities in the images, through point densities and generated point clouds that captured heterogeneity of medical image data.

Having reduced intensity information to point density distribution cloud we then investigated and developed methodology for encapsulating the heterogeneity analytically with a B-Spline surface.

As part of development of the B-Spline surface model we first investigated whether B-Spline curves were capable of encapsulating point density distributions during the process of approximation. Individual cross sections of the point cloud were approximated with B-Spline curves which required establishing the appropriate parameterization method and an adequate number of control points. We tested the efficacy of B-Spline curves to preserve heterogeneity by evaluating fitted curves. Sampling the curves at uniformly spaced parameters demonstrated that B-Spline curves preserved the original distribution of points faithfully.

After the set of cross sectional approximating curves was created, we investigated the process of lofting through the cross section with the focus on preserving the point density distribution. Since point clouds generated from image data contain a large

number of points lofting through cross sections of individually fitted B-Splines can lead to data explosion. We approached this problem and avoided data explosion through fuzzy knot vector methodology.

Once we were able to fit a B-Spline surface to point density data, we were able to test and visualize the densities that were encapsulated. Our rendering evaluated the surface at equally spaced parameters and the B-Spline surface function spaces the parameters according to the heterogeneity preserved in that function. Reverse engineering the intensity assignment allowed generating images that are close replica of the original data.

The B-Spline surface model built based on medical image data is by no means a replacement of that data but rather a mathematical model to enhance it and to extend the capabilities of the existing digital representation. Since the application of digitized representation of anatomy is limited to mostly visualization, having an analytical representation scheme augments the image representation due to the attractive analytical properties of B-Spline such as infinite precision, continuity and smoothness.

Since B-Spline surface models are built on top of the medical data, they are limited to the knowledge contained in the images. MRI coefficients and CT numbers obtained during scanning are normalized and discretized during an image generation process during which some detail is lost and noise is introduced. Our representation scheme can theoretically be built directly based on the density and material information obtained from scanning devices thus bypassing the problems that plague discrete image representation. Data such as bone mass density which provides the information about the condition of the bones and can be used for assessing bone disorders such as osteoporosis

can also be used for generating point clouds. Since point cloud generation is resolution independent, data collected at macroscopic levels can be converted and modeled with B-Spline functions.

Thus, in this work we investigated a challenging problem of generating a mathematical model to represent discrete medical image data. We were able to successfully outline and implement a B-Spline surface fitting methodology that captures and encapsulates the heterogeneity of the medical data while extending the existing discrete representation with a mathematical scheme.

References

1. A. Biswas, "Construction of distance field based representations of material properties," Ph.D Dissertation, Dept. Mech. Eng., University of Wisconsin, Madison, WI, 2008.
2. F. Zhibin, "Image-guided modeling, fabrication and micromechanical analysis of bone and heterogeneous structure." Ph.D Dissertation, Dept. Mech. Eng., Drexel University, Philadelphia, PA, 2005.
3. X.Y. Kou and S.T. Tan, "Heterogeneous object modeling: a review," *Computer Aided Design*, vol. 39, pp. 284-301, 2007.
4. C. Schroeder, W. Regli, A. Shokoufandeh, and W. Sun, "Computer-aided design of porous artifacts," *Computer Aided Design*, vol. 37, pp. 339-353, 2005.
5. M.R. Stytz, G. Frieder, and O. Frieder, "Three-Dimensional medical imaging: algorithms and Computer Systems, " *ACM Computing Surveys*, vol. 23, no. 4, pp. 421-499, 1991.
6. M. Caon, "Voxel-based computational models of real human anatomy: a review," *Radiation and Environmental Biophysics*, vol. 42, no. 4, pp. 229-235, 2004.
7. H. Zaidi and B.M.W. Tsui, "Review of Computational Anthropomorphic Anatomical and Physiological Models," in *Proc. of the IEEE*, vol. 97, no. 12, pp. 1938-1953, 2009.
8. G.K. Knopf and R. Al-Naji, "Adaptive reconstruction of bone geometry from serial cross-sections," *Artificial Intelligence in Engineering*, vol. 15, no. 3, pp. 227-239, 2001.
9. B. Starly, A. Darling, C. Gomez, J. Nam, W. Sun, A. Shokoufandeh, and W. Regli, "Image based bio-cad modeling and its applications to biomedical and tissue engineering," in *Proc. of the ninth ACM symposium on Solid modeling and applications*, SM '04, pp. 273-278, 2004.

10. W. Sun, A. Darling, B. Starly, J. Nam, "Computer-aided tissue engineering: overview, scope and challenges," *Biotechnology and Applied Biochemistry*, vol. 39, no. 1, pp. 29-47, 2004.
11. G. Moustakides, D. Briassoulis, E. Psarakis and E. Dimas, "3D image acquisition and NURBS based geometry modeling of natural objects," *Advances in Engineering Software*, vol. 31, no. 12, pp. 955-969, 2000.
12. M.L. Giger, "Computer-aided diagnosis in medical imaging - a new era in image interpretation," *Next-Generation Healthcare. World Medical Association General Assembly, World Markets Research Centre*, pp. 75-79, 2001.
13. W. Sun, B. Starly, J. Nam and A. Darling, "Bio-CAD modeling and its applications in computer-aided tissue engineering," *Computer-Aided Design*, vol. 37, no. 11, pp. 1097-1114, 2005.
14. T. McInerney and D. Terzopoulos, "Deformable models in medical images analysis: a survey," *Medical Image Analysis*, vol. 1, pp. 91-108, 1996.
15. R.De Santis, L. Ambrosio, F. Mollica, P. Netti, and L. Nicolais , "Mechanical properties of human mineralized connective tissues," in *Modeling of Biological Materials* , Boston, Springer, pp. 211-261, 2007.
16. J.-C. Nebel, "Soft tissue modeling from 3D scanned data," *Deformable Avatars*, N. Magnenat-Thalmann and D. Thalmann, Ed., Kluwer, pp. 85-97, 2001.
17. B. R. Gomberg, P. K. Saha and F. W. Wehrli, "Method for cortical bone structural analysis from magnetic resonance images," *Academic Radiology*, vol. 12, no. 10, pp. 1320-1332, 2005.
18. T. Pavlidis, "Algorithm for shape analysis of contours and waveform," *IEEE Trans. PAMI*, vol. 2, pp. 301-312, 1980.
19. H. Park and J.-H. Lee, "B-Spline curve fitting based on adaptive curve refinement using dominant points," *Computer-Aided Design*, vol.39, pp. 439-451, 2007.
20. J. Koplowitz and S. Plante, "Corner detection for chain coded curves," *Pattern Recognition*, vol. 28, pp. 843-852, 1995.
21. R.G. Aarnik, J.J.M.C.H.De la Rosette, F.M.J. Debruyne and H.Wijkstra, "A preprocessing algorithm for edge detection with multiple scale resolutions," in *Proc. Engineering in Medicine and Biology Society*, vol. 2, pp. 903-904, 1996.
22. J. Cheng and F. Lin, "Approach of heterogeneous bio-modeling based on material features," *Computer Aided Design*, vol. 37, pp. 1115-1126, 2005.

23. J. Liu, S.K. Lau, V. A. Varma, R. A. Moffitt, M. Caldwell, T. Liu, A.N. Young, J.A. Petros, A.O. Osunkoya, T. Krogstad, B. Leyland-Jones, M.D. Wang and S. Nie, "Molecular mapping of tumor heterogeneity on clinical tissue specimens with multiplexed quantum dots," *ACS Nano*, vol. 4, no. 5, pp. 2755-2765, 2010.
24. V. Kumar, D. Burns, D. Dutta and C. Hoffmann, "Framework for object modeling," *Computer-Aided Design*, vol. 31, no. 9, pp. 541-556, 1999.
25. Y. K. Siu and S. T. Tan, "Source-based heterogeneous solid modeling", *Computer-Aided Design*, vol. 34, no. 1, pp. 41-55, 2002.
26. V. Kumar and D. Dutta, "An approach to modeling and representation of heterogeneous objects," *Journal of Mechanical Design*, vol. 120, pp. 659-667, 1998.
27. A.D. Bhatt and R.V. Warkhedkar, "Reverse engineering of human body: a B-Spline based heterogeneous modeling approach," *Computer-Aided Design and Applications*, vol. 5, no. 1-4, pp. 194-208, 2008.
28. A.D. Bhatt and R.V. Warkhedkar, "Material-solid modeling of human body: a heterogeneous B-Spline based approach," *Computer-Aided Design*, vol. 41, pp. 586-597, 2009.
29. X. Qian and D. Dutta, "Design of heterogeneous turbine blade," *Computer-Aided Design*, vol. 35, no. 3, pp. 319-329, 2003.
30. W. Martin and E. Cohen, "Representation and extraction of volumetric attributes using trivariate splines: a mathematical framework," in *Proc. Sixth ACM Symposium on Solid Modeling and Applications*, D. Anderson and K. Lee, Ed., ACM Press, pp. 234-240, 2001.
31. T.R. Jackson, "Analysis of functionally graded material object representation methods," Ph.D. Dissertation, Dept. Ocean Eng., Massachusetts Institute of Technology, Cambridge, MA, 2000.
32. K.-H. Shin and D. Dutta, "Constructive representation of heterogeneous objects," *Journal of Computing and Information Science in Engineering*, vol. 1, no. 3, pp. 205-217, 2001.
33. X. Qian and D. Dutta, "Feature-based design for heterogeneous objects," *Computer-Aided Design*, vol. 36, pp. 1263-1278, 2004.
34. V. Shapiro, "Theory and applications of R-functions: a primer," Technical Report, Cornell University, 1991.
35. A. Pasko, V. Adzhiev, B. Schmitt and C. Schlick, "Constructive hypervolume modeling," *Graphical Models*, vol. 63, no. 6, p. 413-442, 2001.

36. V. Adzhiev and E. Kartasheva, "Cellular-functional modeling of heterogeneous objects," in Proc. Seventh ACM Symposium on Solid Modeling and Applications, pp. 192-203, 2002.
37. B. Bickel, M. Baecher, M. Otaduy, W. Matusik, H. Pfister and M. Gross, "Capture and modeling of non-linear heterogeneous soft tissue," in Proc. ACM SIGGRAPH, vol. 28, no. 3, pp. 1-9, 2009.
38. A. Biswas, V. Shapiro and I. Tsukanov, "Heterogeneous material modeling with distance fields," Computer Aided Geometric Design, vol. 21, pp. 215-242, 2004.
39. V.L. Rvachev, T.I. Sheiko, V. Shapiro and I. Tsukanov, "Transfinite interpolation over implicitly defined sets," Computer Aided Geometric Design, vol. 18, pp. 195-220, 2001.
40. S.-M. Park, R.H. Crawford and J.J. Beaman, "Volumetric multi-texturing for functionally graded material representation," in Proc. Sixth ACM Symposium on Solid Modeling and Applications, pp. 216-224, 2001.
41. H. Liu, T. Maekawa, N.M. Patrikalakis, E.M. Sachs and W. Cho, "Methods for feature-based design of heterogeneous solids," Computer-Aided Design, vol. 36, no. 12, pp. 1141-1159, 2004.
42. B. Schmitt, A. Pasko and C. Schlick, "Constructive sculpting of heterogeneous volumetric objects using trivariate B-splines," The Visual Computer, vol. 20, no. 2-3, pp. 130-148, 2004.
43. W. Pompea, H. Worch, M. Eppel, W. Friess, M. Gelinskya, P. Greil, U. Hempele, D. Scharnweber and K. Schulte, "Functionally graded materials for biomedical applications," Materials Science and Engineering A, vol. 362, no. 1-2, pp. 40-60, 2003.
44. V. Kumar, S. Rajagopalan, M. Cutkosky and D. Dutta, "Representation and processing heterogeneous objects for solid freeform fabrication," in Proc. Sixth IFIP WG 5.2 International Workshop on Geometric Modeling: Fundamentals and Applications, 2005.
45. M. Chen and J.V. Tucker, "Constructive volume geometry," Computer Graphics Forum, vol. 19, pp. 281-293, 2000.
46. S. Angenent, E. Pichon and A. Tannenbaum, "Mathematical Methods in Medical Image Processing," Bulletin of the American Mathematical Society, vol. 43, pp. 365-396, 2006.
47. J. Prince, J. Links, Medical Imaging Signals and Systems, Prentice Hall, Upper Saddle River, NJ, 2005.

48. P.M. Evans, "Anatomical imaging for radiotherapy," *Physics in Medicine and Biology*, vol. 53, no. 12, pp. 151-192, 2008.
49. M. Caon, "Voxel-based computational models of real human anatomy: a review," *Radiation and Environmental Biophysics*, vol. 42, no. 4, pp. 229-235, 2004.
50. M. Bernstein and M. Berger, Ed., *Neuro-oncology: The Essentials*, Second Edition, Thieme, 2008.
51. J.B. Maintz and M.A. Viergever, "A survey of medical image registration," *Medical Image Analysis*, vol. 2, no. 1, pp. 1-36, 1998.
52. W. Xin and T. Yunxia, "A faster B-spline snake," in *Proc. 2009 IEEE International Conference on Robotics and Biomimetics*, pp.2314-2319, 2009.
53. Y. Wang, E.K. Teoh, "Object contour extraction using adaptive B-Snake model," *Journal of Mathematical Imaging and Vision*, vol. 24, no. 3, pp. 295-306, 2006.
54. Y. Wang, Z. Hou, X. Yang and K. Lim, "Adaptive B-Snake model using shape and appearance information for object segmentation," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 27, no. 5, pp. 633-649, 2011.
55. R.G.N. Meegama and J.C. Rajapakse, "NURBS snakes," *Image and Vision Computing*, vol. 21, no. 6, pp. 551-562, 2003.
56. Y. Wang, E.K. Teoh and D. Shen, "A B-Snake model using statistical and geometric information - applications to medical images," in *Proc. Seventh International Conference on Control, Automation, Robotics and Vision*, vol. 2, pp. 793-797, 2002.
57. P. Brigger, J. Hoeg and M. Unser, "B-Spline snakes: a flexible tool for parametric contour detection," *IEEE Transactions on Image Processing*, vol. 9, no. 9, pp. 1484-1496, 2000.
58. R. Crockett, T. Horvath and M. Yang, "Accurate heart model for pacemaker development," *Solid Freeform Fabrication Symposium*, University of Texas, Austin, TX, 2004.
59. N.E. Davisona, H. Eviatarc and R.L. Somorjai, "Snakes simplified," *Pattern Recognition*, vol. 33, no. 10, pp. 1651-1664, 2000.
60. M. Kass and A.P. Witkin and D. Terzopoulos, "Snakes: active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321-331, 1988.
61. L.D. Cohen, "On active contour models and balloons," *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol. 53, no. 2, pp. 211-218, 1991.

62. T. Mcinerney and D. Terzopoulos, "Medical Image Segmentation Using Topologically Adaptable Snakes," in Proc. First International Conference on Computer Vision, Virtual Reality and Robotics in Medicine, pp. 92-101, 1995.
63. Y. Wang, E.K. Teoh, and D. Shen, "Structure-adaptive B-snake for segmenting complex objects," in Proc. International Conference on Image Processing, vol. 2, pp. 769-772, 2001.
64. G. Hamarneh and T. Mcinerney, "Medial profiles for modeling deformation and statistical analysis of shape and their use in medical image segmentation," International Journal of Shape Modeling, vol. 10, no. 2, pp. 187-209, 2004.
65. W.P. Segars, "Development and application of the new dynamic NURBS-based cardiac-torso (NCAT) phantom," Ph.D Dissertation, Dept. Biomed. Eng., University of North Carolina, Chapel Hill, NC, 2001.
66. T. Malleprea and D. Bergers, "Accuracy of medical RP models", Rapid Prototyping Journal, vol. 15, no. 5, pp. 325-332, 2009.
67. C. Alberti, "Three-dimensional CT and structure models", British Journal of Radiology, vol. 53, no. 627, pp. 261-262, 1980.
68. J.R. Fram and E.S. Deutsch, "On the quantitative evaluation of edge detection schemes and their comparison with human performance," IEEE Transactions on Computers, vol. 24, no. 6, pp. 205-213, 1975.
69. I. Pitas, Digital Image Processing Algorithms and Applications, Wiley, New York, NY, 2000.
70. K.R. Castleman, Digital Image Processing, Prentice Hall, Upper Saddle River, NJ, 1996.
71. L. Roberts, Optical and electro-optical information processing, MIT press, 1965.
72. I.E. Sobel, "Camera models and machine perception," Ph.D Dissertation, Dept. Electr. Eng., Stanford University, Palo Alto, CA, 1970.
73. J.C. Muller, "Optimal point density and compaction rates for the representation of geographic lines," in Proc. Eighth International Symposium on Computer-Assisted Cartography, pp. 221-230, 1987.
74. L. Piegl and W. Tiller, "Least-squares NURBS curve approximation with arbitrary end derivatives," Engineering with Computers, vol. 16, pp. 109-116, 2000.

75. L. Piegl and W. Tiller, *The NURBS Book*, Springer-Verlag, New York, NY, 1997.
76. J. Hu, D. Yu and H. Yan, "A multiple point boundary smoothing algorithm," *Pattern Recognition Letters*, vol. 19, pp. 657-668, 1998.
77. D. Yu and H. Yan, "An efficient algorithm for smoothing linearization and detection of structural feature points of binary image contours," *Pattern Recognition*, vol. 30, pp. 57-69, 1998.
78. R. Goldenthal and M. Bercovier, "Spline curve approximation and design by optimal control over the knots," *Computing*, vol. 72, pp. 53-64, 2004.
79. G.E. Holzle, "Knot placement for piecewise polynomial approximation of curves," *Computer-Aided Design*, vol. 15, no. 5, pp. 295-296, 1983.
80. W. Li, S. Xu, G. Zhao and L. P. Goh, "Adaptive knot placement in B-spline curve approximation," *Computer-Aided Design*, vol. 37, pp. 791-797, 2005.
81. H. Park, K. Kim and S.C. Lee, "A method for approximate NURBS curve compatibility based on multiple curve refitting," *Computer-Aided Design*, vol. 32, no. 4, pp. 237-252, 2000.
82. H. Park, "An error bounded approximate method for representing planar curves in B-Splines," *Computer Aided Geometric Design*, vol. 21, no. 5, pp. 479-497, 2004.
83. H. Park and J.H. Lee, "B-spline curve fitting based on adaptive curve refinement using dominant points," *Computer-Aided Design*, vol. 39, no. 6, pp. 439-451, 2007.
84. D.F. Rogers and N.G. Fog, "Constrained B-Spline curve and surface fitting," *Computer-Aided Design*, vol. 21, no. 10, pp. 641-648, 1989.
85. T.I. Vassilev, "Fair interpolation and approximation of B-Splines by energy minimization and point insertion," *Computer-Aided Design*, vol. 28, no. 9, pp. 753-760, 1996.
86. W. Wang, H. Pottmann and Y. Liu, "Fitting B-Spline curves to point clouds by curvature-based squared distance minimization," *ACM Transaction on Graphics*, vol. 25, pp. 214-238, 2006.
87. L. Piegl and W. Tiller, "Algorithm for approximate NURBS skinning," *Computer-Aided Design*, vol. 28, no. 9, pp. 699-706, 1996.
88. H. Delingette, "Towards realistic soft tissue modeling in medical simulation," in *Proc. IEEE*, pp. 512-523, 1998.

89. L. Piegl and W. Tiller, "Surface skinning revisited," *Visual Computer*, vol. 18, pp. 273-283, 2001.
90. N. Amenta, M. Bern, and M. Kamvysselis, "A new Voronoi-based surface reconstruction algorithm," in *Proc. SIGGRAPH '98*, pp. 415-421, 1998.
91. N. Sapidis and P. Besl, "Direct construction of polynomial surfaces from dense range images through region growing," *ACM Transactions on Graphics*, vol. 14, no. 2, pp. 171-200, 1995.
92. S.T. Tuohy, T. Maekawaa, G. Shena and N.M. Patrikalakis, "Approximation of measured data with interval B-Splines," *Computer-Aided Design*, vol. 29, no. 11, pp. 791-799, 1997.
93. Z. Wu, S.H. Soon and F. Lin, "NURBS-based volume Modeling," in *Proc. International Workshop on Volume Graphics*, Swansea, pp. 321-330, 1999.
94. A. Watt, *Fundamentals of Three Dimensional Graphics*, Addison Wesley Publishers, New York, NY, 1989.
95. R.F. Riesenfeld, "Modeling with NURBS curves and surfaces," in *Fundamental Developments of Computer-Aided Geometric Modeling*, L. Piegl, Ed., Academic Press, New York, NY, pp. 77-97, 1993.
96. P.N. Chivate and A.G. Jablokow, "Review of surface representations and fitting for reverse engineering," *Computer Integrated Manufacturing Systems*, vol. 8, no. 3, pp. 193-204, 1995.
97. H. Park, "An approximate lofting approach for B-Spline surface fitting to functional surfaces," *International Journal of Advanced Manufacturing Technology*, vol. 18, no. 7, pp. 474-482, 2001.
98. D. Ma, F. Lin and C.K. Chua, "Rapid prototyping applications in medicine. Part 1: NURBS-based volume modeling," *International Journal of Advanced Manufacturing Technology*, vol. 18, pp. 103-117, 2001.
99. A. Suryanto, K. Herlambang and P. Rachmatullah, "Comparison of tumor density by CT scan based on histologic type in lung cancer patients," *Acta Med Indones*, vol. 37, no. 4, pp. 195-198, 2005.
100. M. Kinoshita, N. Hashimoto, T. Goto, N. Kagawa, H. Kishima, S. Izumoto, H. Tanaka, N. Fujita and T. Yoshimine, "Fractional anisotropy and tumor cell density of the tumor core show positive correlation in diffusion tensor magnetic resonance imaging of malignant brain tumors," *Neuroimage*, vol. 15, no. 43, pp. 29-35, 2008.

101. I. J. Schoenberg, "Contributions to the problem of approximation of equidistant data by analytic functions," *Quarterly of Applied Mathematics*, vol. 4, pp. 45-99 and 112-141, 1946.
102. M. Unser, "Splines: a perfect fit for signal and image processing," *IEEE Signal Processing Magazine*, vol. 16, pp. 22-38, 1999.
103. J.F. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679-698, 1986.
104. L.S. Davis, "A survey of edge detection techniques," *Computer Graphics and Image Processing*, vol. 4, pp. 248-270, 1975.
105. M. Sonka, V. Hlavac and R. Boyle, *Image Processing, Analysis, and Machine Vision*, CL-Engineering, 1998.
106. O.D. Trier, A.K. Jain and T. Taxt, "Feature extraction methods for character recognition-a survey," *Pattern Recognition*, vol. 29, pp. 641-662, 1996.
107. D. Ziou and S. Tabbone, "Edge detection techniques-an overview," *Pattern Recognition and Image Analysis*, vol. 8, pp. 537-559, 1998.
108. S.A. Gelman, T. Tardif, and T. Zrimec, "A knowledge-based approach to 3-D reconstruction of human cerebral vasculature," *Computers in Biology and Medicine*, vol. 28, pp. 405-413, 1998.
109. W. Sabra, M. Khouzam, A. Chanu and S. Martel, "Use of 3D potential field and an enhanced breadth-first search algorithms for the path planning of microdevices propelled in the cardiovascular system," in *Proc. IEEE Engineering in Medicine and Biology Society*, vol. 4, pp. 3916-3920, 2005.
110. O. Grove, L.A. Piegl, K. Rajab and S. K. Lai-Yuen, "From CT to NURBS: contour fitting with B-Spline curves," *Computer-Aided Design and Applications*, vol. 8 no. 1, pp. 3-21, 2011.
111. L.A. Piegl, K. Rajab, O. Grove and K.P. Valavanis, "Fault-tolerant computing in a knowledge-guided NURBS environment," *Computer-Aided Design and Applications*, vol. 6, no. 6, pp. 809-823, 2009.
112. L.A. Piegl, K. Rajab and O. Grove, "Curve interpolation with directional constraints for engineering design," *Engineering with Computers*, vol. 24, no. 1, pp. 79-85, 2008.
113. The Visible Human Project®, U.S. National Library of Medicine [Online], available: http://www.nlm.nih.gov/research/visible/visible_human.html.