MODELING AND PATH PLANNING OF CUSTOMIZED WOUND MATRIX FOR BIOPRINTING

by

Ezgi Küçükdeğer

Submitted to the Graduate School of Engineering and Natural Sciences

in partial fulfillment of

the requirements for the degree of

Master of Science

Sabancı University

Spring 2017

MODELING AND PATH PLANNING OF CUSTOMIZED WOUND MATRIX FOR BIOPRINTING

APPROVED BY:

Assoc. Prof. Dr. Bahattin Koç
(Thesis Supervisor)

Assoc. Prof. Dr. Selim Balcısoy

Prof. Dr. Gözde Ünal

DATE OF APPROVAL: 01/08/2017

MODELING AND PATH PLANNING OF CUSTOMIZED WOUND MATRIX FOR BIOPRINTING

Ezgi Küçükdeğer

Industrial Engineering, Master of Science Thesis, 2017

Thesis Supervisor: Assoc. Prof. Dr. Bahattin Koç

Abstract

Chronic wounds, including pressure and diabetic ulcers, constitute a serious and expensive health care problem as they have a considerable socio-economic impact worldwide. As it may take several months for the healing of chronic wounds, in order to improve wound healing, 3D bioprinting has emerged as an important tool in regenerative medicine to address the challenges of current treatment methods such as the risk and expense of surgery, donor site morbidity, infection and host rejection. Bioprinting fabricates customized biological constructs by depositing living cells, biomaterials and growth factors layer-by-layer based on a computer-aided-design (CAD). However, the data obtained by various imaging methods such as computed axial tomography (CAT), ultrasound, and magnetic resonance imaging (MRI) provides a series of two-dimensional cross-sections. The main problem is to reconstruct the 3D surfaces from two-dimensional contours.

In this study, a new algorithm is proposed to regenerate the wound surface from planar cross-sections and resulting CAD model can be used in wound measurement and patient-specific porous scaffold generation for 3D bioprinting. The proposed method generates smooth and non-twisted NURBS surfaces from a set of planar cross-sections satisfying the assumption of higher cross-sections involves the area bounded by lower cross-sections which is the case of wound geometries. Also, homogeneous porous scaffolds and three-dimensional bioprinting path planning are developed based on the resulting wound surface by using the developed algorithm.

The proposed algorithm provides more accurate surfaces for complex wound geometries, compared to two commonly used methods: curve lofting and swept surface generation. The developed algorithm approximates smooth and non-twisted surfaces on the cross-sections with sudden changes in shape, sharp edges and multiple branches.

3 BOYUTLU BİYO-BASIM İÇİN KİŞİYE ÖZEL YARA MODELLEMESİ VE BASIM YOLU HESAPLANMASI

Ezgi Küçükdeğer

Endüstri Mühendisliği, Yüksek Lisans Tezi, 2017

Tez Danışmanı: Doç. Dr. Bahattin Koç

Özet

Bası yaraları ve diyabetik ülserler de dahil olmak üzere kronik yaralar, psikolojik ve ekonomik açıdan zahmetli sağlık sorunları olduğu için dünya çapında ciddi bir sosyo-ekonomik etkiye sahiptir. Kronik yaraların iyileşmesi birkaç ay sürebileceğinden, yara iyileşimini hızlandırmak için birçok doku rejenerasyonu teknolojileri geliştirilmektedir. Üç boyutlu (3B) biyobasım teknolojisi, doku mühendisliğinde kullanılabilecek esnek bir araç ve geleneksel tedavi yöntemlerinin yan etkilerine çözüm sunabilecek bir alan olarak ortaya çıkmıştır. Bu yöntem kullanılarak canlı hücreler diğer biyo-malzemelerle, bilgisayar destekli tasarıma dayalı olarak katman-katman basılarak 3B doku yapıları oluşturulur. Bilgisayar destekli modeli oluşturmak için hastanin yarasının geometrisi, bilgisayarlı tomografi, ultrason ve manyetik rezonans (MR) gibi çeşitli görüntüleme teknikleri kullanılarak iki boyutlu enine kesitler halinde elde edilir.

Bu çalışmada, iki boyutlu düzlemsel konturlardan 3B yara yüzeyini oluşturmak için yeni bir algoritma geliştirilmiştir. Elde edilen 3B yara modeli, yara boyutu takibi ve 3B biyobasım için gözenekli doku iskelelerin geliştirilmesi için kullanılabilir. Bu yöntem kullanılarak, üst konturların alanlarının alt konturların alanları kapsadığı varsayımını sağlayan her düzlemsel eğri verisi üzerinde kıvrımı olmayan pürüzsüz 3B yüzey üretilebilir. Ayrıca, geliştirilen kod ile elde edilen yüzey kullanılarak, homojen gözenekli doku iskeleleri ve biyo-basım yol hesaplanması otomatik olarak üretilmiştir.

Geliştirilen algoritma ile, yaygın olarak kullanılan iki yöntem olan "iki veya daha fazla geometrik şekil arasında yüzey oluşturma" ve "yol kullanarak yüzel oluşturma" karşılaştırıldığında görsel olarak daha pürüzsüz ve uyumlu yüzeyler oluşturduğu görülmüştür. Geliştiren algoritma, seyrek kontur verileri üzerinde, farklı şekillere ve keskin kenarlara sahip verilerde ve farklı kollara ayrılan konturlarda da düzgün ve bükümlü olmayan yüzeyler oluşturur.

ACKNOWLEDGEMENTS

First, I would like to express my deepest gratitude to my thesis advisor, Dr. Bahattin Koç for introducing me to the exciting field of 3D printing and also for his guidance and encouragement throughout my studies including my bachelor's degree.

I would also like to thank my thesis committee members Dr. Selim Balcısoy and Dr. Gözde Ünal for their participation and time. I am very grateful to Dr. Çağla Aydın for her endless support and helping me with my academic and personal development with her friendly advice. My gratitude also goes to Dr. Kemal Kılıç, Dr. Barış Balcıoğlu and Dr. Tonguç Ünlüyurt for believing in me and supporting me to pursue an academic career. Also, I would like to extend my thanks to all the faculty, staff, and students I have been fortunate to encounter at the Sabancı University for their support and advice during my studies.

I would like to thank TUBİTAK 1003 project (Project number: 213M687) for their support and the Sabancı University Faculty of Engineering and Natural Sciences for financing my master's studies.

Special thanks also goes to the members of Koç research group for the lovely atmosphere at the office and nice times shared, especially to Navid Khani for his helps, inspiring ideas and fostering my motivation. I am heartily thankful to Emirhan Emir and Dr. Cenk Yanık for their constant friendship and helpful suggestions that guided me along the way and to all my friends with whom I shared all my joys and frustrations.

Finally, I owe my deepest gratitude to my mother for her unconditional love and unwavering belief in me, to my sister, Ayşe, and my cousin, Tuğçe, for their emotional support during my studies.

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 Motivation

Chronic wounds, including pressure and diabetic ulcers, constitute a serious and expensive health care problem as they have a considerable socio-economic impact worldwide. The diabetic ulcer is the main problem of the patients with diabetes, from which 422 million suffers based on the report published by World Health Organization in 2016. The number of diabetes patients quadrupled since 1980 and is projected to increase with the rising proportion of aged population [6]. As it may take several months for the healing of chronic wounds, it is necessary to measure wound healing in order to assess the success of a treatment and to predict the healing time. In clinical practices, the care staff usually evaluates the wounds visually and because of qualitative assessment, coordination in the staff and clinical studies become limited [7]. Lack of quantitative data has generated wide interest in development of methods for calculating ulcer geometry.

The most straightforward method to calculate the wound area is measurement of length and width by a ruler with an assumption of that the wound has a rectangular shape. Therefore, this method tends to overestimate the area 10 to 44% [8]. Wound tracing with a pen on a gridded transparent foil is another commonly used method where the perimeter is calculated by adding the areas of grids. This method suffers from an error rate of up to 30% and a possibility of ulcer contamination [9]. Another most common contact-less method is digital planimetry, where the wound is photographed by a digital camera and the area is calculated by a software based on the wound boundary drawn manually on the computer screen. However, manual tracing and digital planimetry have some disadvantages: tracing the wound margin is time consuming process and these methods have low inter-rater and intra-rater reliability [10, 11]. Compared to calculation of wound area and

perimeter only, measuring the volume provides more information on prediction of wound healing [12]. Many three-dimensional (3D) techniques for wound volume measurement have been proposed in recent years. The methods of stereovision, photogrammetry, laser scanning and structured light scanning provides reliable measurement of chronic wounds [13].

In medical area, the data generated by mentioned 3D techniques and various imaging methods such as computed axial tomography (CAT), ultrasound, and magnetic resonance imaging (MRI) provide a series of two-dimensional cross-sections through the object of study or point locations on the object boundary [14]. The main problem is to reconstruct the 3D surfaces from two-dimensional contours. Although many research studies have been carried out to propose efficient methods for surface reconstruction, most of them have been found to be insufficient when the contours differ in size and shape. Furthermore, many methods require dense data of cross-sections to perform well and cannot handle the case in which there are more than one contours on each slice. In addition, the triangular surface-based approach, which constitutes a considerable portion of the proposed methods, is inefficient in storage and computation because of large number of triangles on the surface that is obtained by stitching triangular patches between the contours and lost of topological information [15].

Surface reconstruction has been used for medical applications for measurement and tissue engineering purposes. Biofabrication of porous wound scaffolds has been extensively studied to improve wound healing process [3–5]. In order to generate a toolpath for scaffold based 3D bioprinting, Computer-Aided-Design (CAD) model is required. However, most of the wound imaging methods do not provide a CAD model of the wound volume.

This thesis proposes a new algorithm for smooth surface reconstruction from planar contours obtained by imaging of chronic wounds, allowing any number of contours in each slice and contours with dissimilar shapes. This algorithm provides a surface formed by NURBS surface patches automatically without user interaction during the process. The resulting surface of a wound can be used to assess the healing process or as a CAD model for porous scaffold generation to be used in patient-specific 3D bioprinting of functional wound matrix. In this thesis, the cross-sections are obtained by processing the digital images of chronic wound by using ImageJ software. Then, the CAD model of porous

scaffold and the tool path are generated by the developed code in VBScript language using Rhinoceros software.

1.2 Overview

The second chapter of this thesis serves as a brief review of some important concepts in tissue engineering for wound healing improvement and some important techniques that have been proposed for wound measurement, wound imaging and surface reconstruction from cross-section.

In the third chapter, the developed algorithms are given with explanations. Firstly, how 2D cross-sections are obtained to demonstrate the results of the developed algorithm is described. Then, a new methodology for surface reconstruction is proposed for three kinds of surfaces: skinned, branched and capped. Finally, an algorithm to achieve a porous scaffold design and toolpath planning based on a given surface is explained.

In the fourth chapter, developed algorithms are applied on several inputs. Input contours are generated from 2D images of two chronic wounds. The developed surface reconstruction algorithm and tool path generation script is applied to these cross-sectional data. Also, the surface generation algorithm is tested on several cross-sectional data and the resulting surfaces are compared with two commonly used and readily available methods in several CAD software: curve lofting and swept surface generation by using a baseline or a base-curve. Also, a methodology developed by [3] for toolpath planning of multi-material scaffolds is applied to the surface obtained by our algorithm and the results are given in this chapter.

The last chapter includes a short summary of the developed algorithms and the main results obtained in this thesis.

Chapter 2

LITERATURE REVIEW

2.1 Tissue Engineering in Wound Care

2.1.1 Cutaneous Wounds and Wound Healing Process

Skin, which has two main layers: the epidermis and dermis, and a layer of subcutaneous fat, is the largest organ in human body. Together these layers support survival by protecting the human body from damages caused by outside effects. This way, the skin protects against water loss, and damages caused by toxins, microorganisms, chemicals in the environment and sunlight. Also, it helps to maintain body temperature and functions as a sensor of outside changes. Furthermore, it synthesizes vitamin D which is necessary for bone growth and bone remodeling by osteoblasts and osteoclasts; controlling the amount of calcium and phosphate in the body [16, 17]. Due to injuries and illnesses, skin integrity can be lost resulting in imbalance in skin physiology and loss of crucial skin properties. If skin cannot function properly, it may cause disabilities or even death [18].

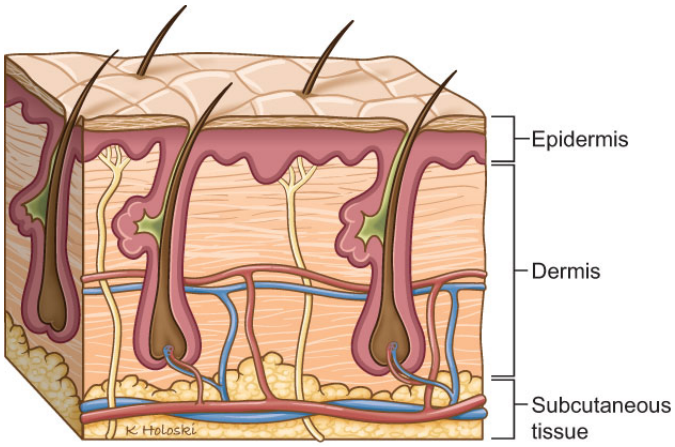

Figure 2.1: The layers of human skin [1]

There are two main causes for significant skin loss: thermal injury and chronic ulceration [19]. Approximately one million people visit hospital emergency because of thermal injury per year in the US and about 11 million thermal injuries per year worldwide out of which approximately 265,000 leading to death [20, 21]. Chronic wounds, including pressure and diabetic ulcers, constitute enervating and expensive health care problem as they have a considerable socio-economic impact worldwide. The diabetic ulcer is the main problem of the patients with diabetes, from which 422 million suffers based on the report published by World Health Organization in 2016. The number of diabetes patients quadrupled since 1980 and is projected to increase with the rising proportion of aged population [6, 22]. Only in the US, out of approximately 21 million who suffer from diabetes, 2 million have chronic diabetic ulcers leading to amputation in roughly 4% of the patients with chronic ulcers [23]. According to Supp and Boyce [24], the total treatment cost of chronic ulcers is estimated as $8 billion annually only in the US.

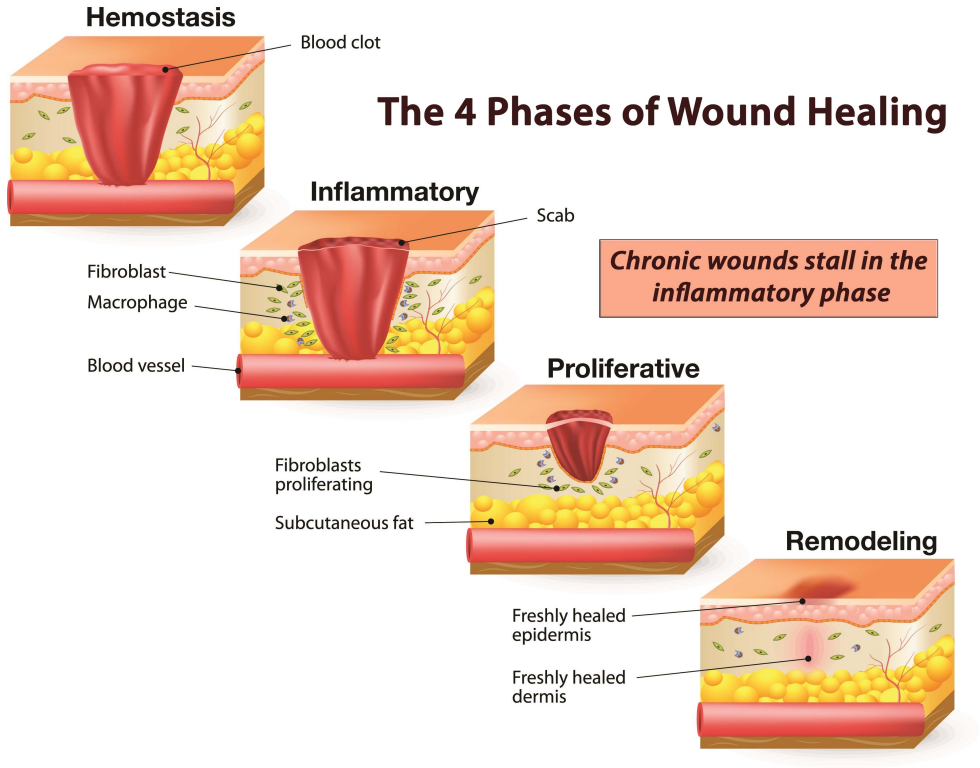

Figure 2.2: The 4 phases of wound healing [2]

The complex biology of wound healing has been a popular research area over the past three decades. Although wound healing seems like a familiar and everyday process, underlying biology is consisted of complex and orderly events [25]. Any disruption in this

process may result in a chronic wound that may fail to heal. Serious skin loss damages epidermis which is keratinized stratified outer layer and may even penetrates through dermis which is collagen-rich inner layer. In deep wound healing, epidermal cell migration and proliferation occur as well as wound contraction and matrix remodeling in the dermis. Wound healing process may be triggered in the cells at the wound margin by "free-edge effect" which means the absence of neighboring cells [19]. With the increased knowledge on wound healing process, innovations in wound care that have provided rapid closure of chronic wounds has increased.

2.1.2 Tissue Engineering for Wound Healing

Tissue engineering is an interdisciplinary field that integrates many areas of science and engineering aiming to regenerate new tissue replacement for diseased or damaged tissue. Scientists from different fields such as material science, molecular and cell biology, biomedical engineering, mechanical engineering, chemical engineering and many more work together to design, develop and test the tissue engineered substitutes. In order to develop a new biological material for tissue replacement, a source of cells is required as well as an artificial extracellular matrix (ECM) which stimulates cells to migrate, proliferate and eventually differentiate.

In order to improve healing in chronic wounds and speed up the wound healing process, composite synthetic and biological dressings are commonly used [26–28]. However, these wound dressing fails to provide permanent treatment. Therefore, further surgical operations for application of a split-thickness autograft or allograft are required, bringing some undesired results such as the risk and expense of surgery, donor site morbidity, and rejection of the transplanted tissue [18]. In order to solve these problems, autologous and allogenic epidermal sheets are proposed to be used to improve epidermal healing. The epidermal grafts are obtained from a small skin biopsy by growing epidermal cells in large quantities [29, 30]. Although autologous graft avoids immune-rejection, there are disadvantages such as long and expensive process of autologous graft development. A patient need to wait 2-3 weeks for obtaining sufficient amount of keratinocytes after epidermal harvest. This process costs approximately $13,000 per 1% of total body surface [31]. As a result, autologous grafts are not available for large or severe wounds [32, 33].

Another approach to solve host rejection against allografts is to decellularize the al-

lografts to remove immunogenic cellular elements. This method has been successfully used alone or with cultured autologous keratinocytes to improve wound healing [34]. At present, there are a number of different engineered skin substitutes, such as acellular implants including fibroblasts and acellular collagen matrices, available for clinical use, all of which fail to provide fully functional skin regeneration [35]. There are several disadvantages that limit their use; the epidermal grafts are difficult to handle because of their fragile stucture, the body may reject to allografts or xenografts, and skin substitutes may carry infectious agents [18].

Further improvements are needed to develop tissue-engineered skin constructs which are less complex, more cost effective, easy to use and avoid immune rejection leading to a new tissue having a similar structural and functional properties with the healthy tissue.

2.2 3D Bioprinting for Wound Healing

2.2.1 3D Bioprinting

Three-dimensional (3D) bioprinting, is a medical procedure in regenerative medicine and aims to construct artificial tissues or organs that show similar biochemical properties and functions of their natural counterparts by printing cells and biomaterials based on the principles of additive manufacturing. Similar to the deposition based additive manufacturing techniques, biologically relevant materials are deposited layer-by-layer in a desired pattern with the help of high-precision bioprinting equipments [36, 37]. In contrast to the traditional biofabrication methods which allow random distribution of biomaterial and cells within the fabricated construct, such as molding, electrospinning and solvent evaporation, spatial control of the materials and cells is possible during deposition process by the 3D bioprinting techniques [38, 39]. In addition, 3D bioprinting requires different materials and deposition methods than 3D printing of non-biological materials. The biomaterials that can be used in bioprinting has been extensively studied in recent years [40]. Hydrogel is one of the commonly used biomaterial for the generation of soft tissues such as skin because hydrogels have similar structural properties with the extracellular matrix of many tissues and they encapsulate cells in the process of 3D bioprinting [41, 42]. As for the hard tissue such as bone, polymers and ceramics have been commonly used materials [43].

In order to print a biological construct, first of all a CAD model of the desired tissue should be designed or achieved by using several imaging techniques such as computed tomography (CT) and magnetic resonance imaging (MRI). Based on the 3D computer model of tissue, the printing tool path is planned based on the computer model [37]. The second step is material preparation and cell selection based on the desired functions of the final product. The selected material should be biocompatible and its degredation by-products should be non-toxic to the cells and the surrounding tissue. Also, the selected material should have supportive mechanical and biochemical properties to provide full integration for the cells [44, 45]. The next step is to select an appropriate bioprinting method. The most common techniques are inkjet bioprinting and extrusion based 3D bioprinting. The inkjet bioprinting is a process of discrete delivery of biomaterial on specified locations. On the other hand, extrusion based bioprinting is 3D continuous process of material extrusion from a capillary or a syringe by following the tool path plan. 3D bioprinting is more advantegous than inkjet bioprinting for generating high cell densities and large biological constructs [46].

2.2.2 Tissue Scaffold Requirements for Bioprinting

The scaffold-based approach is the most commonly used method for extrusion based bioprinting. Layers of cell-laden hydrogel are printed to form a scaffold which the skeleton of the desired tissue structure providing structural support to the cells. A scaffold is designed to mimic the natural complex shape of both external and internal architecture of the tissue and it should provide the optimal environment for cell attachment, proliferation, regeneration and growth [47, 48]. If the scaffold structure enhances the cell growth by cell migration, proliferation, waste removal and such dynamic regulatory systems, the interaction between the cells and the scaffold can be considered as successful interaction [49].

Figure 2.3: An example of a wound scaffold including cells, growth factors and proteins [3]

Determining porosity is one of the important factors that affect the functionality of the scaffold design as well as the selected material. Porosity directly affects the nutrient and oxygen flow which is very critical for cell survival [50]. In traditional scaffolds with homogeneous porosity, the cells that are away from the boundary might have not access to the nutrients and oxygen because of limited nutrient flow. As a result, the size, geometry, interconnectivity and surface chemistry are important parameters that affect the access of cells to the necessary materials. Since tissues do not have homogeneous properties, the pore size should vary spatially based the required properties of the intended tissue [51]. As porosity increases, more cell growth and regeneration are supported. On the other hand, low porosity increases the integrity and the material capacity of scaffolds. Therefore, an optimum porosity should be used while designing the scaffolds. In the studies, wide range of pore size ranging between 50-400µm is found as acceptable, but there is no agreement on the optimal pore size [52].

Another important property of scaffolds is to ensure interconnectivity of pores, so the survival materials and waste will be able flow between cells. Well-connected internal network channels enable oxygen and nutrition to reach the cells in the deeper sides of the scaffold [53]. As for manufacturing process, the path planning is an important step to ensure interconnected scaffold architecture. The path plan should be designed in a way that the deviation between the design and the actual fabricated structure is minimized and the number of discontinuities is minimized to decrease the build time by avoiding frequent starts and stops while material deposition[54].

2.2.3 Bioprinting of Human Skin

Bioprinting is an advanced tissue engineering technique that fabricates customized biological constructs by depositing living cells, biomaterials and growth factors layer-by-layer based on a computer-aided-design (CAD). According to Ng et al. [4], there two main strategies that have been utilized in skin bioprinting for wound healing. Laser-based bioprinting is one of the two main techniques. A recent in vitro study, 20 layers of fibroblasts and keratinocytes were successfully fabricated by using laser-based approach. The resulted artificial skin constructs showed similar properties with the native tissue after 10 days of cultivation [55]. In another study, a layer-based approach was utilized to print skin construct in the dorsal skin fold chamber of nude mice and its in vivo transplantation resulted in good attachment to the surrounding tissue and ingrowth of blood vessels [56].

Another approach is deposition-based bioprinting where the cell embedded biomaterial is deposited through a nozzle by following a predetermined toolpath. In the study of Binder et al. [57], fibrin-collagen layers were printed using a deposition-based bioprinting on fullthickness wound of nude mice by using amniotic fluid derived stem cells or human fibroblasts (hFBs) and keratinocytes (hKCs). Although the results of in situ printing were promising, a further development is needed to apply on human patients. Also in a recent study of Cubo et al. [58], a human plasma derived bilayered skin is fabricated by using fibroblasts and keratinocytes that are obtained by skin biopsies. The results of the analysis of in-vitro cultures and in-vivo applications on mice, suggested that the printed skin resembles normal human skin in terms of structural and functional characteristics and appearance.

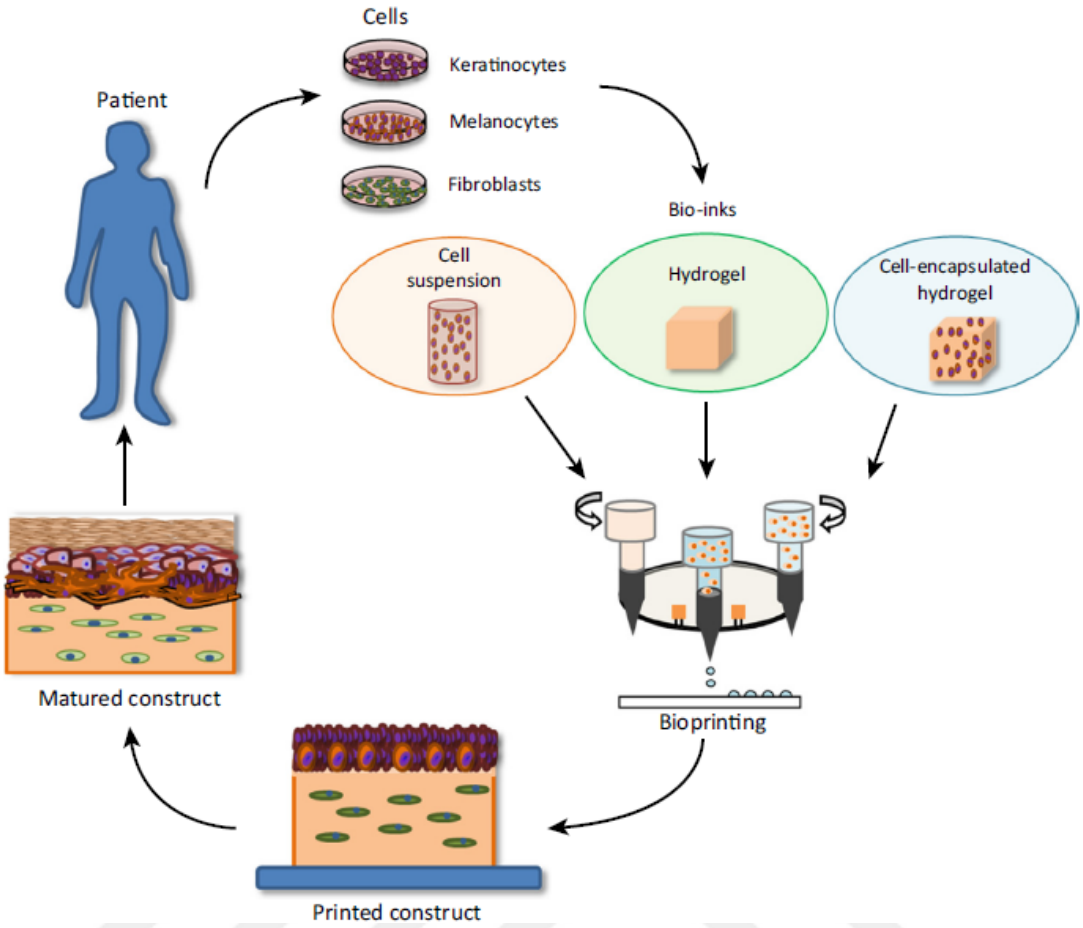

Figure 2.4: The steps of bioprinting technology [4]

An example of skin bioprinting includes the steps shown in Figure 2.4. Firstly, a sufficient amount of cells are obtained by culturing a skin biopsy collected from the patient. Then, bio-inks are prepared by using these cells in bioprinting of customized skin constructs. Finally, the printed skin constructs are matured under submerged condition to become convenient for transplantation [4].

2.3 Wound Imaging and Measurement

Wound measurement is a crucial step to assess the effects of a treatment. It is necessary to develop quick, affordable, repeatable and accurate method for calculating the wound geometry. However, it is hard to measure the geometry of the wounds with varying sizes and shapes [12]. Most of the developed methods for wound measurement are costly and require specially trained medical personnel to perform it. Therefore, in order to measure the wound quickly and easily, the wound is measure either by qualitative visual evaluation

or painful methods such as filling wound void by saline liquid or alginate paste in clinical practices [7].

The three most common methods to measure a wound are linear measurement, wound tracing and digital planimetry. Firstly, linear measurement is the most straightforward method assuming the wound has a rectangular shape. The area of the wound is calculated by multiplying the lengths of the vertical and horizontal lines of the bounding box around the wound bed [59]. Ruler-based technique has a good reliability for the wounds with small areas. As the wound area increases, the reliability of this method decreases [11]. Furthermore, ruler-based assessment usually overestimate the area of the wound between 10-44% [7]. The second commonly used method is manually tracing wound edges on a gridded transparent material by using a pen. The wound area is calculated either by summing the areas of each grid inside the drawn wound edge or weighing the cutout tracings. The method is relatively fast; however it can be time consuming for the large wounds [7]. Also, the accuracy of this method is limited as the measured area may differ from the real with an error of 25%, since it is hard to estimate the area of the grids on the border. Furthermore, manual tracing fails to provide inter-rater and intra-rater reliability because of inaccurate drawings and varying thickness of the lines drawn. Different observers measure the same wound giving different results, and the observers could fail to provide exactly same tracing on the same wound giving different result each time they measure [60]. Another problem with this method is the possibility of ulcer contamination, since the transparent material touches to the wound bed [12]. The third method is digital planimetry; a digital image of the wound is captured by a camera and the boundaries are manually drawn on a computer screen. The wound area is calculated by a software. However, there are some drawbacks of this method since it is hard to photograph a wound twice from exactly same distance and manual tracing fails inter-rater and intra-rater reliability [10].

Measuring the area of the wound is a common way to assess the healing process; however, changes in the volume of the wound provides additional necessary information for wound assessment. Compared to area measurement techniques, less attention has been given to volume measurement in the literature since it is more complicated than area calculation and requires three-dimensional measurement. The most straightforward method is to measure the maximum depth and to estimate the volume by multiplying

measure depth by maximum length and maximum width obtained by linear measurement, but this method is inaccurate and unreliable as it overestimated the volume [12].

The most commonly used reliable methods for 3D measurement are stereophotogrammetry, laser or white light triangulation and depth sensing cameras [7]. In stereophotogrammetry, images of a wound from different angles are taken by two cameras to reconstruct a 3D image for wound volume estimation. However, this method is time consuming, expensive and required special training. Therefore, it is not appropriate for routine clinical use [11]. In structured light triangulation, parallel stripes of light are projected onto the wound in order to create 3D point cloud of the object. In the study of Jones and Plassmann [61], user interaction was needed to define wound boundaries manually for wound area calculation. Also, this method overestimated the area and the volume, since the volume is calculated by multiplying the area with an average depth obtained from the depth map. In laser line triangulation technique, the laser line moves over the object surface and the scanner collects a series of profiles to be merged into a complete 3D model of the object. Similar to the structured light triangulation method, laser triangulation requires manual tracing of wound boundaries [7]. Also, depth sensing cameras can be used to obtain geometric information of wounds. One type of depth sensing cameras is RGB-D cameras which provide depth information of each pixel on the captured RGB (Red-Green-Blue) images. RGB-D cameras approximate depth information on RGB pixels by using either structured light patterns combined with stereo sensing or time-of-flight laser sensing [62].

In order to overcome both subjective evaluations and invasive applications of traditional methods, recently various optical imaging techniques have been developed for wound assessment. These optimal imaging techniques not only provide area and volume estimation of the wound but also gives detailed information regarding tissue characteristics which cannot be evaluated by visual inspection. Blood flow, collagen information, hemoglobin content, temperature, water content and such characteristics of the wound can be obtained by using optical imaging methods such as thermal imaging, laser Doppler imaging, microscopy and near infrared imaging [63, 64]. In addition to optical imaging techniques, a computed tomography (CT) scan and magnetic resonance imaging (MRI) are used to assess wounds. However, these technologies are costly and require special training [64].

Most of the wound imaging techniques need post-processing for reconstruction of

wound surface in order to provide a CAD model of the wound which is required for 3D bioprinting applications.

2.4 Surface Reconstruction on Cross-sections

The purpose of surface reconstruction is to create a surface model from a set of points or cross-sections that lie on or near an unknown surface [65]. In many medical studies, it is common to obtain 2D cross-sections by Computer Assisted Tomography (CAT), Nuclear Magnetic Resonance Imaging (NMR) and some ultrasound techniques. These techniques slice biological specimens into thin layers and medical doctors uses the serial sections to understand 3D shape of the object. Also, in geographical information systems such as terrain reconstruction, 3D terrain surfaces need be approximated from planar cross-sections of terrain at different heights.

Image-based modeling is a commonly used method for geometric surfaces or for terrain and city modeling. Image-based methods generate 3D object from 2D images through mathematical models or approximation methods such as shape from shading, shape from texture, shape from specularity and shape from contour methods [65].

The problem of 3D surface reconstruction from 2D cross-sections has received a good deal of attention [14, 66]. According to Meyers et al. [14], the surface regeneration problem can be divided into four main subproblems.

The first one is the correspondence problem which arises when there are more than one contour on a cross-section. If there are multiple contours on the same layer, the topological relationship and linking of the contours should be determined. The most common method is to examine overlap between the contours on adjacent layers, but this method fails to generate correct solutions when the data is sparse. However, it is difficult to generate an automatic solution for all kinds of cross-sections [14].

The second problem is the tiling problem which is to link the points on pairs of contours from adjacent layers. The most common way is constructing a triangular mesh by linking the points [67, 68]. First, Keppel [69] suggested an approach to solve the problem of matching points on successive cross-sections. In order to find the optimum matching, several objective functions have been used in the literature. Keppel [69] suggested to use "Maximize Volume", Fuchs et al. [70] used "Minimize Area" as objective function, Cook

et al. [71] tried to match the directions of the points from the mass center of the contours they lie, and more. These triangulation approaches generate triangular meshes resulting in large memory requirement. Also, these methods fail to produce correct results when the data is sparse, there is noise and the contours have different shapes and orientations.

The branching problem is the third problem which arises when a pair of adjacent cross-sections have different number of contours. Multiple branching is a problem that is required to be solved to approximate a surface on complex object. Christiansen and Sederberg [72] proposed an algorithm that deals with two-branched surfaces by adding a connecting point between the pairs of contours but requires user interaction for complex shaped objects. Meyers et al. [14] proposed a new algorithm for composite contour construction and generation of a canyon which is a region between two contours on a plane, merging the contours by extending a portion of their perimeters. In the last decade, several approaches for generating tangent continuous (G^1 continuous) branched surfaces have been proposed and applied successfully [73, 74].

Finally, the surface fitting problem is a problem of fitting the best surface to the triangular mesh that is found by skinning and branching algorithms. The surface that fit to the mesh provides detailed information on the geometry of the object. The mesh is filled by triangular or rectangular surface patches [75]. Several methods have been proposed to fill each triangular facet with one or more triangular patches with G^1 continuity by using the connectivity information of the triangular mesh [76–78]. In addition, skinning techniques have been suggested for rectangular surface interpolation by blending a group of adjacent contours [79–81].

Chapter 3

METHODOLOGY

3.1 Imaging and Contour Generation

In order to model a patient-specific wound geometry for bioprinting, firstly reverse engineering is applied to obtain a CAD model of the specific wound. The cross-sectional data of the wound can be generated by scanning the wound with CT or MRI scanner or by image based techniques such as stereophotogrammetry and depth sensing cameras. Resulting data consists of 2D images of the cross-sections of the object with certain intervals. These images can be converted into cross-sectional curves indicating the wound geometry by using segmentation methods. If a mesh model is available, cross-sectional curves can be obtained by slicing the mesh in certain intervals. Also, cross-sections can be obtained by curve fitting to a point cloud. In this thesis, 2D cross-sections are obtained by processing 2D medical images of wounds, but the developed surface reconstruction algorithm can be applied to any 2D cross-sectional data.

A point is generated representing each pixel of the wound image. The point's x and y coordinates are taken as the vertical and horizontal coordinates of the pixel. To give the third dimension to the point, the color intensity of each pixel is used as z axis. To do so, the first step is to convert the medical image into 8-bit image. In the 8-bit image, the color intensity is used as an indicator of the distance from the camera since the surface that is closer to the lighting source reflects more light to the camera resulting in lighter colors and vice versa. The lighter points are assumed to represent higher z-values, being closer to the camera and the darker points are assumed to have lower z-values, being further from the camera. In the 8-bit images, 255 is the value of white and 0 corresponds to black. Therefore, the intensity values are scaled based the deepest point's distance to the skin surface provided by an expert. For example, if the expert says the deepest point of the

wound is 8 mm lower then the skin surface, then all intensity values should be multiplied by $8/255$ to find the accurate locations of the points. By using the resulting point cloud, curve fitting algorithms can be applied to achieve cross-sectional curves.

In order to achieve better results from 2D images, medical images of relatively clean and red wounds should be used and lighting conditions should be optimized. For the treatment of secreting yellow wounds or sloughy wounds, first of all the wound should be cleaned by the application of encymes and treated surgically and with 0.9% sodium chloride solution. After the cleansing phase, the imaging and treatment can be applied [61]. Also, lighting conditions should be optimized to avoid shadows and camera flash reflections, and to increase image contrast. It is recommended to use a circular polarized filter while imaging the wound to avoid erroneous assessments.

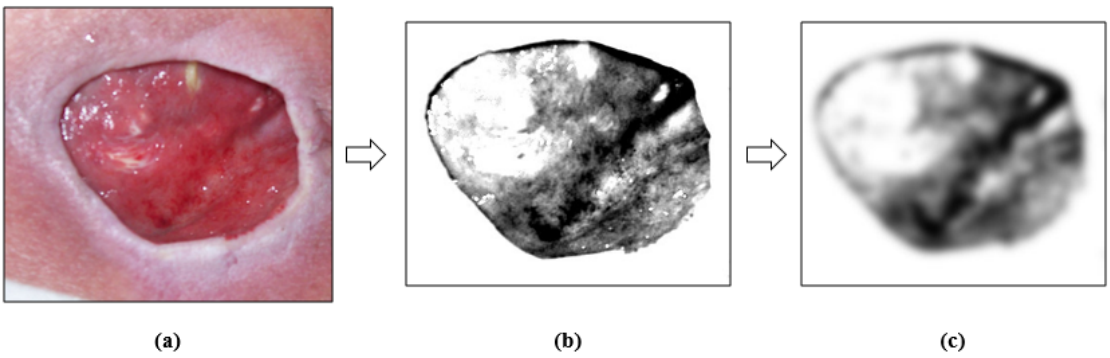

Figure 3.1: Image preprocessing **(a)** the medical image **(b)** converted into 8-bit image **(c)** blurred image

This methodology is applied to an image, Figure 3.1(a), by using a software called ImageJ. Firstly, the image is converted into 8-bit image and contrast of the image is adjusted to differentiate the deepest point of the wound and the points closer to the camera, as in Figure 3.1(b). Then, gaussian blurring is applied to the image to avoid sudden changes in intensity values, as shown in Figure 3.1(c).

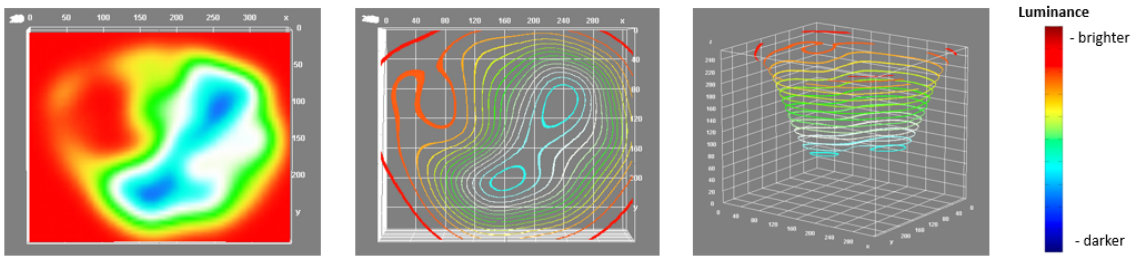

Figure 3.2: Intensity values of pixels

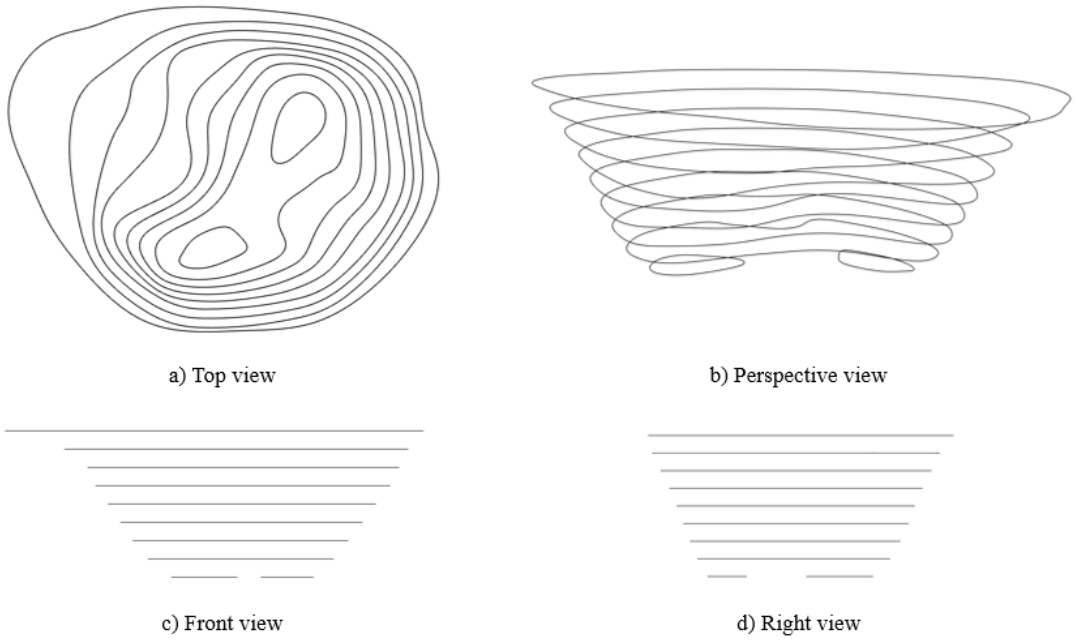

Figure 3.3: The resulting set of cross-sections

Figure 3.2 shows the intensity values of each pixels with different colors. This thermal map and thermal isolines that indicate the pixels that have the same intensity value are generated by ImageJ. Resulting contours' validity can be checked by the similarity with the thermal isolines.

Once the 8-bit blurred image is obtained, a point representing each pixel is drawn in a CAD software, Rhinoceros 5.0. After point cloud generation, cross-sectional curves are generated by curve fitting as shown in Figure 3.3. A sparse set of cross-sections is more preferable than a dense set, since the surface reconstruction algorithm may generate smoother surfaces with a sparse data.

Generating the cross-sectional data however is not the main scope of this thesis. Thus, the reader is referred to 3D scanners and depth sensing image based techniques for highly accurate and computationally efficient methods to obtain cross-sectional data of the wound.

3.2 Surface Reconstruction from Contours

A new methodology is proposed to reconstruct an accurate and non-twisting smooth 3D surface based on the planar contours obtained by cross-sectional data of a specific wound. Since most of the wounds has a concave shape, it is assumed that on successive planes, the

curve on the upper plane contains the curves on the lower plane. In other words, when the curves are projected onto the same plane, the projected curves do not intersect and they do not contain separate regions. This assumption is valid since chronic ulcer beds have a concave shape. Thus, the algorithm deals with three types of links between contours in successive planes: one-to-one link, one-to-many link and capping, as shown in Figure 3.4. Many-to-many link is not considered because the assumption do not allow this kind of cross-sections.

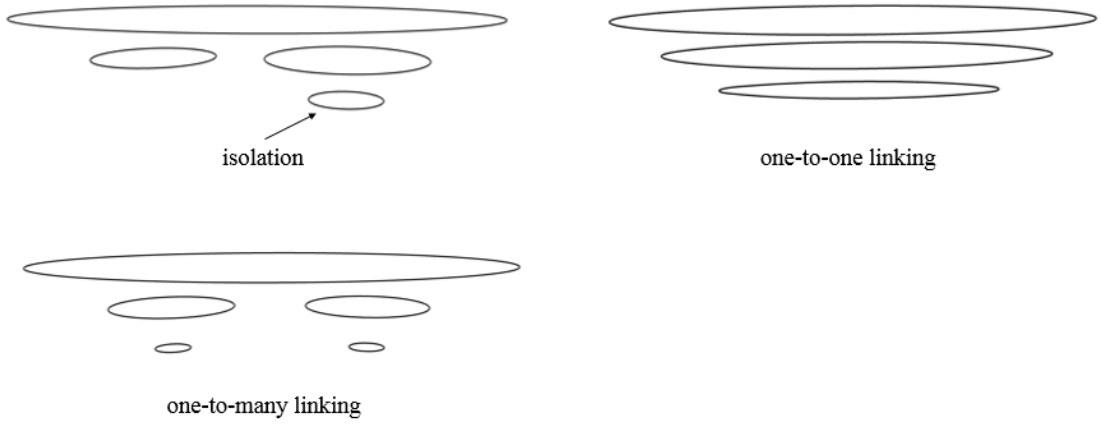

Figure 3.4: Three kinds of contour linking

The algorithm generates a smooth NURBS surface through the given cross-sections in 4 steps which are region detection, surface reconstruction in skinned regions, surface approximation in branched and capped regions and joining the surfaces. First, the curves are grouped into three categories based on their linking type by region detection algorithm. Then, a surface is generated for each group of curves in skinned, branched and capped regions. In the last step, all surfaces are joined and the final smooth surface is constructed.

The artificial contours, shown in Figure 3.5, will be used to explain each step of the proposed algorithm.

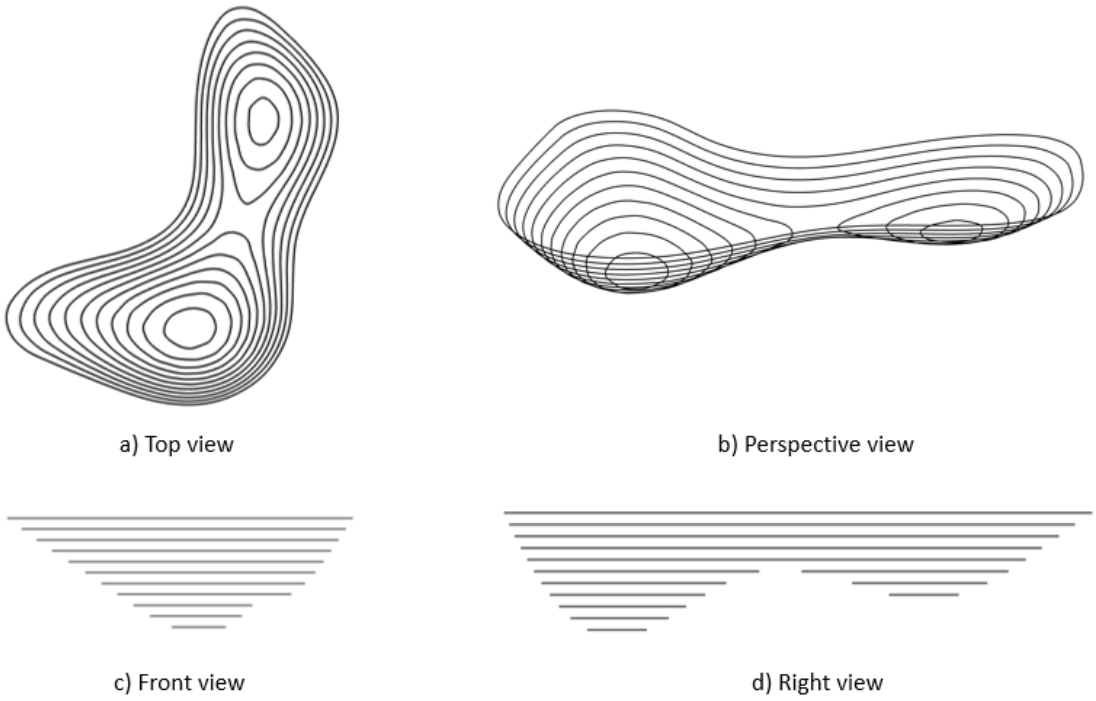

Figure 3.5: The sample cross-sections

3.2.1 Region Detection

In the first step of the algorithm, the obtained cross-sections are grouped into skinned, branched and capped regions based on their linking types. A skinned region denotes a list of successive planar contours that form a surface by one-to-one linking. A branched region is a list of contours, in which one contour lies on a plane and the others lie on the next plane and the contours require one-to-many linking. A capped region is defined by a contour in isolation, in other words a contour which does not require any further linking [15].

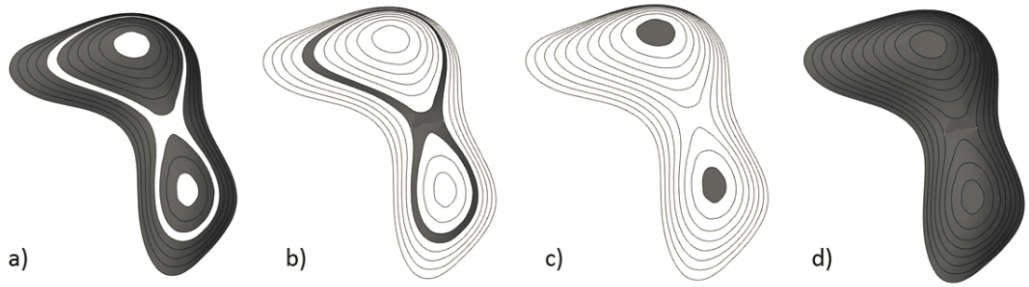

Figure 3.6: Surface generated through contours **a)** skinned surfaces **b)** branched surfaces **c)** capped surfaces **d)** resulting surface

To detect the regions where the related surface is created, identification of the contours that correspond to each other is important. This correspondence problem arises when there are multiple contours in a cross section. In this case, the contours on the adjacent cross sections should be grouped based on their topological links. The correspondence problem has been investigated by many researchers since it is difficult to solve the problem in its general form especially when the spacing between cross sections is large. Many current solutions assume that the data is dense enough to examine overlap between the contours [14]. Similarly, because of the containment assumption in this study, the correspondence problem is solved with the following algorithm.

The contour list, *CList*, is taken as input and the region detection algorithm generates three output arrays; *SkinList* which is a list of the contours in skinned regions, *CapList* in capped regions, *BrList* in branched regions. *SkinList* and *BrList* are nested arrays, which include lists of contours. The pseudo code of region detection is given in Algorithm 1.

Algorithm 1 Region Detection

INPUT: $C_i \in$ ***CList***: A list of planar curves

OUTPUT: ***SkinList*(k)** $\in$ ***SkinList***: A list of curves in skinned regions

CapList: A list of curves in capped regions

BrList*(k)** $\in$ ***BrList: A list of curves in branched regions

1: **procedure** REGIONDETECTION(***CList***)
2: Sort curves in ***CList*** based on the plane that they lie from the highest z value to the lowest; /* using any sorting algorithm */
3: Create a new array *ZGroups*; /* Size of the array = the number of cross sections */
4: **for each** C_i in ***CList*** **do** /* group the contours based the z value of the plane they lie */
5: Add C_i to the *ZGroups(j)*; /* j^{th} element corresponds to the plane that C_i lies */
6: **end for**
7: **for each** element of *ZGroups* **do**
8: **for each** C_i in the element *ZGroups(j)* **do**
9: **if** j is the last index of *ZGroups* **then**
10: ***CapList*** $\leftarrow$ ***CapList*** $\cup$ C_i;
11: **else if** j is the first index **then**

12: Create an element ***SkinList(k)***;
13: Add C_i to the ***SkinList(k)***;
14: **else**
15: **for each** C_m in the next element of *ZGroups(j+1)* **do**
16: Project C_m onto the plane of C_i;
17: **if** C_i contains the projected C_m **then**
18: *Candidates* $\leftarrow$ *Candidates* $\cup$ C_m;
19: **end if**
20: **end for**
21: **if** there is only one curve, C_n, in the *Candidates* **then**
22: Add C_n to the ***SkinList(p)*** where $C_i \in$ ***SkinList(p)***;
23: **else** /* if C_i contains more than one curve, then it indicates a branched region */
24: Create an element ***BrList(l)***;
25: Add C_i to ***BrList(l)***;
26: **for each** C_n in*Candidates* **do**
27: ***BrList(l)*** $\leftarrow$ ***BrList(l)*** $\cup$ C_n;
28: Create an element ***SkinList(k)*** and add C_n to the ***SkinList(k)***;
29: **end for**
30: **end if**
31: **end if**
32: **end for**
33: **end for**
34: **for each** last element C_i in all ***SkinList*** elements **do**
35: **if** C_i is not the first element of ***BrList(j)*** for all j **then**
36: ***CapList*** $\leftarrow$ ***CapList*** $\cup$ C_i;
37: **end if**
38: **end for**
39: **return** ***CapList***, ***SkinList***, ***BrList***;
40: **end procedure**

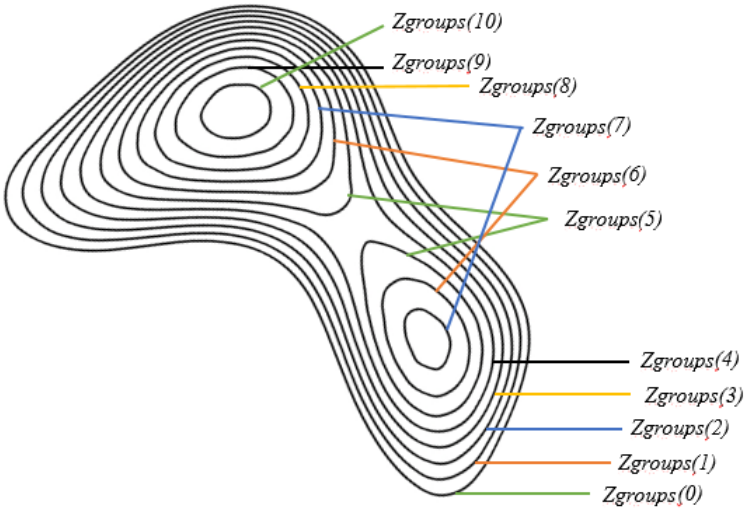

Figure 3.7: Contour assignments to each element of *ZGroups* array

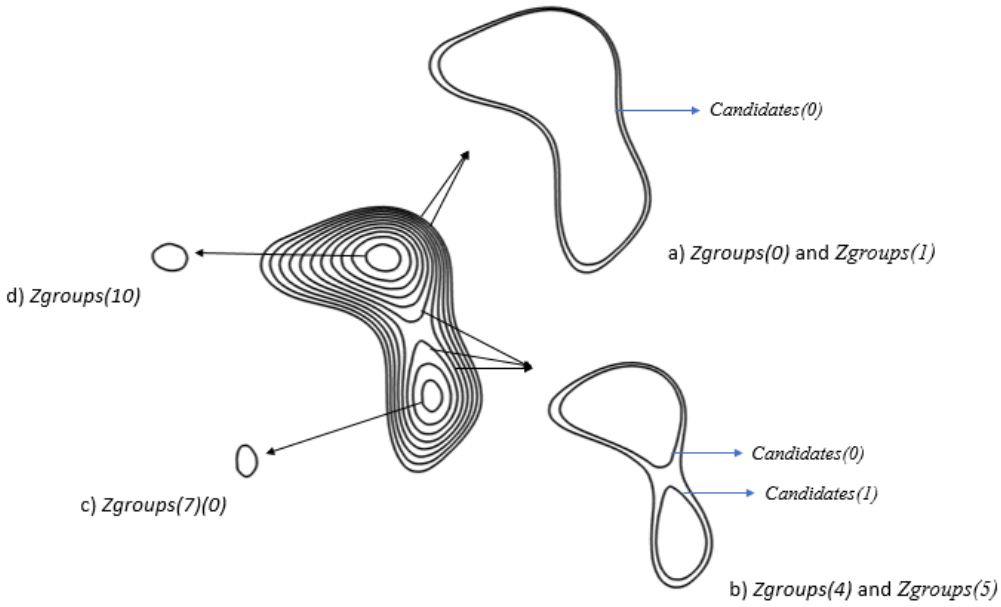

Figure 3.8: 4 different cases encountered during region detection algorithm **(a)** only one element in *Candidates* list **(b)** more than one element in *Candidates* list **(c)** no element in *Candidates* list **(d)** the last element of *ZGroup* array

First, two empty arrays, which are *ZGroups* and *Candidates*, are defined in addition to output arrays. *ZGroups* is a nested array whose elements include a list of curves that lie on the same z plane. The algorithm starts by sorting the planar curves based on the z plane they lie from the highest to the lowest. In this algorithm, Bubble Sorting has been

used for this purpose. In the first for loop, the *ZGroups* array is achieved by assigning each curve that lie on the same layer to the same element of the array, as shown in Figure 3.7. After that, each cross section is investigated to determine the curves in skinned and branched regions. In the first iteration of the third for loop, a new element in *SkinList* is created for each contour on the highest cross section. In the following iterations of the for loop, each curve on the next cross section is assigned to the corresponding *SkinList* element by the help of the *Candidates* list. To obtain the *Candidates* list, each curve, C_i, on the current plane is projected to the next plane, resulting in C_i'. Then, a list of curves that lie on the next plane and are contained by the projected curve, C_i', are assigned as the elements of the *Candidates* list. If the *Candidates* list has only one curve, it indicates that this curve belongs to a skinned region, as in Figure 3.8(a). In the case of multiple elements in the list, it implies that the projected curve and the curves in the *Candidates* list create a new branched region, as in Figure 3.8(b). Also, a new element in *SkinList* is created for each contour in the *Candidates* list if the contours are not in the last element of *ZGroups*, which means they do not lie on the last cross section. In the last iteration of the for loop, all contours in the last element of *ZGroups*, which are the curves in isolation, are added to the *CapList*, as shown in Figure 3.8(d). The *CapList* is completed in the sixth for loop by adding the curves, each of which is the last element of a *SkinList* element and does not lie on the upper plane of a branched region, as in Figure 3.8(c).

3.2.2 Skinning

In this step, the serial contours in each skinned region is approximated to a smooth and non-twisted NURBS surface. Skinning, which is also known as lofting, is a process of blending the cross sectional curves to form a surface [82]. Until recently, many algorithms have been proposed to reconstruct three-dimensional surface from a cross sectional data set using triangulation methods [14, 69, 70]. However, rectangular methods result in efficient data reduction since each skinned region is approximated by a NURBS surface rather than a triangular surface interpolant [15].

The proposed algorithm generates global point-to-point matching of the contours on a skinned region by a set of ruling curves Samanta et al. [83]. If the data is given as a mesh of the object, the input cross-sections can be obtained by slicing the mesh in certain intervals. If the data is given as point sets or polygonal contours, a closed B-spline curve

that satisfies positional continuity (C^0 continuity) can be fitted through the data points. A B-spline curve of degree p can be defined by the following equation 3.1 [82].

$$C(u) = \sum_{i=1}^{n} N_{i,p}(u) P_i \qquad a \leqslant u \leqslant b \tag{3.1}$$

Where $\{P_i\}$ are the control points, and the $\{N_{i,p}(u)\}$ are the p^{th} degree B-spline basis functions defined on nonuniform knot vector $U = \{u_0, \ldots, u_h\}$, where $h = n + p + 1$.

The developed algorithm projects input contours to the highest plane and generates ruling lines connecting the projected contours on one plane. As the number of input curves increase, the number of ruling lines increase. In order to reduce the number of ruling lines and preserve the topological information of the contours, only the points where the curvature changes the most are used for curve matching. Curvature is the degree to which a curve deviates from a straight line. This algorithm is applied to the points where the curvature begins to decrease after a local maximum and where the curvature begins to increase after a local minimum. Thus, the proposed algorithm generates a near optimal surface also for the complex shapes where the curvature changes frequently through the curve. By matching the points where the curvature starts to decrease and increase, it is enabled to reconstruct a surface from the contours with different shapes.

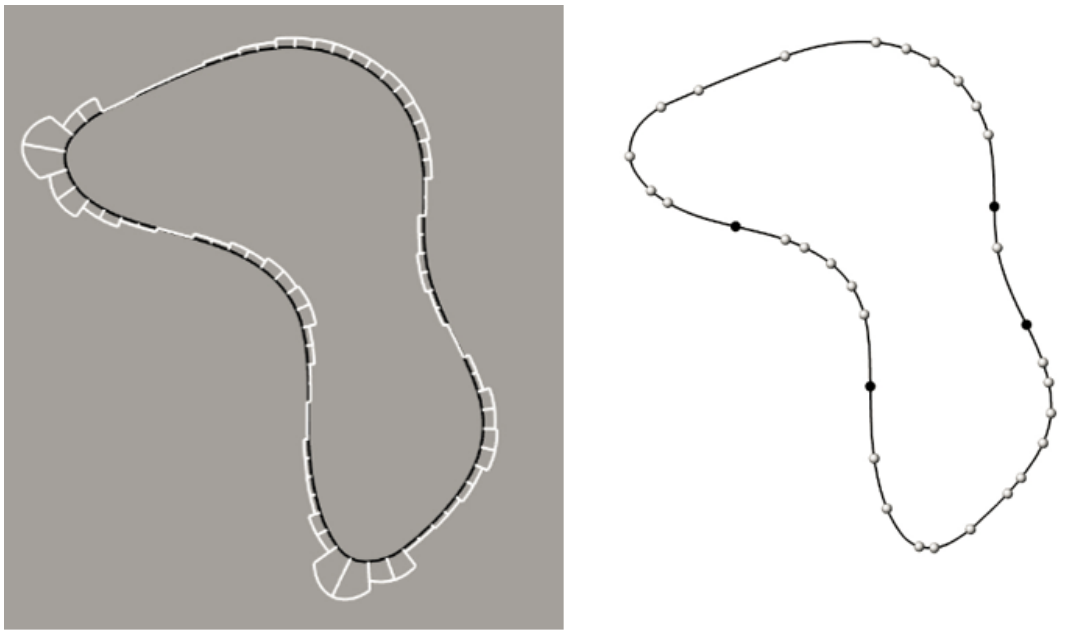

Figure 3.9: Curvature evaluation of the curve **(a)** the curvature graph **(b)** the points in *CurvaturePoints* list

First, the curvature value on sampled points on the curve are calculates as shown in 3.9(a). Then, the local minimum and maximum curvature points on each contour of a skinned region are found and added to *CurvaturePoints* list. White points in Figure 3.9(b) indicate the maximum curvature points where the curvature starts to decrease in both directions from the points. Black points show the minimum curvature points where the curvature changes its direction from one side of the curve to the other side. The curvature at the black points is close to 0 since curvature radius is close to infinity. Curvature is calculated as the rate of change in tangential angle ϕ with respect to the arc length s (Equation 3.2) [84].

$$\kappa = \frac{\partial \phi}{\partial s} \tag{3.2}$$

The proposed method generates ruling curves between the contours by using the curvature points. The input B-spline curves $C(u)$ are assumed to be non-self-intersecting, closed, at least positional continuous (C^0 continuous) and planar. As a first step, each pair of contours, $C_i(u)$ and $C_j(v)$ of degrees p_i and p_j respectively, in *SkinList(k)* which is achieved by the region detection algorithm is projected to the plane where the contour with the biggest area lies. The projected contours, $C_i'(u)$ and $C_j'(v)$, do not self-intersect and do not intersect with each other because of the containment assumption. In other words, they satisfy the following conditions [83].

$$\begin{aligned}
&C_i'(u_i) \neq C_i'(u_j); && u_i \in (u_{p_1}, u_{h_1-p_1}); && u_i \neq u_j \\
&C_j'(v_i) \neq C_j'(v_j); && v_i \in (v_{p_2}, v_{h_2-p_2}); && v_i \neq v_j \\
&C_i'(u) \neq C_j'(v); && u \in [u_{p_1}, u_{h_1-p_1}]; && v \in [v_{p_2}, v_{h_2-p_2}]
\end{aligned} \tag{3.3}$$

To connect the contours in 3D space, the curvature points are connected to a contour using a heuristic optimization method Samanta et al. [83]. Our method tries connect each minimum and maximum curvature point to the highest contour, to ensure smooth global connection and to reduce computational complexity. In our algorithm, a ruling line is introduced between the first and the last contour in a skinned region and passes over a curvature point. After that, the intersection points between a ruling line and each projected contour are found. These intersection points are projected back to the contours

they belong and the projected points are kept in a list, *PointList*. Next, a ruling curve *RC(t)* is fitted to the points in the list. A ruling line *RL(t)*, passing over a curvature point $\boldsymbol{q_{ik}}$, is introduced only when the following two conditions are satisfied [83]:

(a) Smooth transition requires the normal vectors of matching points to be collinear. Therefore, the curvature point $\boldsymbol{q_{ik}}$, which is the k^{th} curvature point on the i^{th} contour of a skinned region, is matched with $\boldsymbol{p_{jt}}$, t^{th} point on the j^{th} contour of the same skinned region, if the point $\boldsymbol{p_{jt}}$ minimizes the angle between their unit normal vectors. Thus, the inner product of the unit normal vectors to the curves at $\boldsymbol{q_{ik}}$ and $\boldsymbol{p_{jt}}$ is maximized.

(b) To avoid buckling on the surface, the length of ruling lines needs to be minimized. Since matching contours lie on different planes, the length of the ruling line is calculated as the distance between projected matching points by ignoring the distance in z direction.

By combining two conditions, a function f can be defined that assigns a value to a ruling line that is connecting two points $\boldsymbol{p_{1t}}$ and $\boldsymbol{p_{nt'}}$, a point on the first contour and a point on the last contour respectively, and passing over a curvature point $\boldsymbol{q_{ik}}$ as follows.

$$f(\boldsymbol{p_{jt}}, \boldsymbol{q_{ik}}) = \frac{\langle \boldsymbol{N}(\boldsymbol{p_{jt}}), \boldsymbol{N}(\boldsymbol{q_{ik}}) \rangle^{w^n}}{|\overrightarrow{\boldsymbol{p_{jt}} - \boldsymbol{q_{ik}}}|^{w^d}} \tag{3.4}$$

In the function f, w^n stands for the weight factor of the normal similarity and w^d is the weight factor of the ruling line length. Different weight factors can be applied to achieve the optimum matching. Since the space curves are project to a single plane, the curves where the unit normal vector is calculated become planar curves. For the planar curves, a unit normal vector of a planar curve at a parameter u can be calculated by rotating the tangent vector at parameter u contour-clockwise by an angle of $\frac{\pi}{2}$ [85]. The coordinates of the tangent vector $T(u)$ of a space curve C at a parameter u can be calculated by using the Equation 3.5.

$$\begin{aligned} T(u) &= (\dot{x}, \dot{y}) \\ \dot{x} &= \frac{\partial x(u)}{\partial u} \qquad \dot{y} = \frac{\partial y(u)}{\partial u} \end{aligned} \tag{3.5}$$

Then, the normal vector is obtained by multiplying $\frac{\pi}{2}$ contour-clockwise rotation matrix and the tangent vector by the Equation 3.6. The curve directions should be matched before the normal vector calculation.

$$
\begin{aligned}
N(u) &= (x', y') \\
\begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{bmatrix} \cos\left(\frac{\pi}{2}\right) & -\sin\left(\frac{\pi}{2}\right) \\ \sin\left(\frac{\pi}{2}\right) & \cos\left(\frac{\pi}{2}\right) \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \\
N(u) &= (-\dot{y}, \dot{x})
\end{aligned}
\tag{3.6}
$$

Therefore, the skinning problem can be formulated as an optimization problem where the objective is to maximize the function f for each curvature point that is defined on a contour in a skinned region Samanta et al. [83].

Maximize f
Subject to

1. Two consecutive ruling lines connecting the first and the last contours $\overline{C'_1(u_i) - C'_n(v_j)}$ and $\overline{C'_1(u_{i+1}) - C'_n(v_{j+1})}$ should not intersect.

2. Any ruling line $\overline{C'_1(u_i) - C'_n(v_j)}$ should have one and only one intersection point with all the contour in a skinned region, $C'_1(u), \ldots, C'_n(v)$.

On each projected curve C'_i, points are sampled at certain intervals taken as input. Resulting set of $(a + 1)$ points constitute the members of a candidates list for the i^{th} projected contour and *CandidatesList*$_i$ can be defined as follows:

$$
P_i = \{p_{it}\}_{t=0,\ldots,a} \qquad \text{for all } i = 1, \ldots, n; \qquad \text{where} \qquad p_{it} = C'_i(u_t); \qquad u_t < u_{t+1}
\tag{3.7}
$$

Starting from the first projected contour to the last one, at each iteration our algorithm adds a ruling line which is achieved by solving the optimization problem to the *RL* list until one and only one ruling line is passing over each curvature point $\boldsymbol{q_{ik}}$. At every step, a matching point that satisfies both constraints and maximize the function f, from the *CandidatesList*$_n$ when the curvature point lies on the 1^{st} projected contour and from the *CandidatesList*$_1$ for the other curvature points, is selected to define a ruling line.

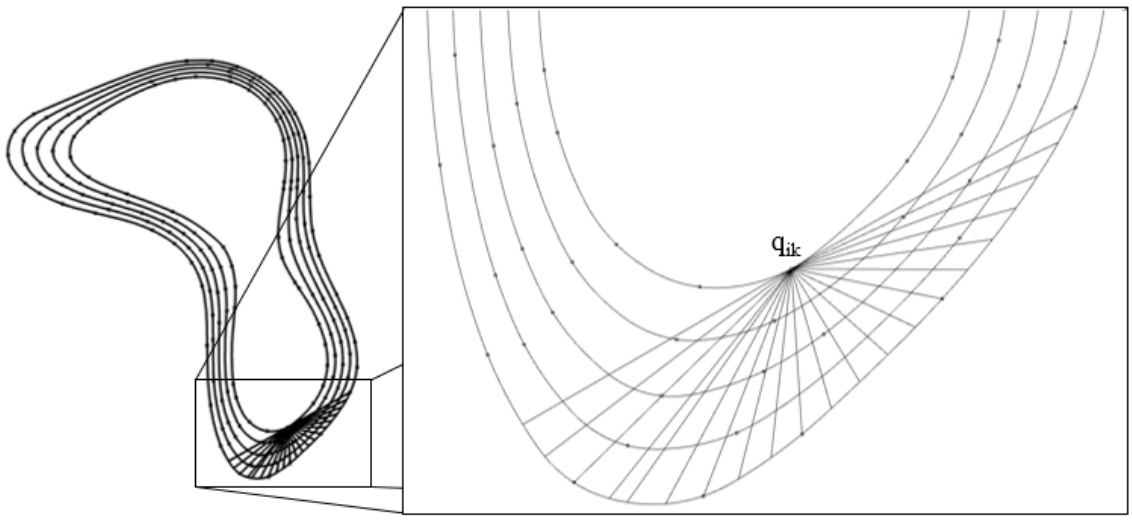

Figure 3.10: Ruling lines that connect $\boldsymbol{q_{ik}}$ and all visible points in $\boldsymbol{V_{ikj}}$

The second constraint is met by a visibility checking method Samanta et al. [83]. To find a matching point for the curvature point $\boldsymbol{q_{ik}}$, only the points in the *CandidatesList*$_1$ or *CandidatesList*$_n$ are considered which are visible to $\boldsymbol{q_{ik}}$. A Boolean function **IsVisible(**$\boldsymbol{q_{ik}}$**,** $\boldsymbol{p_{jt}}$**)** returns *true* only if a ruling line $\overline{\boldsymbol{q_{ik}p_{jt}}}$ intersects with the i^{th} and j^{th} projected contours only on the points $\boldsymbol{p_{jt}}$ and $\boldsymbol{q_{ik}}$. A list $\boldsymbol{V_{ikj}}$, which includes all visible points on the j^{th} contour to the curvature point $\boldsymbol{q_{ik}}$, is the subset of *CandidatesList*$_j$ and $\boldsymbol{V_{ikj}}$ is defined as follows [83].

$$\boldsymbol{V_{ikj}} = \{\boldsymbol{p_{jt}} \in \textit{CandidatesList}_{\boldsymbol{j}} | \textbf{IsVisible}(\boldsymbol{q_{ik}}, \boldsymbol{p_{jt}}) = \textit{true}\}_{t=0,\ldots,a_j}; \qquad \text{for all } j = 1, \ldots, n \tag{3.8}$$

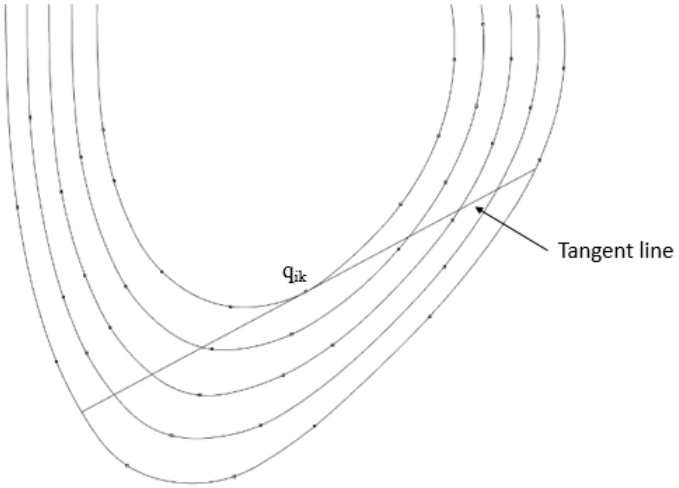

Figure 3.11: Boundaries of the reduced search space of ruling lines connecting $\boldsymbol{q_{ik}}$ and the points in $\boldsymbol{V_{ikj}}$

To find the ruling lines that connect $\boldsymbol{q_{ik}}$ and visible points on the j^{th} contour, a line is drawn between $\boldsymbol{q_{ik}}$ and each sampled point on the j^{th} contour. The search space can be reduced by using a tangent line passes over the curvature point, $\boldsymbol{q_{ik}}$, because the ruling lines, that are out of the region covered by the tangent line, intersect with the i^{th} contour where the curvature point lies. Therefore, only the sampled points which stays within the tangent line are examined, as the boundaries are shown in Figure 3.11.

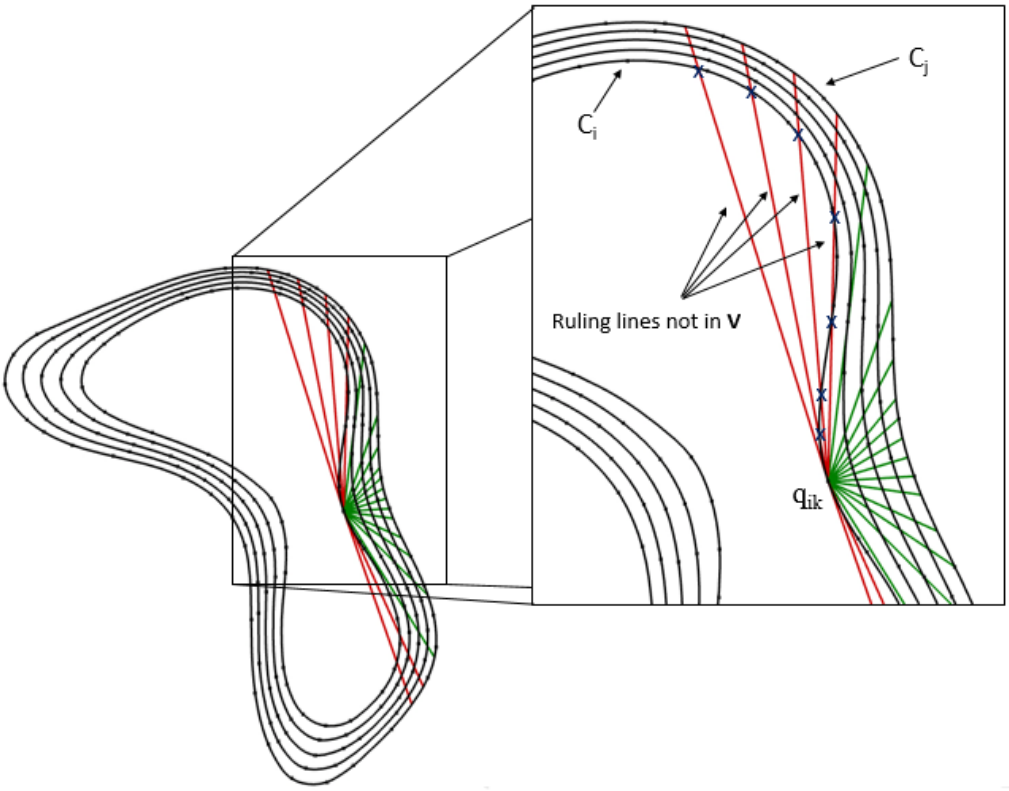

Figure 3.12: The ruling lines that connect $\boldsymbol{q_{ik}}$ and both visible and non-visible sampled points within a reduced search space by the tangent line

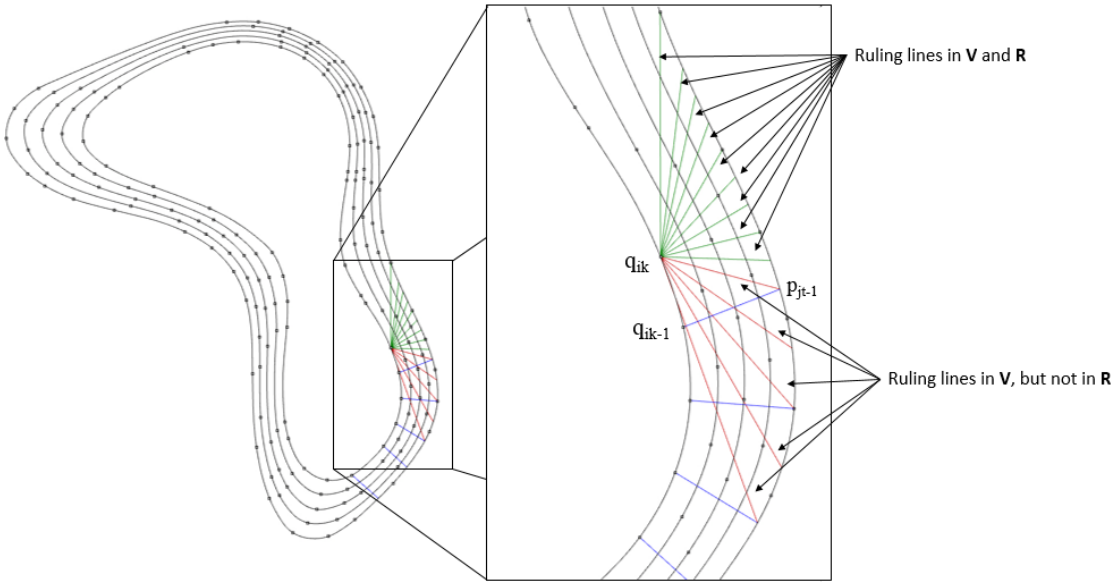

Figure 3.13: Finding ruling lines passing through $\boldsymbol{q_{ik}}$ and satisfying both constraints 1 and 2

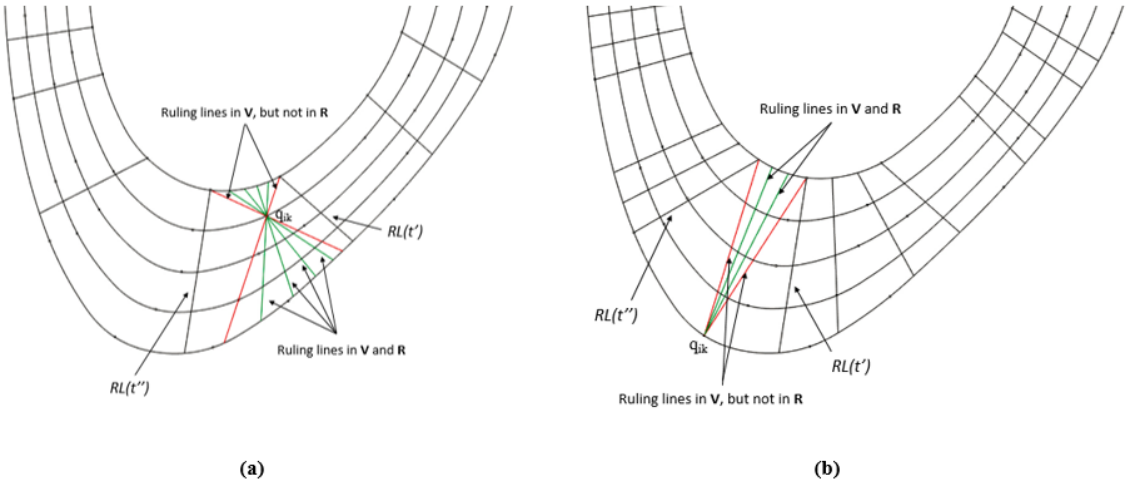

Figure 3.14: Finding ruling lines passing through $\boldsymbol{q_{ik}}$ and satisfying both constraints 1 and 2 **(a)** Ruling lines passing over a curvature point on mid-contours **(b)** Ruling lines passing over a curvature point on the 1^{st} contour

Similarly, the first constraint is checked by a boolean function **IsValid($\boldsymbol{q_{ik}}$, $\boldsymbol{p_{jt}}$)** Samanta et al. [83] While traversing the points starting from $\boldsymbol{p_{jt+1}}$ until $\boldsymbol{p_{jt-1}}$, let $RL(t')$ and $RL(t'')$ be the first and the last ruling lines encountered. **IsValid($\boldsymbol{q_{ik}}$, $\boldsymbol{p_{jt}}$)** function returns *true* only if there is no other ruling line between $\boldsymbol{q_{ik}}$ and $RL(t')$ and between $\boldsymbol{q_{ik}}$ and $RL(t'')$, as shown in Figure 3.14. A list $\boldsymbol{R_{ikj}}$ is subset of $CandidatesList_j$ and contains all valid points to $\boldsymbol{q_{ik}}$ [83].

$$\boldsymbol{R_{ikj}} = \{\boldsymbol{p_{jt}} \in \textit{CandidatesList}_j | \textbf{IsValid}(\boldsymbol{q_{ik}}, \boldsymbol{p_{jt}}) = \textit{true}\}_{t=0,\ldots,a_j}; \qquad \text{for all } j = 1, \ldots, n \tag{3.9}$$

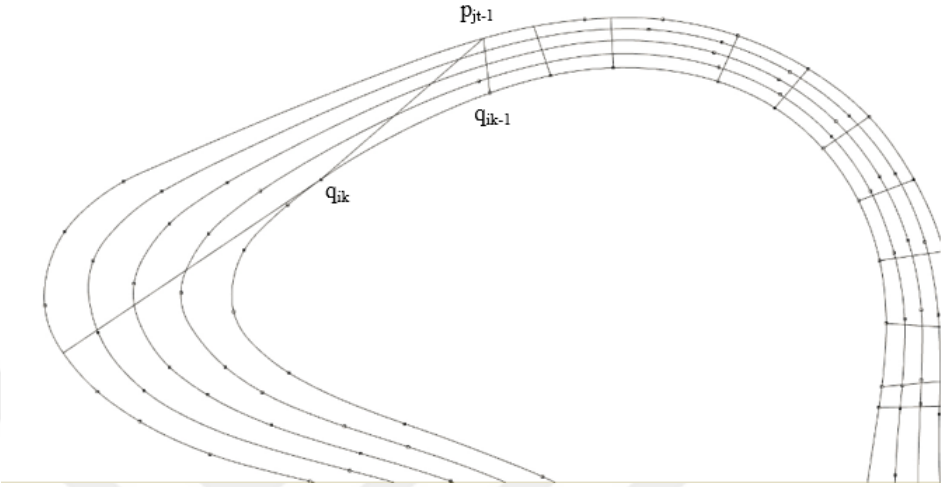

Figure 3.15: Finding ruling lines passing through $\boldsymbol{q_{ik}}$ and satisfying both constraints 1 and 2

Similarly, in the first iteration after adding the first ruling line to the *RL* list, the search space for the curvature points, $\boldsymbol{q_{ik}}$ where $k \neq 1$, can be reduced by taking one boundary as a part of the tangent line and the other boundary as the line between $\boldsymbol{q_{ik}}$ and the point $\boldsymbol{p_{jk-1}}$ of last element of the *RL* list, as shown in Figure 3.15.

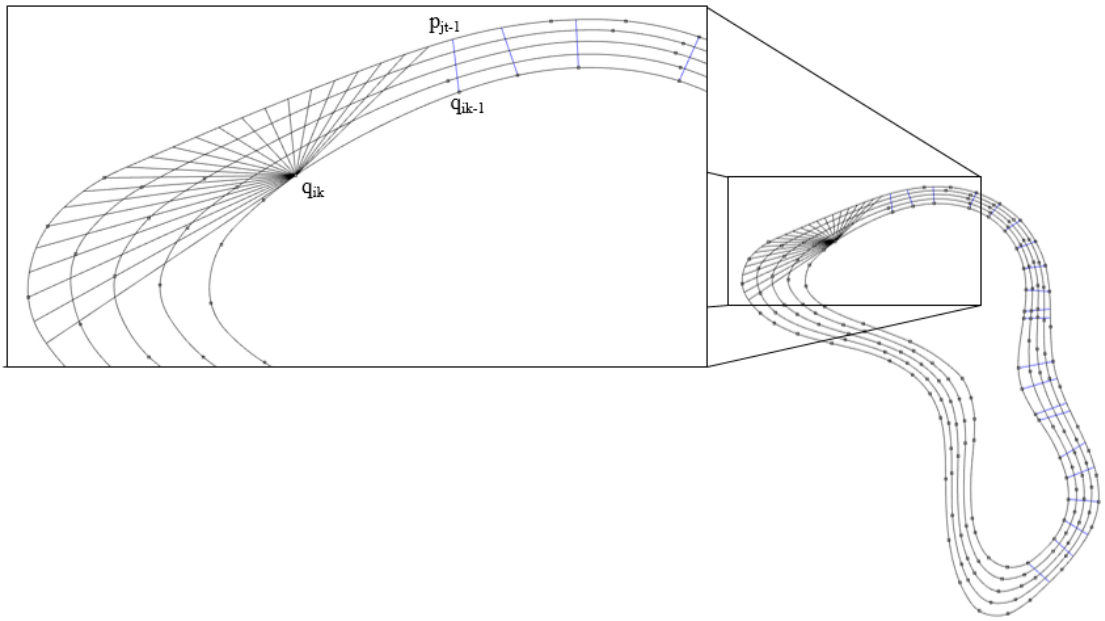

Figure 3.16: Finding all ruling lines passing through $\boldsymbol{q_{ik}}$ and satisfying both constraints 1 and 2

A point $\boldsymbol{p_{jt}}$, which defines a ruling line passing over $\boldsymbol{q_{ik}}$, is found by selecting a point in $\boldsymbol{V_{ikj}} \cap \boldsymbol{R_{ikj}}$ that maximizes the function f (Figure 3.13). The pseudo code to find all ruling lines is given in Algorithm 2 below.

Algorithm 2 Finding Optimal Set of Ruling Lines

INPUT: ***CandidatesList***$_j$ $\in$ ***CandidatesList***: A list of sampled points $\boldsymbol{p_{jt}}$ on the j^{th} projected contour, $j = 1, \ldots, n$ and $t = 1, \ldots, b$

$\boldsymbol{N_j} \in$ ***Normal***, $\boldsymbol{N_j} = \boldsymbol{N}(\boldsymbol{p_{jt}})$: A set of the unit normal vector associated with each sampled point on the j^{th} projected contour

$\boldsymbol{q_{ik}} \in$ ***CurvaturePoints*** : the k^{th} curvature point on the i^{th} projected contour, $i = 1, \ldots, n$ and $k = 1, \ldots, c$

$\boldsymbol{N}(\boldsymbol{q_{ik}}) \in$ ***Normal***: The unit normal vector associated with $\boldsymbol{q_{ik}}$

OUTPUT: ***RL*** : A list of optimum ruling lines passing over each curvature point

ProjectedPointList(m) $\in$ ***ProjectedPointList*** : A list of intersection points of a ruling line ***RL(m)*** and each projected contour

procedure OPTIMALRULINGLINES(***CurvaturePoints***, ***Normal***, ***CandidatesList***)
    Initiliaze Max, Value, MatchingPoint, ConnectingPoint, ConnectingCandidate, IntersectP ;
    for each $\boldsymbol{q_{ik}}$ in ***CurvaturePoints*** **do**
        if $i = 1$ **then** /* If the curvature point lies on the first projected contour in a skinned region */
            FindRulingLine($\boldsymbol{q_{ik}}$, ***CandidatesList***$_n$, $\boldsymbol{N}(\boldsymbol{q_{ik}})$, $\boldsymbol{N_n}$;
        else
            FindRulingLine($\boldsymbol{q_{ik}}$, ***CandidatesList***$_1$, $\boldsymbol{N}(\boldsymbol{q_{ik}})$, $\boldsymbol{N_1}$;
        end if
    end for
    return ***RL*** , ***PointList*** ;
end procedure
procedure FINDRULINGLINE($\boldsymbol{q_{ik}}$, ***CandidatesList***$_j$, $\boldsymbol{N}(\boldsymbol{q_{ik}})$, $\boldsymbol{N_j}$)
    for $t = 1$ to $b, t++$ **do**
        if IsVisible($\boldsymbol{q_{ik}}$, $\boldsymbol{p_{jt}}$)and **IsValid**($\boldsymbol{q_{ik}}$, $\boldsymbol{p_{jt}}$) **then**
            Draw a line between $\boldsymbol{p_{jt}}$ and $\boldsymbol{q_{ik}}$;
            Make the line longer in the direction of $\overrightarrow{\boldsymbol{q_{ik}} - \boldsymbol{p_{jt}}}$;
            ConnectingCandidate $\leftarrow$ the insection point between the line and the n^{th}

projected contour;

Calculate Value = $f(\boldsymbol{p_{jt}}, \text{ConnectingCandidate})$ from Equation 3.4 ;

if $t = 1$ **then** /* if it is the first step in the second for loop */

Max $\leftarrow$ Value;

MatchingPoint $\leftarrow \boldsymbol{p_{jt}}$;

ConnectingPoint $\leftarrow$ ConnectingCandidate;

else

if Value $>$ Max **then**

Max $\leftarrow$ Value;

MatchingPoint $\leftarrow \boldsymbol{p_{jt}}$;

ConnectingPoint $\leftarrow$ ConnectingCandidate;

end if

end if

end if

end for

Draw a line between the MatchingPoint and ConnectingPoint;

Add this line, ***RL(m)***, to the ***RL*** list;

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

IntersectP $\leftarrow$ the intersection point between ***RL(m)*** and the projected i^{th} contour;

Add IntersectP to the ***ProjectedPointList(m)***;

if IntersectP $\in$ ***CurvaturePoints*** **then**

Remove IntersectP from ***CurvaturePoints*** ;

end if

end for

return ***RL*** , ***ProjectedPointList*** ;

end procedure

Before running the algorithm, the contours in a skinned region are projected to the plane that the contour with the largest area lies. Algorithm 2 works on projected contours and takes sampled points and curvature points on the projected contours as input. For each projected curvature point, algorithm tries to find a matching point on the projected contour with largest area which is shown as C_1' but if the curvature point is on the C_1' then

algorithm tries to find a matching point on the smallest contour C'_n. For the curvature points that are not on the 1^{st} contour, the second for loop searches through the sampled points on the 1^{st} contour that satisfy both constraints and return *true* from the **IsValid** and **IsVisible** functions. For the curvature points on the 1^{st} contour, the second for loop examines the sampled points on the last contour for the constraints. Then, a line is drawn between the sampled point $\boldsymbol{p_{jt}}$ and the curvature point $\boldsymbol{q_{ik}}$. After that, the line is scaled until it intersects with the last projected contour C'_n. This intersection point is referred as *ConnectingCandidate*. The objective function value is calculated by using the scaled line that is drawn between $\boldsymbol{p_{jt}}$ and *ConnectingCandidate* and passing over $\boldsymbol{q_{ik}}$ (Figure 3.14). The line that maximizes the objective function is added to the ruling line list, *RL*. In the third for loop, a *ProjectedPointList(m)* is created for each ruling line *RL(m)* and intersection points between the ruling line and each projected contour is added to the *ProjectedPointList(m)*. If one of the intersection points is one of the curvature points, the curvature point is removed from the curvature point list. Therefore, a ruling line may pass over more than one curvature point. At the end of the algorithm, ruling line list *RL* and the *ProjectedPointList* is returned as output.

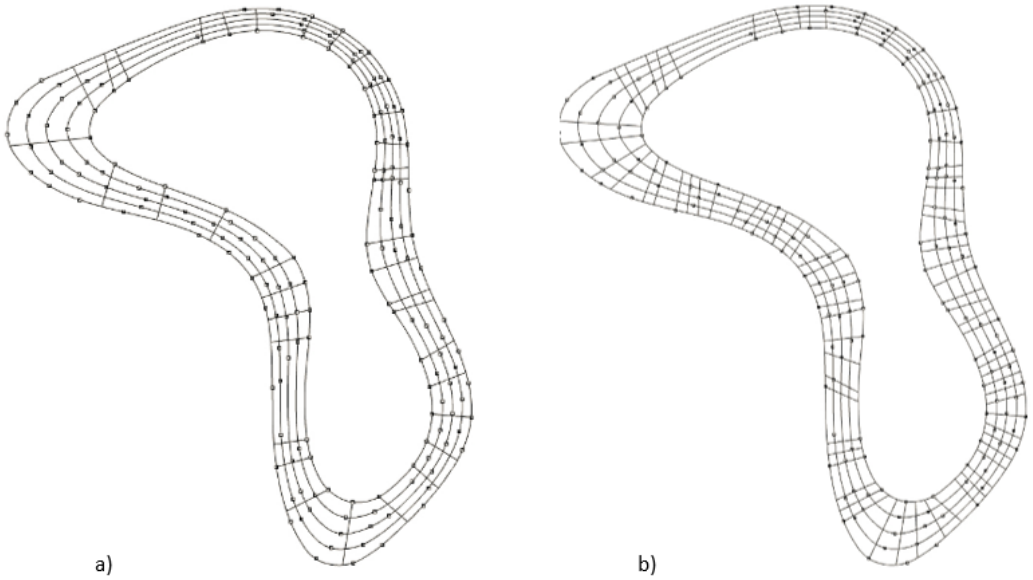

Figure 3.17: The ruling lines in the *RL* list **(a)** after the first iteration of the procedure *OptimalRulingLines* **(b)** when the procedure *OptimalRulingLines* ends

In order to decrease storage requirements and computational time, redundant ruling lines need to be removed from the ruling line list *RL*. A redundant ruling line can be

defined as a ruling line which has a length and an inclination within a predetermined tolerance range of another ruling line's length and inclination (Figure 3.18. In other words, the connection provided by a redundant ruling line can be represented by a very similar ruling line in the list. After obtaining the list *RL*, the redundant ruling lines are removed by the following algorithm.

1: **for each** ***RL(i)*** in ***RL*** **do**

2: **if** $|RL(i)| = |RL(i+1)| \mp t^d$ **and** $\nabla(RL(i)) = \nabla(RL(i+1)) \mp t^a$ **then**

3: ***RL = RL - RL(i)*** ;

4: ***ProjectedPointList = ProjectedPointList - ProjectedPointList(i)*** ;

5: **end if**

6: **end for**

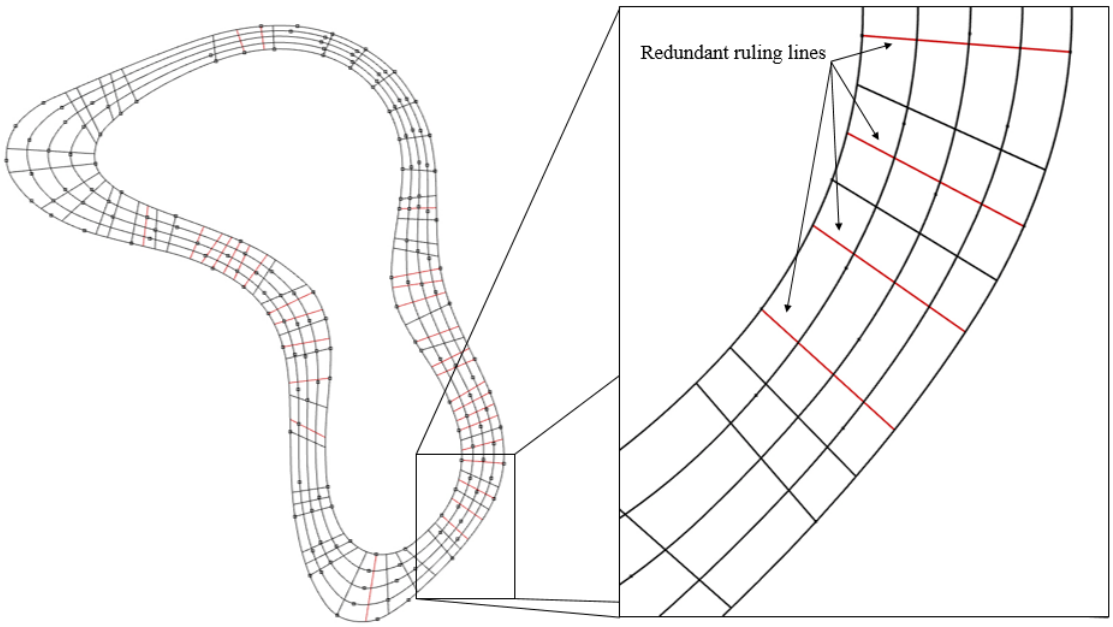

Figure 3.18: Finding ruling lines passing through q_{ik} and satisfying both constraints 1 and 2 **(a)** Ruling lines passing over a curvature point on mid-contours **(b)** Ruling lines passing over a curvature point on the 1^{st} contour

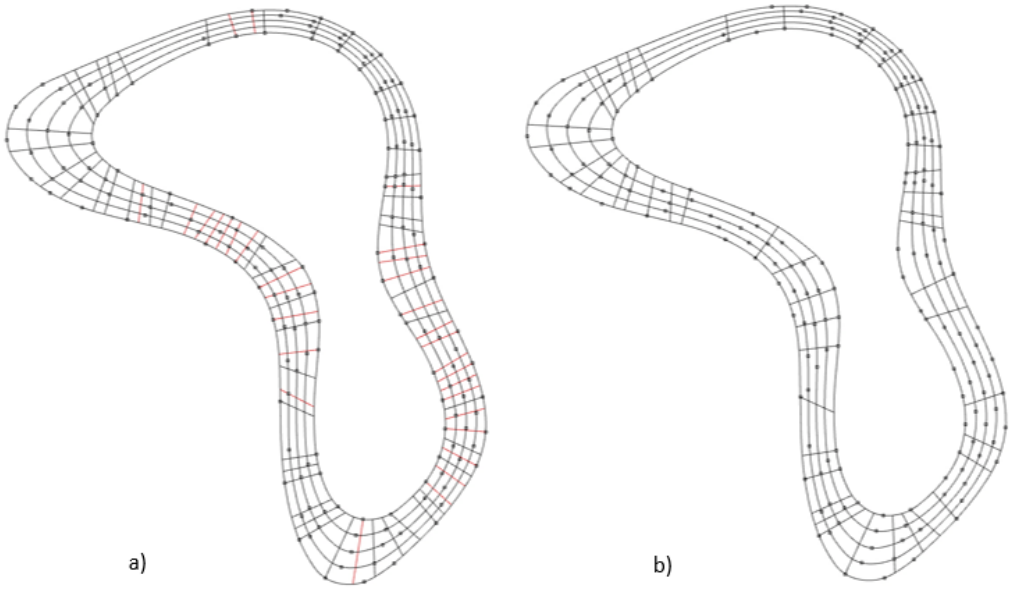

Figure 3.19: The ruling lines in the *RL* list **(a)** before and **(b)** after removing the redundant ruling lines

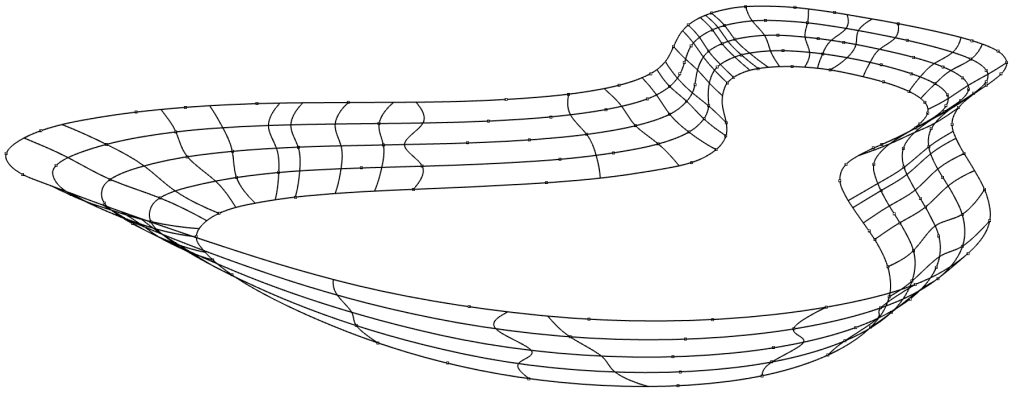

Figure 3.20: The ruling curves in the *RC* list

In order to create a surface, each point in the *ProjectedPointList* is projected back to the contour that they belong to. A three-dimensional ruling curve *RC(m)* is created by fitting a curve through the points in the *ProjectedPointList(m)*, as shown in Figure 3.20. Then, a NURBS surface is approximated by interpolating a bidirectional curve network where the curves in the ***RC*** list and the contours in the skinned region are taken as network curves. To interpolate a curve network, each curve in ***RC*** list and each contour in a skinned

region should have an intersection, satisfying the following condition [82]:

$$p_{kl} = C_k(u_l) = C_l(v_k) \qquad \forall k; \quad C_k \in \boldsymbol{RC} \quad \text{and} \quad \forall l; \quad C_l \in \boldsymbol{SkinList} \tag{3.10}$$

Therefore, resulting NURBS surface $S(u, v)$ satisfies the following equation.

$$\begin{aligned} S(u_l, v) &= C_l(v) \qquad \forall l \\ S(u, v_k) &= C_k(u) \qquad \forall k \end{aligned} \tag{3.11}$$

Figure 3.21 shows that approximated a NURBS surface and does not have any buckling and distortion.

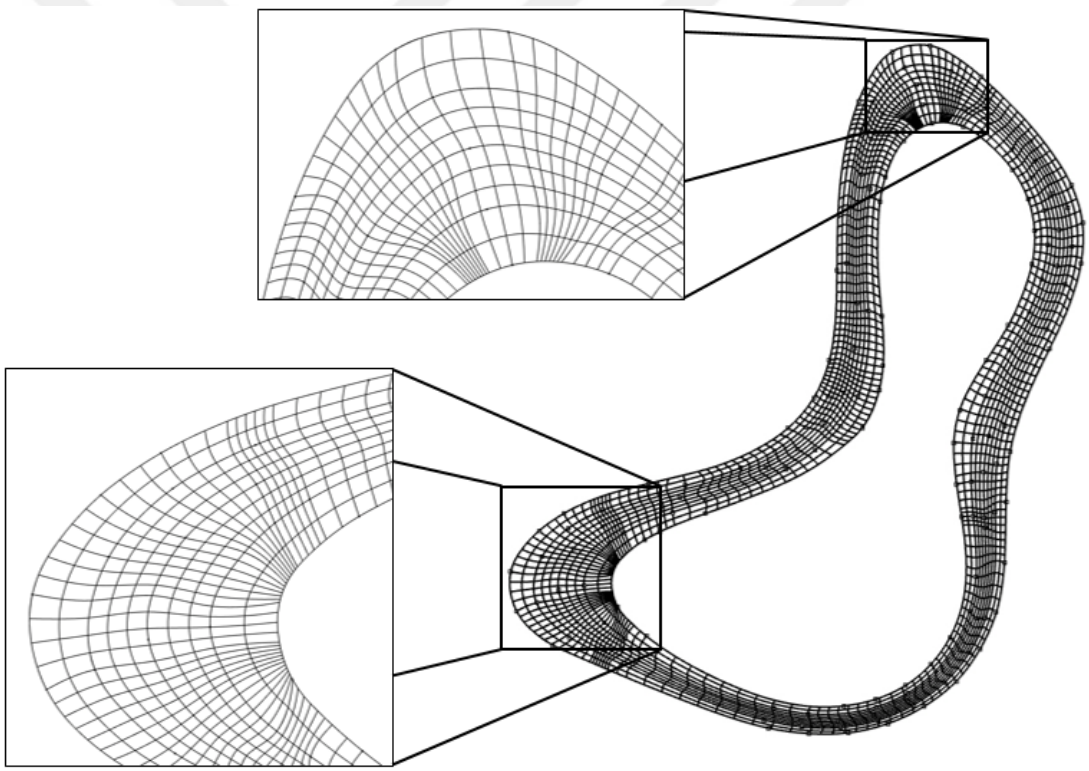

Figure 3.21: Wireframe display of the NURBS surface approximated through the curves in ***RC*** list and the contours

3.2.3 Branching

The contours in a branched region require one-to-many linking where the first contour, C_1, in the list lies on the upper plane and its area contains the rest of the contours on the lower plane. Many methods have been proposed to solve the branching problem and

most of them require user interaction [14]. In this step, our method decomposes a one-to-many linking problem into multiple one-to-one linking problem by transforming the first contour into multiple transition curves. Then, the skinning algorithm which is proposed in the previous section is applied to each contour in a branched region and its corresponding transition curve. The approximated NURBS surfaces for each Voronoi cell satisfies twist compatibility.

The transitions curves are created by using Voronoi diagram in the same way as it is used to solve one-to-many morphing problem on the same plane by Ozbolat and Koc [3]. The Voronoi diagram is generated by interpolating the offset curves as implemented by the several studies [3, 86, 87]. The offset C_i^d of a closed curve C_i is achieved by moving all locations on the curve C_i by a specified distance d. A point at a parameter u, $C_i^d(u)$ can be defined as follows.

$$C_i^d(u) = C_i(u) + \alpha d\overrightarrow{\boldsymbol{N}(u)} \tag{3.12}$$

Where $\overrightarrow{\boldsymbol{N}(u)}$ is the unit normal vector of the curve C_i at a parameter u and α is the direction sign. Alpha takes value of 1 for inward offsetting and -1 for outward offsetting. In this study, outward offsetting will be used to achieve Voronoi diagrams. The pseudo code of the branched surface generation is given in Algorithm 4 below.

Algorithm 4 Converting a branched region to one-to-one linking problem

INPUT: $\boldsymbol{C_i} \in$ ***BrList(k)*** : the i^th contour in the ***BrList(k)***, $i = 1, \ldots, n$ where ***BrList(k)*** $\in$ ***BrList***

$\boldsymbol{d}$: the offset distance

OUTPUT: ***VC*** : A list of Voronoi Cells related to $C_2, \ldots, C_n$

CSkinList(k) $\in$ ***CSkinList*** : A list of two contours that will be used to generate skinned surfaces

1: **procedure** CONVERTTOSKINNED(***BrList(k)***)

2: Initiliaze Binary, *VS*, *BrPt*, *VPtList*, *ProjectedList*, p_i, *IntersectP*, $\hat{d}$;

3: **for** $i = 2$ to $n, i++$ **do**

4: $\boldsymbol{C_i'} \leftarrow$ Project $\boldsymbol{C_i}$ to the upper plane; /* Project the curve to the plane where $\boldsymbol{C_1}$ lies */

5: Add $\boldsymbol{C_i'}$ to *ProjectedList*;

6: **end for**

7: **for each** pair of $\boldsymbol{C'_a}$ and $\boldsymbol{C'_b}$ in *ProjectedList* **do**

8: **while** Binary $= true$ **do** /* do until an intersection point lies outside of $\boldsymbol{C_1}$ */

9: $\boldsymbol{C'^{\hat{d}}_a} \leftarrow$ offset of $\boldsymbol{C'^{\hat{d}}_a}$;

10: $\boldsymbol{C'^{\hat{d}}_b} \leftarrow$ offset of $\boldsymbol{C'^{\hat{d}}_b}$;

11: Find intersection points between $\boldsymbol{C'^{\hat{d}}_a}and\boldsymbol{C'^{\hat{d}}_b}$;

12: **if** all intersection points are inside of $\boldsymbol{C_1}$ **then**

13: Add intersection points to *VPtList* ;

14: $\hat{d} \leftarrow \hat{d} + d$;

15: **else**

16: Add intersection points to *VPtList* ;

17: Binary $\leftarrow false$;

18: **end if**

19: **end while**

20: Sort the points in *VPList* in a reasonable polyline order;

21: Fit a curve, VS_i, through the points in *VPList*;

22: **end for**

23: **for each** pair of VS_a and $VS_b \in VS$ **do**

24: $p_i \leftarrow$ intersection point of VS_a and VS_b;

25: $i \leftarrow i++$;

26: **end for**

27: *BrPt* = the center point of a circle defined by p_0, p_1andp_3 ;

28: **for each** $VS_i \in VS$ **do**

29: $p_i \leftarrow$ intersection point of *VS* and C_1;

30: Add p_i to the list *IntersectP*;

31: Trim VS_i by preserving the curve between p_i and *BrPt*;

32: **end for**

33: Segment C_1 at the points in *IntersectP*;

34: **for** $i = 2$ to $n, i++$ **do**

35: $\boldsymbol{VC_i} \leftarrow$ merge the segments of C_1 and segments in *VS* that are enclosing C'_i;

36: Add $\boldsymbol{VC_i}$ and C_i to ***CSkinList(i-1)***;

37: **end for**

38: **return** ***CSkinList*** , ***VC*** ;

39: **end procedure**

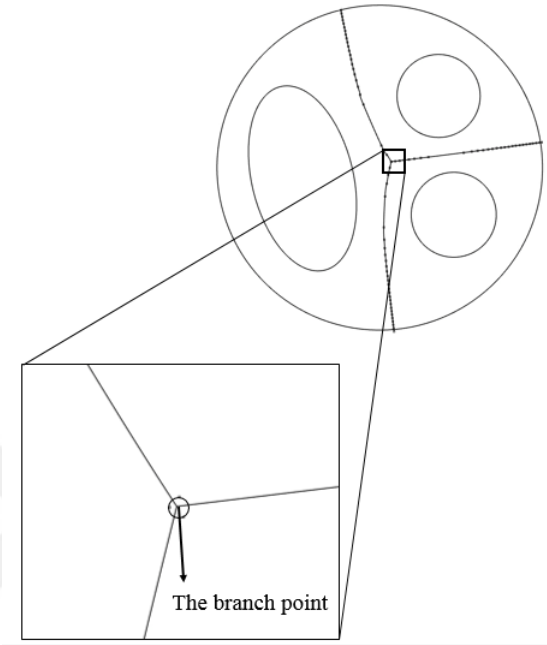

Figure 3.22: Finding the branch point when there are more than two contour on the same z plane

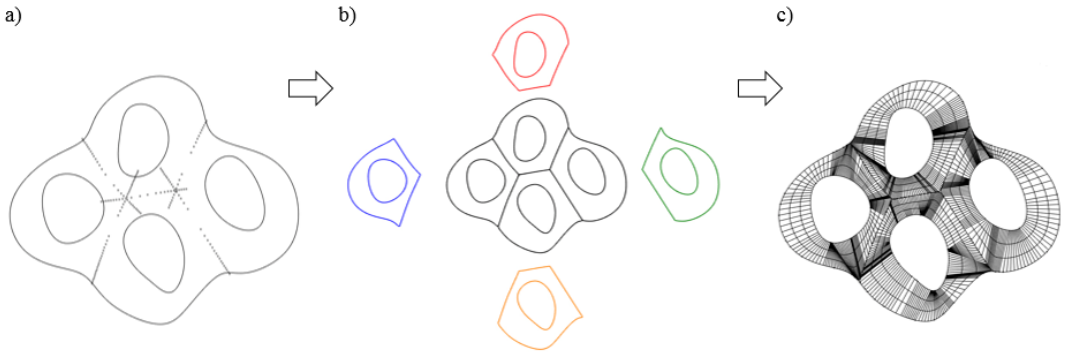

Figure 3.23: Depiction of a branched surface generation **a)** Intersection points on the offsets of the projected curves, $C_a'^{\hat{d}}$ and $C_b'^{\hat{d}}$ **b)** Voronoi cells for each branch **c)** Wireframe display of resulting branched surface

This algorithm is applied to each *BrList(k)* in ***BrList***. First of all, the contours lying on the lower plane of a branched region are projected to the upper plane, since the Voronoi diagram needs to be created inside of the first contour, C_1. The intersection points of

each pair of projected contours $C_i'^d$ and $C_j'^d$ are added to the list, *VPtList*, recursively by incrementing the distance d until an intersection point lies outside of a region contained by C_1 and including the first outside point in the list. Then, a segment, VS_i, on a Voronoi boundary is generated by fitting a closed curve through the points in the list (Figure 3.23(a)). This procedure is repeated for each pair of contours lying on the lower plane of a branched region. A branch point, *BrPt*, is determined as the center of a circle defined by three points which are the closest points obtained by intersecting each pair of three Voronoi segments. After that, the segment, VS_i, is trimmed as it is terminated at the branch point and the excess portion lying outside C_1 is removed. Figure 3.22 depicts finding a branch point when there are more than 2 contours lying on the lower plane.

After obtaining each Voronoi segment, VS_i, the curve C_1 is split into pieces at the intersections points between C_1 and each Voronoi segment. A Voronoi cell, VC_i, which is enclosing the projected curve C_i' is obtained by merging the Voronoi segments around C_i' and corresponding curve segment of C_1. Figure 3.23(b) shows the Voronoi diagram consisting of several Voronoi cells. By decreasing the offset distance parameter d, more exact but computationally expensive Voronoi diagram can be obtained. Any other Voronoi diagram generation algorithm can be implemented in the branching algorithm. After achieving the Voronoi diagram, the skinning algorithm defined in the previous section is applied to each lower plane contour C_i and its corresponding Voronoi cell VC_i. Resulting smooth skinned surfaces are combined to achieve a branched surface. The branched surface generated by using the proposed algorithm on 4 branched region can be seen in Figure 3.23(c) and Figure 3.24.

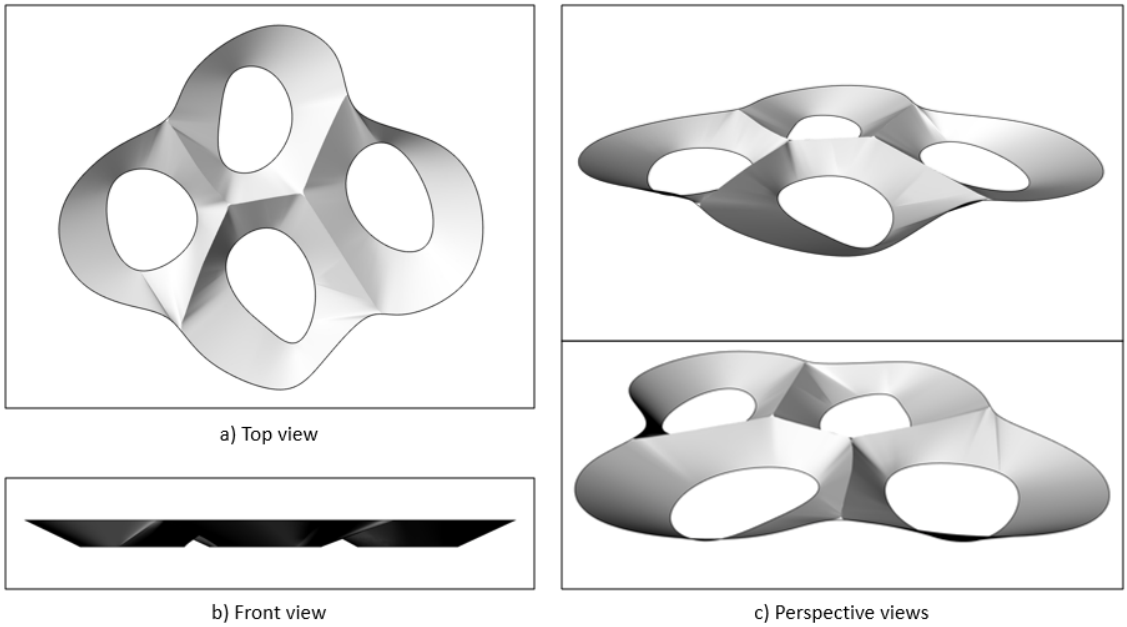

Figure 3.24: The resulting branched surface from different views

Since in this study, the Voronoi segments are generated from offset curves, the resulting Voronoi segments may not be G^0 continuous (position continuous). Therefore, the resulting branched regions also fail to demonstrate G^0 continuity; the resulting surfaces might not share a common edge or they might intersect with each other. However, the Voronoi diagram generation is not the main scope of this study. Thus, G^0 continuous Voronoi diagrams can be obtained by several methods [88–92] or further merging and smoothing operations can be applied to achieve G^0 continuity in branched regions.

3.2.4 Capping

Capping or contour closing is applied to isolation curve which is the lowest curve in a skinned region and is not involved in a branched region. In order to achieve dome shaped closing surfaces, additional circles are drawn at the area centroid of the isolation curve with a small diameter satisfying that the area bounded by the additional circle is involved in the area bounded by the isolation curve. The circle is located below the isolation curve with a distance of the smallest difference in z values of the consecutive cross-sections. Then, a NURBS surface patch is created on the isolation and the circle and the patched surface is trimmed based on the isolation curve. Therefore, the trimmed planar NURBS surface does not have any holes and fills the boundary completely. The capped surface S(u,v) of the the isolation curve can be defines as a NURBS surface of degree p in the u

direction and degree p in the v direction [82] :

$$S(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_{i,j}} \tag{3.13}$$

where the $\{\mathbf{P}_{i,j}\}$ are the control points of the capped surface, the $\{w_{i,j}\}$ are the weights of the control points, and the $\{N_{i,p}(u)\}$ and $\{N_{j,q}(v)\}$ are the B-spline basis functions defined on the knot vectors, U and V.

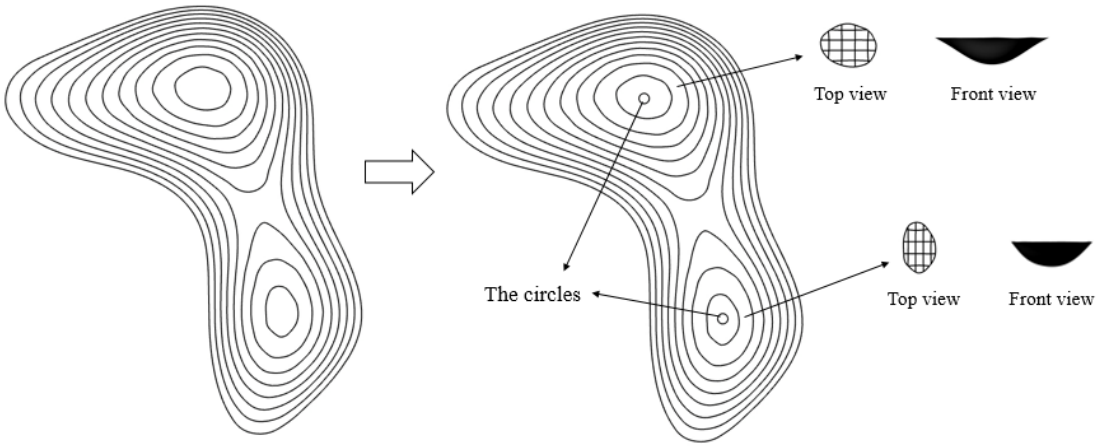

Figure 3.25: Depiction of capped surface generation of two isolation curves. Two NURBS surfaces with degree 3 and 8 control points in both direction u and v, are generated and trimmed with the isolation curves.

If the contour data is dense, additional circle may not be needed and a planar NURBS surface is approximated only on the isolation curve.

3.3 Toolpath Planning for Scaffold-based Bioprinting

In order to bioprint the patient-specific wound matrix by layer-by-layer deposition, the scaffold model is generated based on the resulting surface of the wound. The internal architecture of the surface is formed by depositing cylindrical fibers layer by layer with a certain infill pattern and infill ratio. Infill ratio can be defined as the ratio of printed fiber volume to total volume. A linear pattern is one of the common infill patterns that is used to generate porous scaffold, since it ensures interconnected internal architecture to enhance cell ingrowth and proliferation and it is easy to control infill ratio by using pore

size, d^p. In linear patterns, cylindrical fiber is deposited parallel to each other on every layer and the angle between the deposited fibers of consecutive layers, α, is 90° as shown in Figure 3.26.

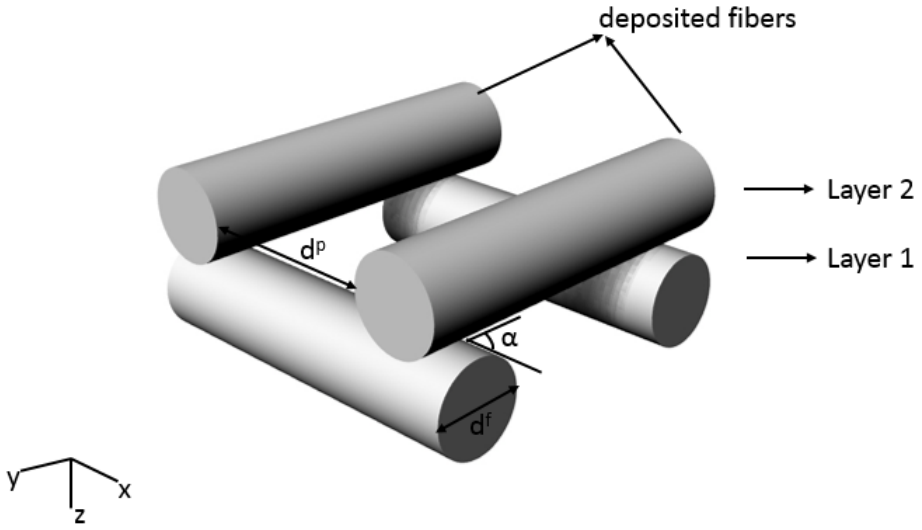

Figure 3.26: Representation of two layer deposition

A VBScript code is developed to generate the porous scaffold model and the G-code for deposition based manufacturing techniques by using a linear infill pattern. The developed code takes the wound surface S, the fiber diameter d^f and the pore size d^p as input. Following Algorithm 5 explains the steps of the tool path generation for a patient-specific wound geometry.

Algorithm 5 Toolpath planning for porous scaffolds

INPUT: d^f : the fiber diameter

d^p : the pore size

S : the surface for toolpath generation

OUTPUT: The porous scaffold

Toolpath : an array of the points that the nozzle will follow

1: **procedure** TOOLPATHPLANNING(d^n, d^p, S)

2: Create a bounding box on S;

3: Adjust the bottom and top plate of the bounding box by $d^p/2$ calculating the nozzle center point;

4: Draw contours on S with an increment of d^p, indicating each layer to be printed;

5: **for each** contour **do**

6: **if** the layer number is even **then**
7: **Layer(d^n, d^p, contour, x, y)** ;
8: **else**
9: **Layer(d^n, d^p, contour, y, x)** ;
10: **end if**
11: **end for**
12: **end procedure**
13: **procedure** LAYER(d^n, d^p, contour, i, j)
14: Create a bounding box on the contour;
15: x_{lowest} = the lowest x axis value of the bounding box;
16: $x_{highest}$ = the highest x axis value of the bounding box;
17: y_{lowest} = the lowest y axis value of the bounding box;
18: $y_{highest}$ = the highest y axis value of the bounding box;
19: **for** $k = i_{lowest}$ to $i_{highes}t$, with an increment of ($d^n + d^p$) **do**
20: Draw a line on the bounding box between the point ($i = k$, $j = j_{lowest}$) and the point ($i = k$, $j = j_{highest}$);
21: Then, draw a cylinder between the intersection points of the contour and the line;
22: Add the intersection points to the $Toolpath$ list;
23: $k = d^n + d^p$;
24: Draw a line on the bounding box between the point ($i = k$, $j = j_{highest}$) and the point (k, j_{lowest});
25: Then, draw a cylinder between the intersection points of the contour and the line;
26: Add the intersection points to the $Toolpath$ list;
27: **end for**
28: **end procedure**

Firstly, the script creates a bounding box over the object surface to be converted to porous scaffold. Then, it creates contour boundaries of each layer depending on the fiber diameter and fills the boundaries of each contour with continuous linear pattern with a given pore size, as shown in Figure 3.27. The script also keeps the G1 command locations for G-code generation. The nozzle of the machine moves in a straight line to the location

that G1 command specifies in the G-code. A sample G-code for a Fused Deposition Modeling (FDM) technique is given in the Figure 3.28.

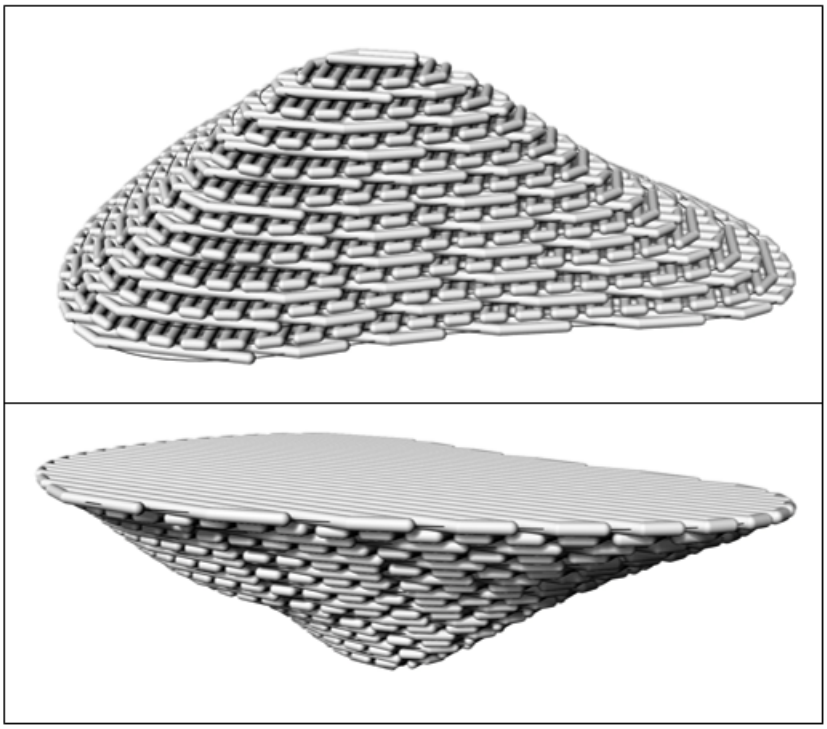

Figure 3.27: An example of resulting porous scaffold of a chronic wound

```
G90
M107
M104 S195
M551 P19200 S200
M543
M227 P3200 S3200
M106
G21
M103
M113 S1.0
M108 S400.0
M108 S600.0
M103
G1 Z0 F4200
G1 X5.6 Y-27.5 Z0 F4200
M101
G1 X5.6 Y-23.32 Z0 F1000
G1 X6.35 Y-20.39 Z0 F1000
G1 X6.35 Y-29.46 Z0 F1000
G1 X7.1 Y-30.92 Z0 F1000
G1 X7.1 Y-17.79 Z0 F1000
G1 X7.85 Y-15.64 Z0 F1000
G1 X7.85 Y-32.24 Z0 F1000
```

Figure 3.28: An example of G-code for deposition based additive manufacturing

Chapter 4

IMPLEMENTATION AND RESULTS

The proposed methodology is implemented in RhinoScript, which is a scripting tool of the Rhinoceros software based on Microsoft's VBScript language. The developed script asks user to select the cross-sections, point sampling distance and the values of w^n, the weight factor of the normal similarity, and w^d,the weight factor of the ruling line length. The proposed approach has been tested for various data sets of cross-sections. In this thesis, the data implemented is divided into there categories: implementation of the proposed surface reconstruction algorithm on artificial data of branched and skinned regions, and implementation on chronic wounds. In order to demonstrate the advantage and the quality of the proposed methodology, the results are compared with the results of two common approaches, non-rational cubic NURBS curve lofting and swept surface generation along a base-curve. These methods are readily available in the commercial software Rhinoceros 5.0.

4.1 Surface Reconstruction on Skinned Regions

First, the proposed skinned surface reconstruction algorithm, given in Algorithm 2 is applied to an artificial data with 3 cross-sections which have very similar shapes, shown in Figure 4.1. First of all, optimal set of ruling lines are found and redundant ruling lines are removed. Resulting set of ruling lines is shown in Figure 4.1. Then, a NURBS surface is fit through the contours and resulting set of ruling curves. The wireframe display of the resulting surface in Figure 4.2(a) and the stripe map showing the surface smoothness in Figure 4.2(b) verifies that the resulting surface is smooth and non-twisted, since there are no intersecting isolines and curvature is changing smoothly.

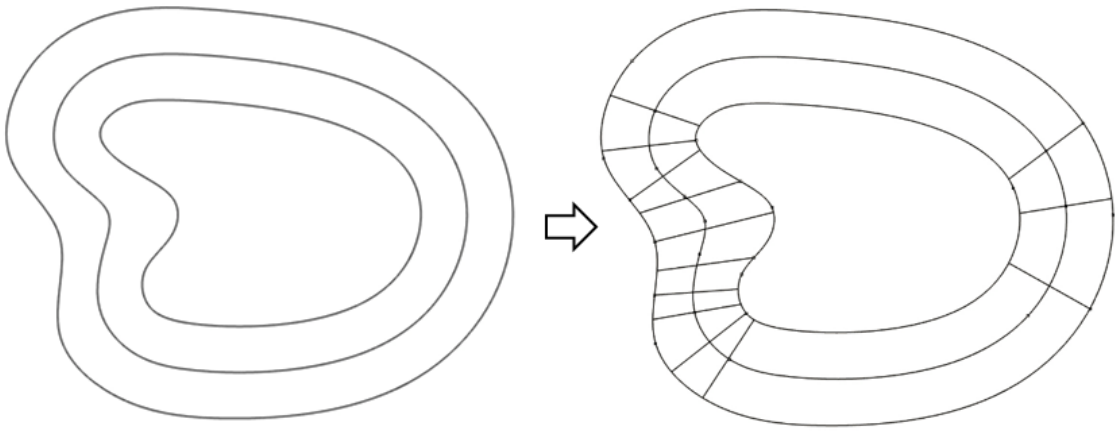

Figure 4.1: Finding maximum values ruling lines on smoothly changing cross-sections given on the left side

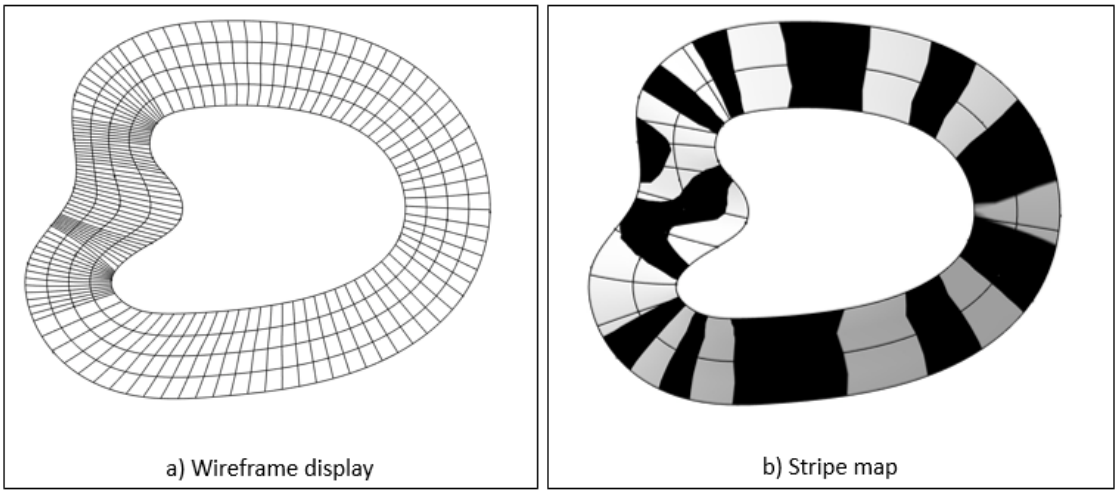

Figure 4.2: Continuity analysis of the resulting surface **(a)** wireframe display of the surface **(b)** stripe map indicating the smoothness of the surface

The resulting surface in Figure 4.3(a) is smoother than the surface generated by lofting, shown in Figure 4.3(b), and sweeping, in Figure 4.3(c). The proposed approach results in untwisted surface with smooth connection between cross-sections, while lofting and sweeping approaches cause unacceptably sudden changes as bumps and ripples. As in Figure 4.4(a), the resulting surface by our algorithm has the same Gaussian curvature through the surface, while the surfaces generated by lofting and sweeping have sudden changes in curvature. The Gaussian curvature is a product of two principal curvatures of the surface.

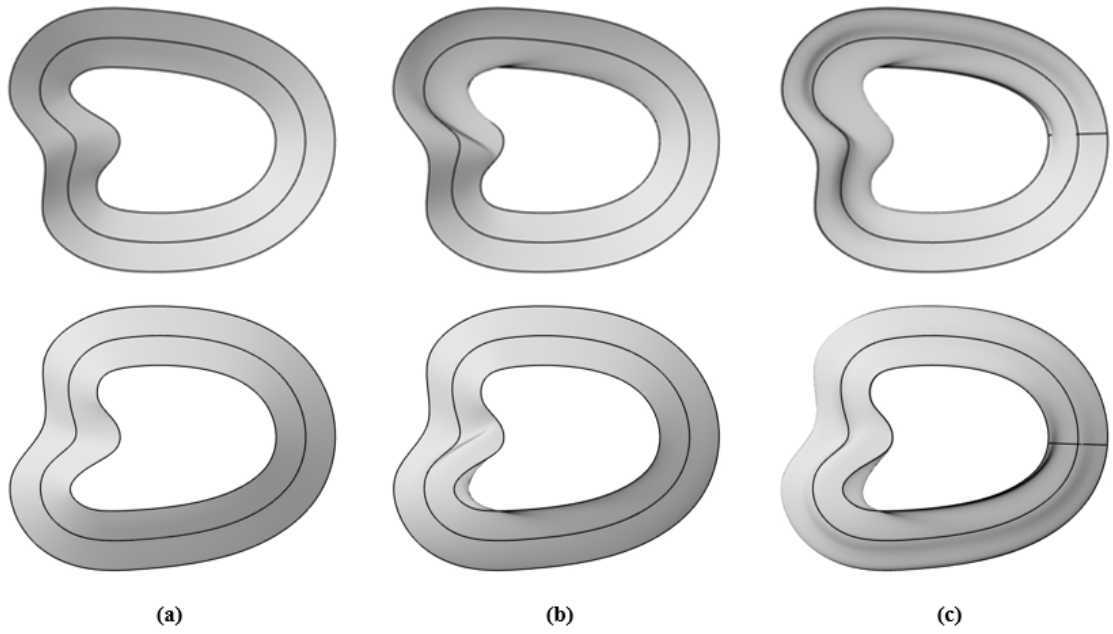

Figure 4.3: Skinned surface generation using **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve. Upper figures show top view and lower figures show bottom view of the surface.

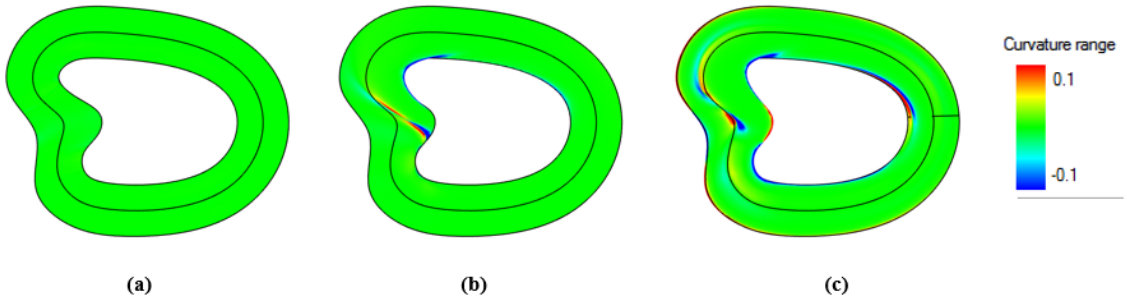

Figure 4.4: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve

Secondly, a NURBS surface is fit through 4 cross-sections where one of the contours with sharp corners. The proposed skinned surface generation algorithm is implemented as Figure 4.5 shows the input contours and maximum values ruling lines. Figure 4.6 indicates that the resulting surface is smooth non-twisted surface since the isolines do not intersect and Figure 4.7 shows smooth curvature change by a stripe map. Stripe maps visually evaluate surface smoothness and continuity using NURBS surface evaluation and rendering techniques to visually analyze the surface.

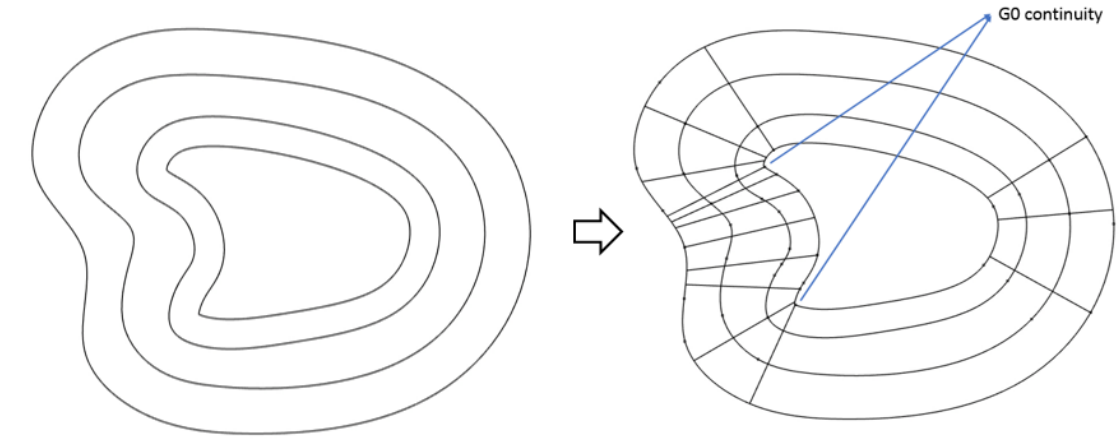

Figure 4.5: Finding maximum values ruling lines on cross-sections with sharp corners

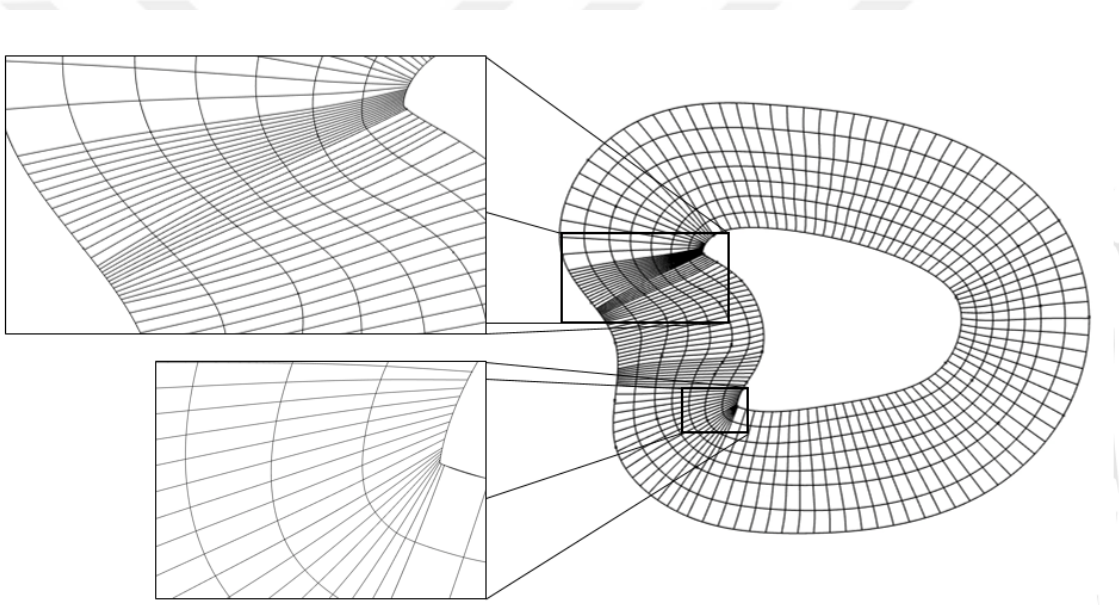

Figure 4.6: Isolines of the resulting surface in wireframe display

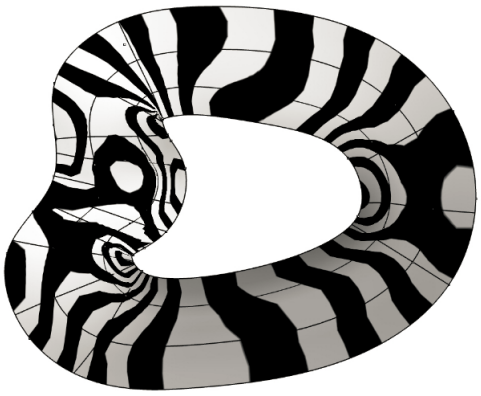

Figure 4.7: Continuity analysis of the resulting surface with a stripe map

Similarly, Figure 4.8(a) shows the proposed algorithm performs better than two common approaches: curve lofting and swept surface generation in terms of surface quality and smoothness. The surfaces in Figure 4.8(b) and Figure 4.8(c) are not compatible with CAD modeling for additive manufacturing. Also, Figure 4.9 suggests that the surface generated by our algorithm has less changes in Gaussian curvature compared to the surfaces generated by other methods.

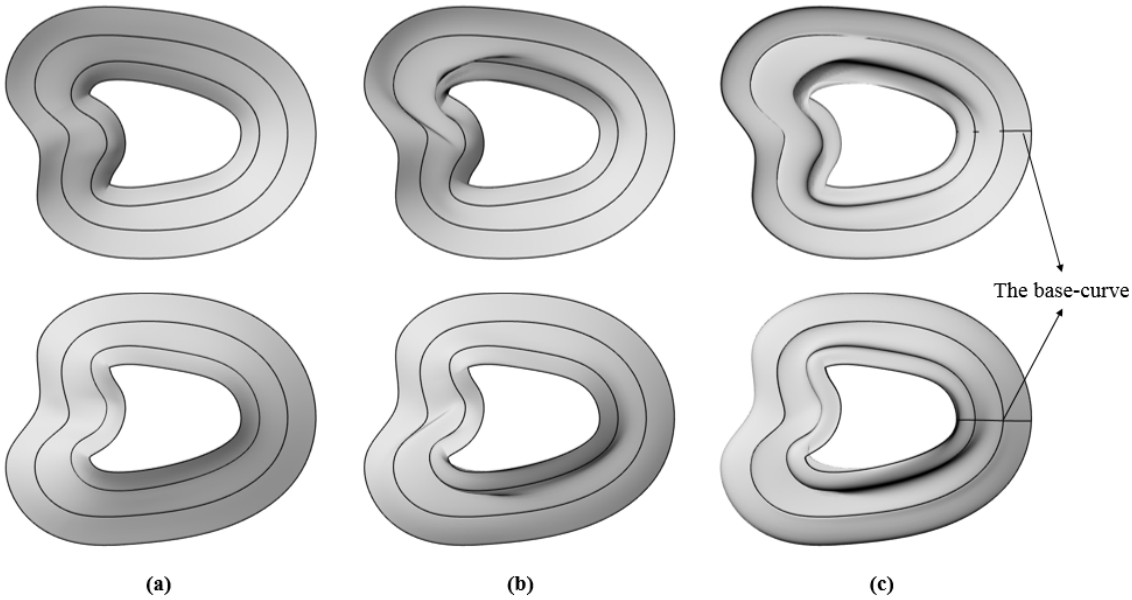

Figure 4.8: Skinned surface generation using **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve. Upper figures show top view and lower figures show bottom view of the surface.

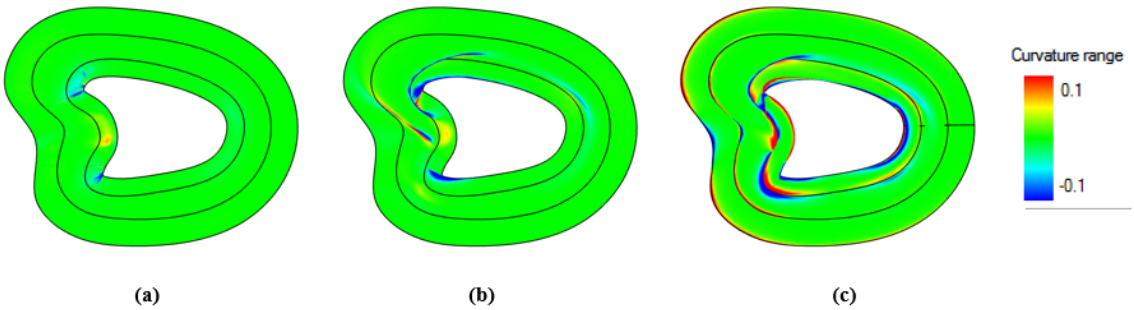

Figure 4.9: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve

Thirdly, the algorithm is tested on cross-sections with sudden changes in shape which is the case when a sparse data is available. Figure 4.10 shows the maximum valued ruling lines on inflection points to preserve the curvature information of each contour while

fitting a surface. Figure 4.11 indicates that the resulting surface is smooth and non-twisted despite of the sparse data of cross-sections.

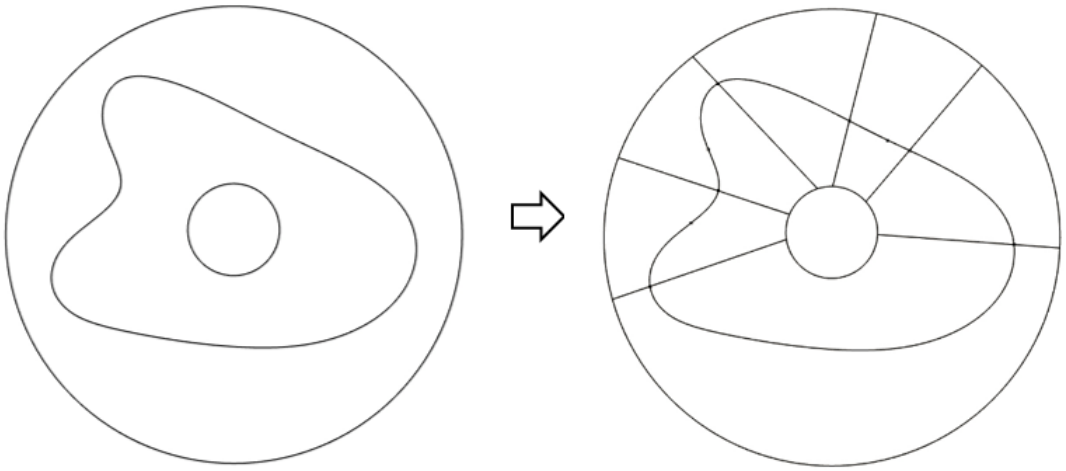

Figure 4.10: Finding maximum values ruling lines on cross-sections with sudden changes in shapes

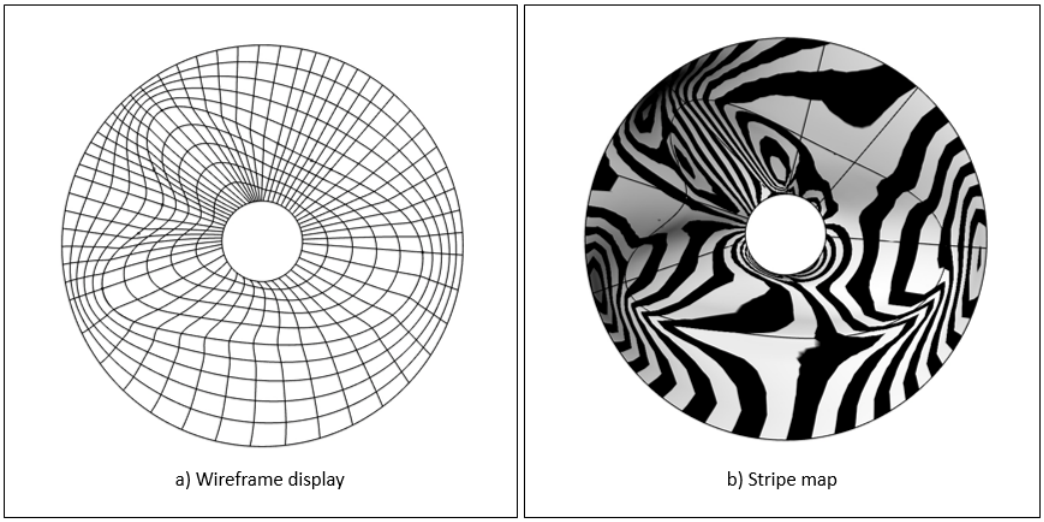

Figure 4.11: Continuity analysis of the resulting surface **(a)** wireframe display of the surface **(b)** stripe map indicating the smoothness of the surface

Figure 4.12(a) demonstrates the proposed algorithm can manage to provide smooth non-twisted surface even with a sparse cross-sections, while curve lofting and swept surface generation methods result in distorted surfaces as shown in Figure 4.12(b) and Figure 4.12(c). Figure 4.13 indicates that the surfaces generated by lofting and sweeping have instant changes in Gaussian curvature which is shown by color changing from blue to red suddenly.

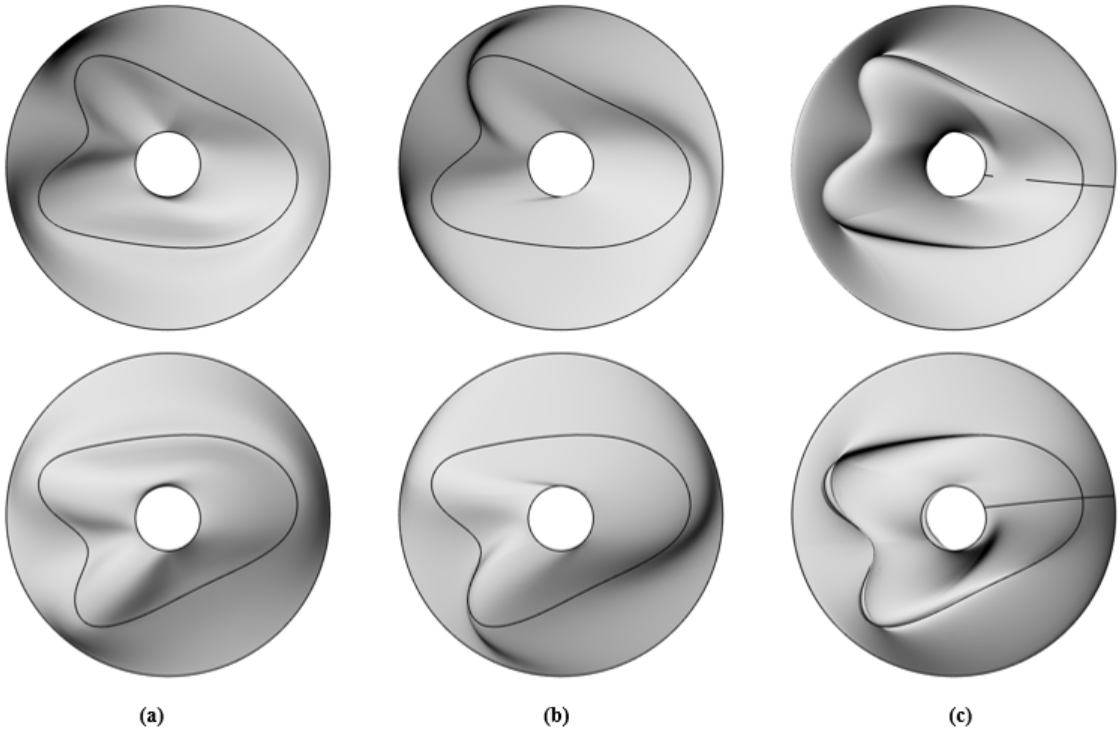

Figure 4.12: Skinned surface generation using **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve. Upper figures show top view and lower figures show bottom view of the surface.

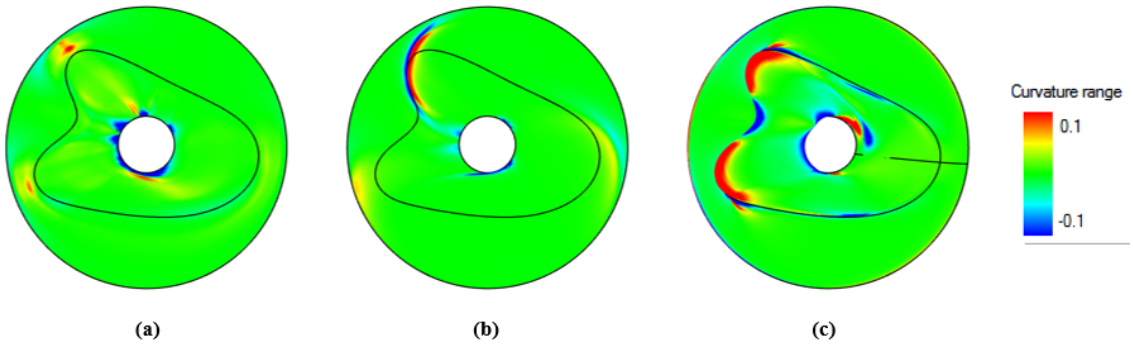

Figure 4.13: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve

Finally, the proposed algorithm is applied to the cross-sections with both sudden changes in shape and sharp corners. Similar to the previous applications, our algorithm generates better surface quality than curve lofting and swept surfaces, and it ensures global untwisting as shown in Figure 4.14. Figure 4.15(a) shows that the our algorithm generates surfaces with the same Gaussian curvature value at all points on the surface, while the surfaces in 4.15(b) and 4.15(c) have sudden curvature changes.

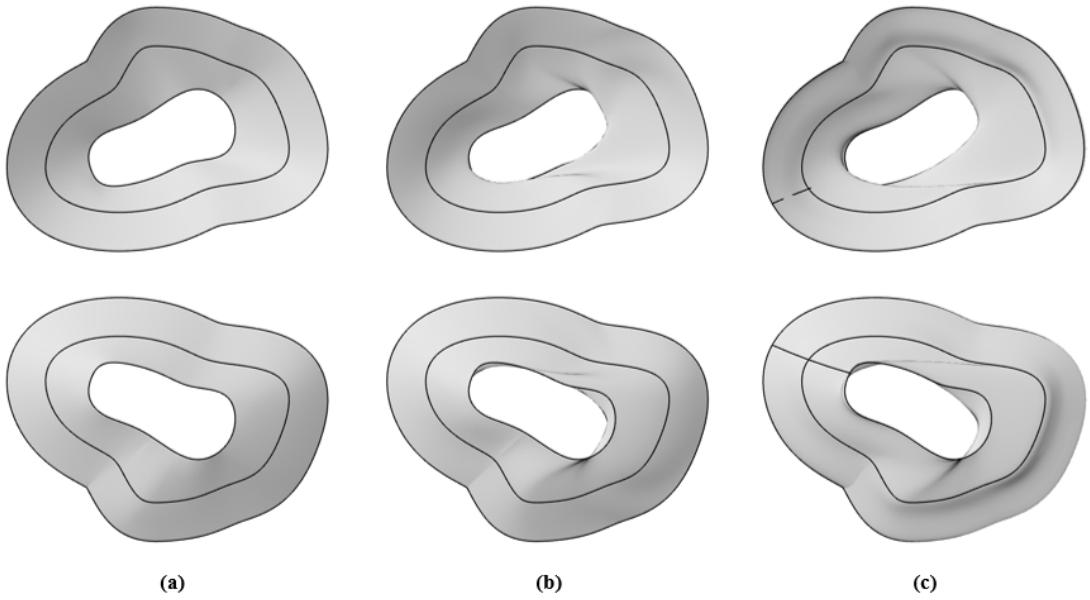

Figure 4.14: Skinned surface generation using **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve. Upper figures show top view and lower figures show bottom view of the surface.

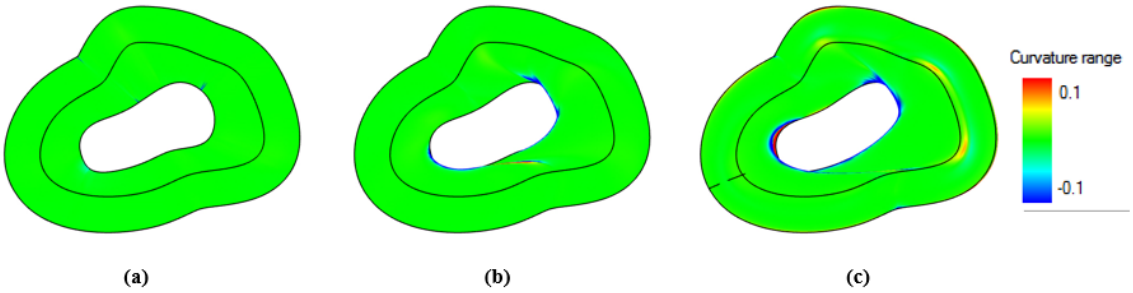

Figure 4.15: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve

4.2 Surface Reconstruction on Branched Regions

The proposed algorithm for branched regions are tested on cross-sections with 2, 3 and 4 branches. Resulting surfaces are smooth and non-self-intersecting surfaces on each Voronoi cell of each branch. These resulting surfaces are separate; not joined. Further operations can be applied to achieve joined surfaces; however, it is not necessary for tool path planning to be used in additive manufacturing.

Firstly, the methodology is tested on a branched region with 3 contour and 2 branches which are different in shape. Figure 4.16 shows the steps of surface construction for a branched region.

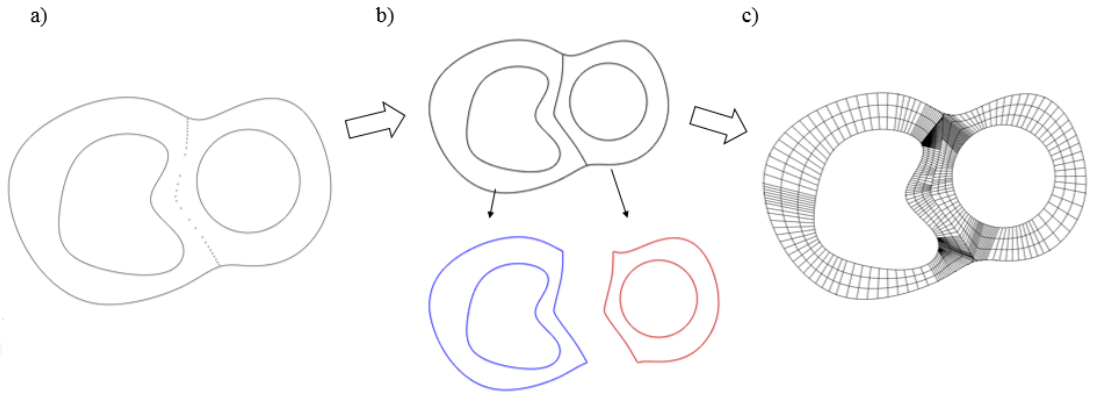

Figure 4.16: Depiction of surface reconstruction steps **(a)** Intersection points of the projected offset curves **(b)** Voronoi cells of each branch **(c)** Resulting surface on each Voronoi cell

Curve lofting which is one of the common approaches for surface fitting to given curves, cannot generate surface for branched regions. In order to compare the results of the proposed algorithm with the results of curve lofting, Voronoi cells are generated and the curves on each Voronoi cell are lofted. Figure 4.17 and Figure 4.18 suggests that the curve lofting approach results in ripples on the surface in contrast to the smooth non-twisted surface generated by the proposed approach.

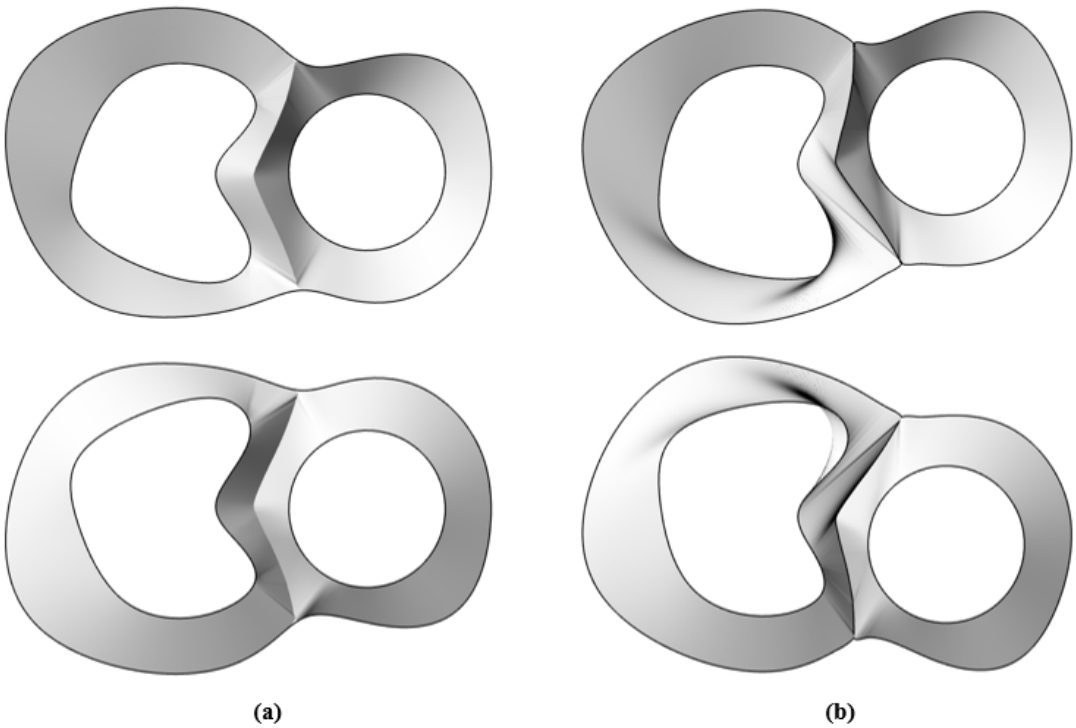

Figure 4.17: Branched surface generation using **(a)** the proposed approach and **(b)** NURBS curve lofting. Upper figures show top view and lower figures show bottom view of the surface.

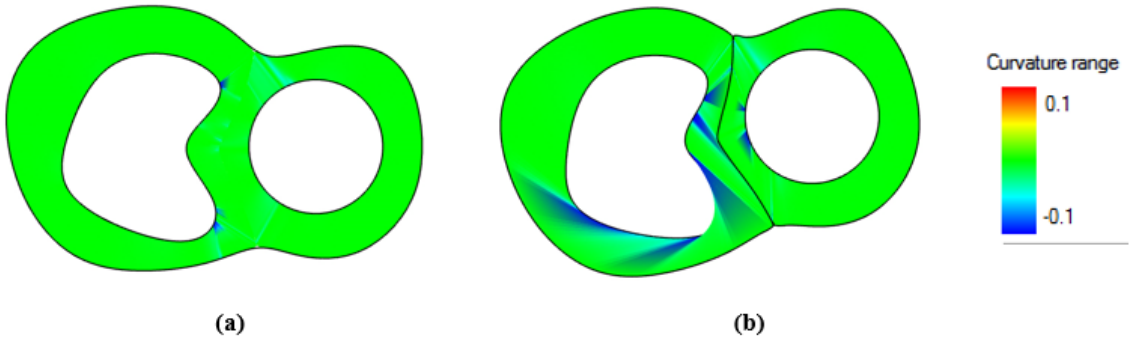

Figure 4.18: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach and **(b)** NURBS curve lofting

Secondly, it is tested on 14 contours with 3 skinned regions and 1 branched region to understand the quality of the methodology on a dense data. Figure 4.19(a)-(b) shows the steps of Voronoi cell generation on the upper curve of the branched region and the resulting branched surface can be seen in Figure 4.19(c).

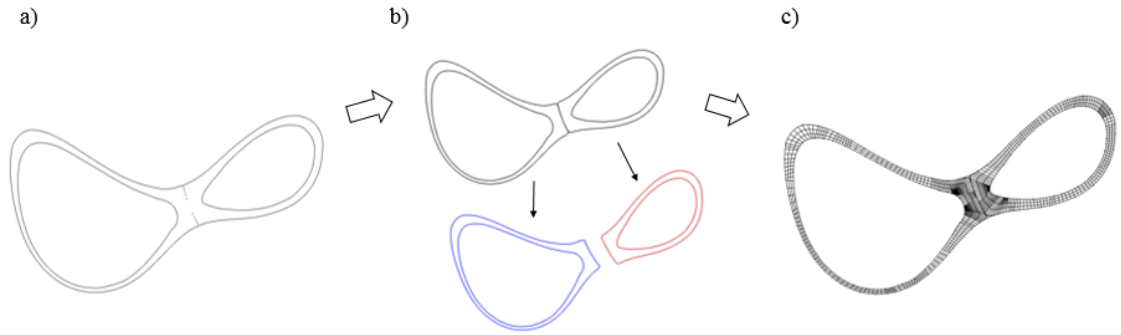

Figure 4.19: Depiction of surface reconstruction steps **(a)** Intersection points of the projected offset curves **(b)** Voronoi cells of each branch **(c)** Resulting surface on each Voronoi cell

The resulting hybrid surface generated by the proposed approach is compared with the results of curve lofting and swept surface generation. The result of the proposed approach is smoother and more accurate than the twisted surfaces generated by other two approaches, as shown in Figure 4.20 and Figure 4.21. Curve lofting performs well on the dense data of skinned regions, while it fails to generate smooth branched surfaces, as shown in Figure 4.20(b). In contrast, the swept surface generation approach results in twisted surfaces on both types of regions, as in Figure 4.20(c).

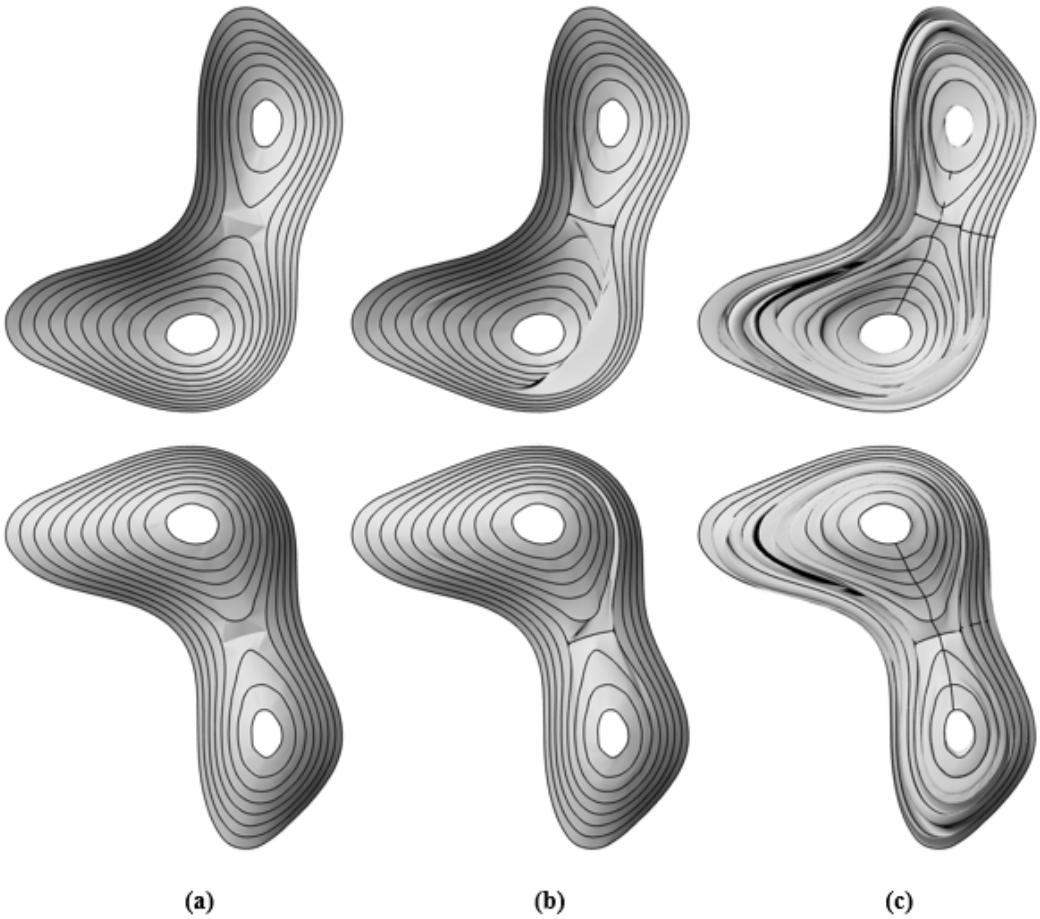

Figure 4.20: Hybrid surface generation using **(a)** the proposed approach, **(b)** NURBS curve lofting and **(c)** swept surface generation along a base-curve

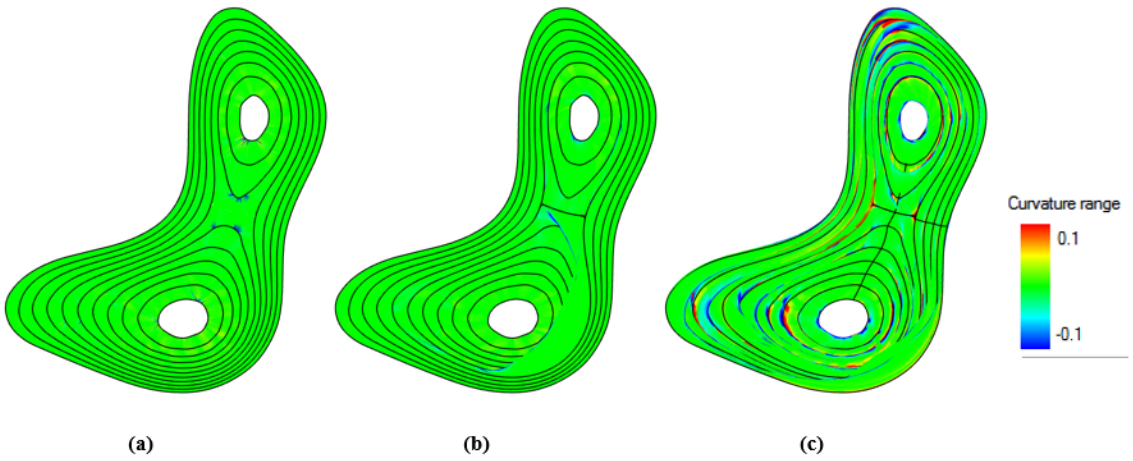

Figure 4.21: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach and **(b)** NURBS curve lofting

Thirdly, the proposed approach is tested on two set of cross-sections with 3 branches which vary in size and shape. Figure 4.22(a) and Figure 4.23(a) show the generated

Voronoi cells over the branches. It can be seen that the isolines of the resulting surfaces do not intersect in three-dimensions in Figure 4.22(b) and Figure 4.23(b). The generated surfaces are non-twisted compatible surfaces, as shown in Figure 4.22(c) and Figure 4.23(c); however, the smoothness of the surface in Figure 4.23(c) is poor because of the disparate branch contours.

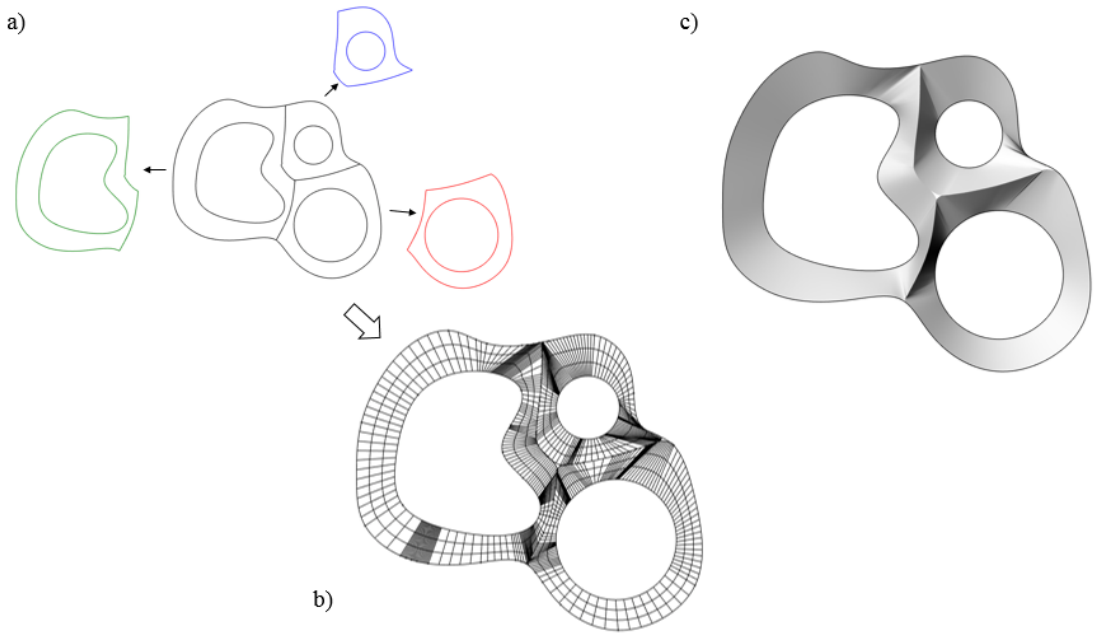

Figure 4.22: Depiction of surface reconstruction steps **(a)** Voronoi cells of each branch **(b)** Isolines of the resulting surface in wireframe display **(c)** Resulting branched surface

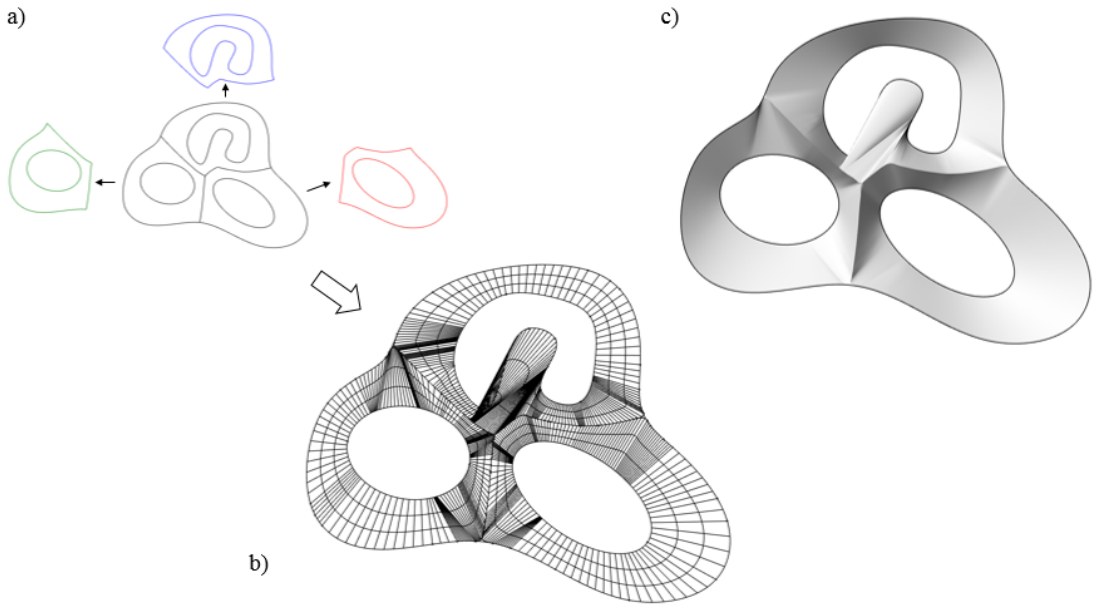

Figure 4.23: Depiction of surface reconstruction steps **(a)** Voronoi cells of each branch **(b)** Isolines of the resulting surface in wireframe display **(c)** Resulting branched surface

The resulting branched surface is compared with the curve lofting approach and the proposed algorithm generated smoother and non-self-intersecting surface than the lofting approach as shown in Figure 4.24 and 4.25.

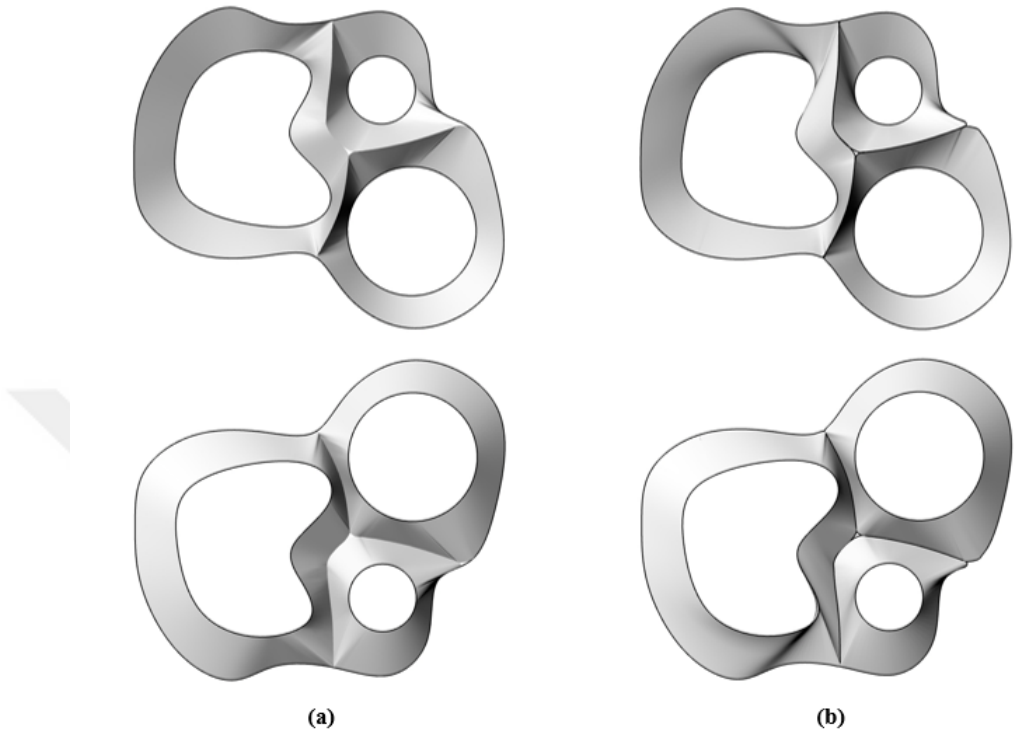

Figure 4.24: Hybrid surface generation using **(a)** the proposed approach and **(b)** NURBS curve lofting

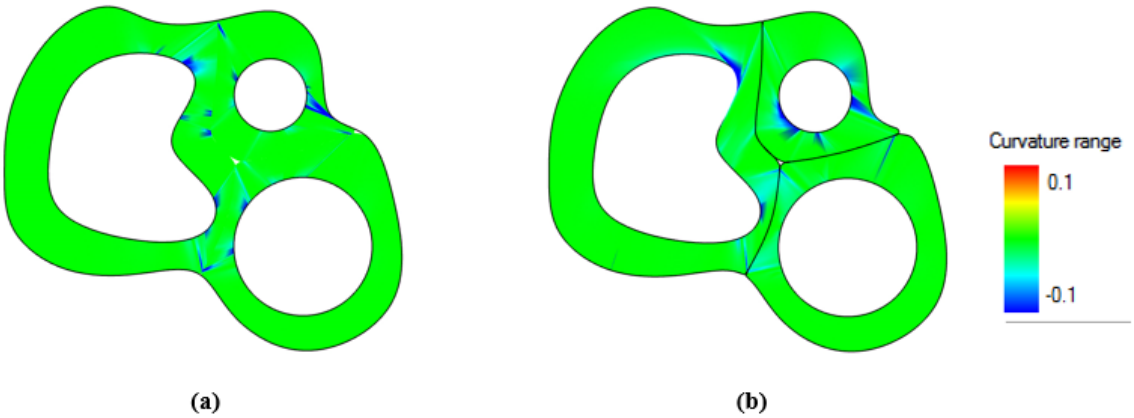

Figure 4.25: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach and **(b)** NURBS curve lofting

Also for the second set of cross-sections, the surface achieved by our algorithm, Figure 4.26(a) is non-twisted and smooth, in contrast to the distorted and self-intersecting surface

generated by curve lofting, Figure 4.26(b). In addition, Figure 4.27 indicates that the resulting surface of our algorithm has less curvature change through the surface compared to the lofted surface.

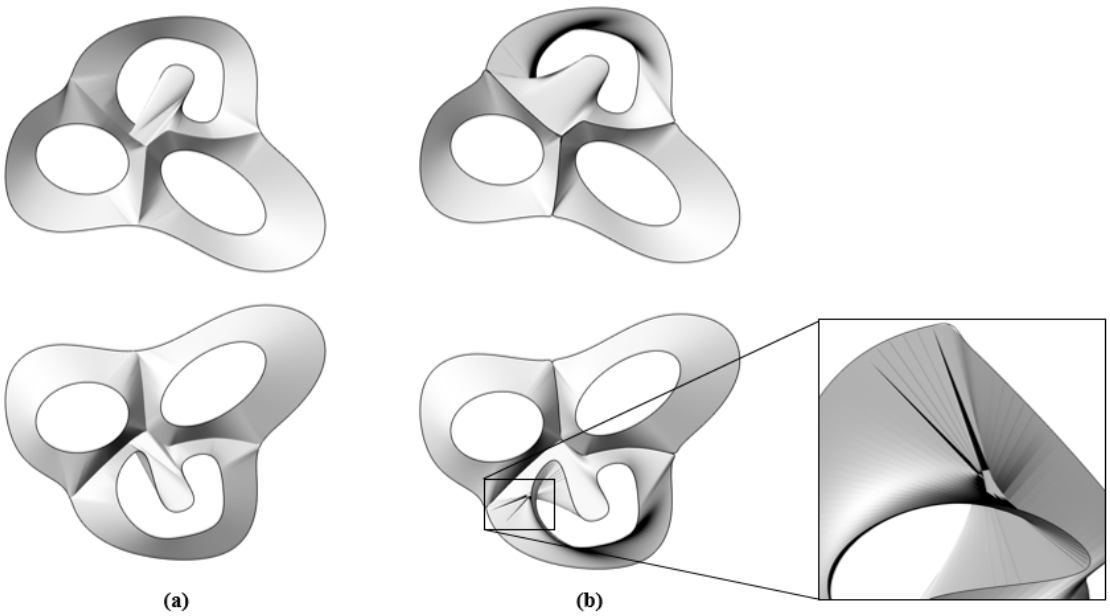

Figure 4.26: Hybrid surface generation using **(a)** the proposed approach and **(b)** NURBS curve lofting

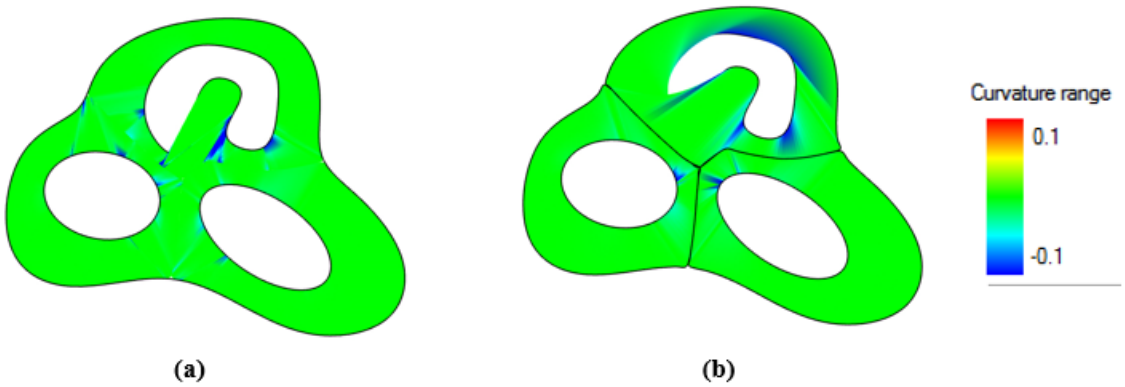

Figure 4.27: Gaussian curvature analysis of the surface generated by **(a)** the proposed approach and **(b)** NURBS curve lofting

The proposed approach generates visually pleasing, non-twisted surfaces on branched regions regardless of the shape and size difference in branch contours. It performs better than the lofted and swept surface generation methods. The implementation has been done using up to 4 branches; however, once the Voronoi cells based on each branch contour is obtained using several Voronoi generation algorithms, the proposed approach can be applied to any branched region.

4.3 Implementation on Chronic Wounds

4.3.1 Surface Fitting

The proposed methodology is applied to the medical images of a pressure ulcer, as shown in Figure 4.28(a). First of all, cross-sectional data is achieved by generating a point representing each pixel of the image by using ImageJ software. X and Y coordinate values of the pixel is saved as X and Y coordinate values of the point and the color intensity of the image is taken as Z coordinate value of the point, since it is assumed that the lighter colored areas of the image is located near to the camera. By curve fitting through the point cloud, approximated cross-sections of the wound is achieved as shown in Figure 4.28(b). Although, the cross-sections are approximated from single 2D image in this study, the developed surface reconstruction algorithm can be applied to any cross-sectional data of a chronic wound. More accurate cross-sectional data can be achieved by using 3D scanners or depth sensing cameras.

Once cross-sections of the wound is achieved, the proposed surface reconstruction algorithm is applied to achieve a CAD model of the wound geometry. The resulting surface is a smooth and non-twisted surface, as shown in Figure 4.29.

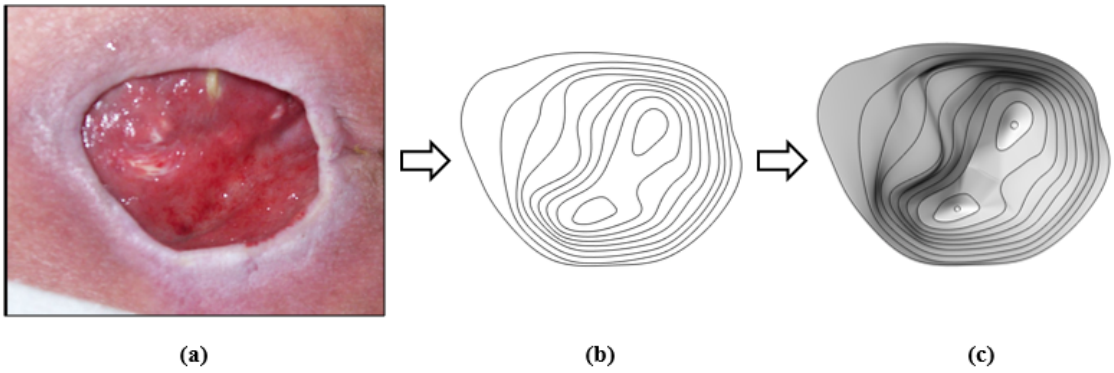

Figure 4.28: Application to the cross-section data of the pressure ulcer **(a)** the wound image **(b)** the cross-sections of the wound **(c)** the surface approximated by the proposed approach

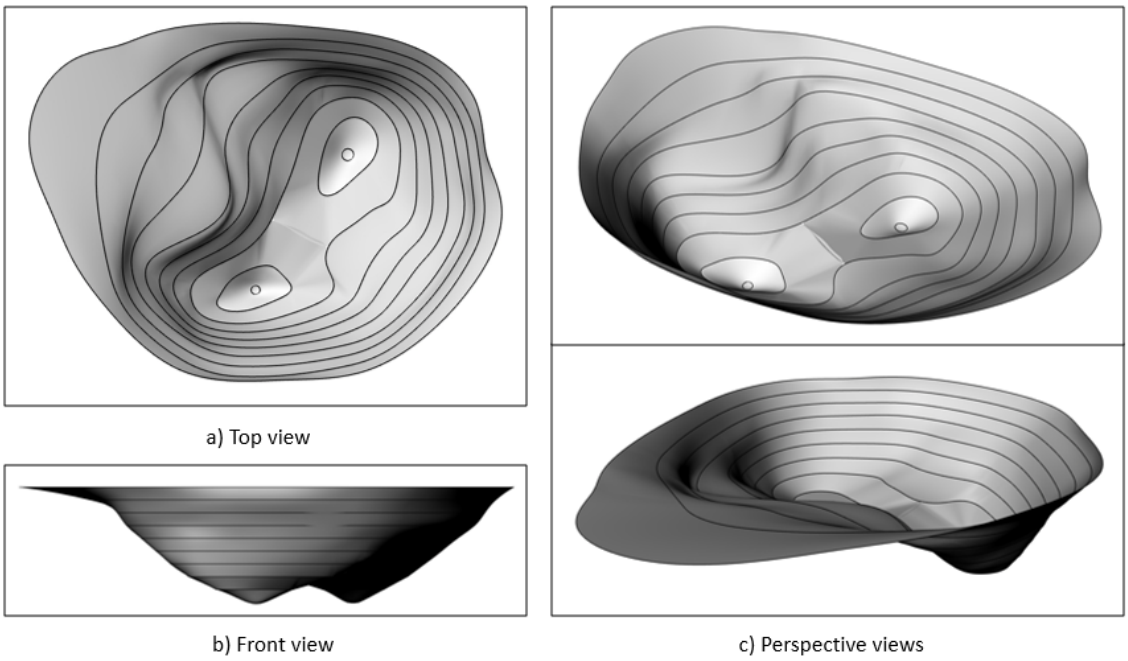

Figure 4.29: Application to the cross-section data of the pressure ulcer **(a)** the wound image **(b)** the cross-sections of the wound **(c)** the surface approximated by the proposed approach

To obtain the resulting surface in Figure 4.29, the developed algorithm is performed on a 3.40 GHz CPU in approximately 2 s for 10 curves and a total of 191 curvature points.

Similar procedure is applied to another chronic wound image (Figure 4.30(a)). After processing the image by ImageJ software, 3D cross-sections are extracted as in Figure 4.30(b). Then, a smooth surface is approximated over the obtained planar curves by the proposed surface reconstruction algorithm. The resulting smooth and non-self-intersecting surface can be seen in Figure 4.31.

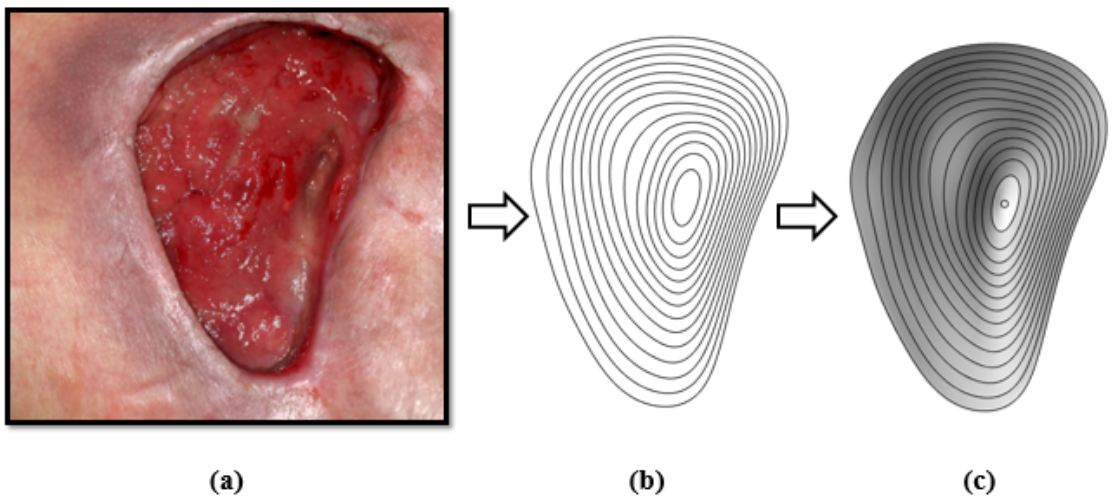

Figure 4.30: Application to the cross-section data of the diabetic ulcer **(a)** the wound image **(b)** the cross-sections of the wound **(c)** the surface approximated by the proposed approach

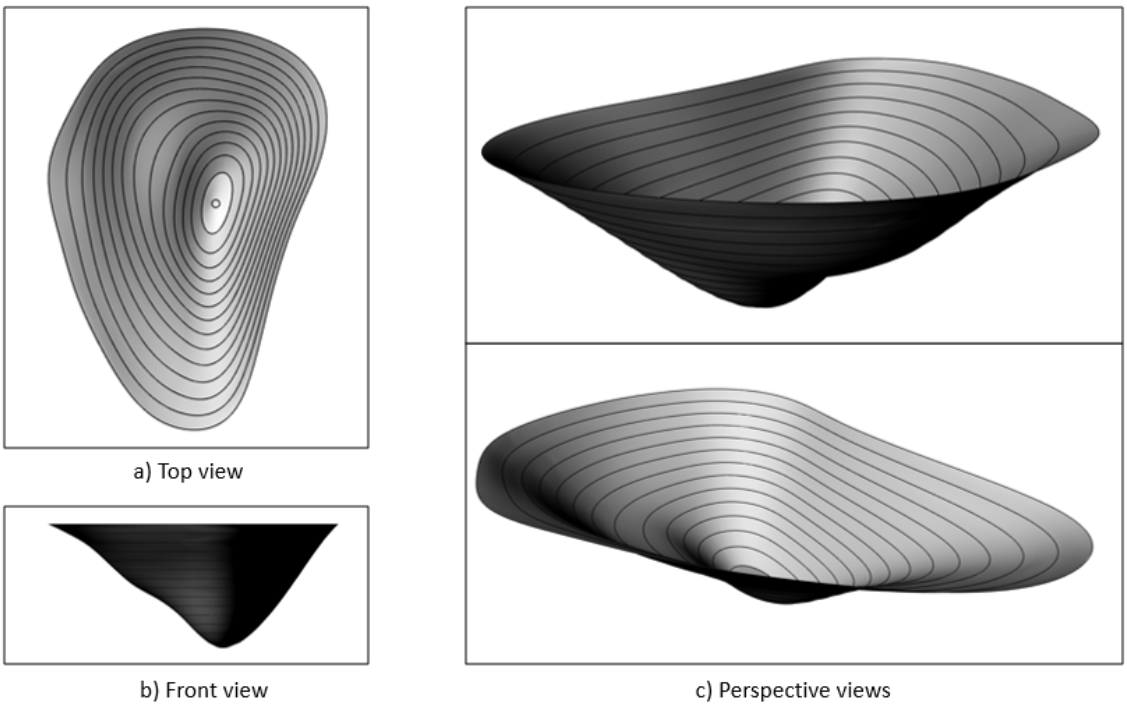

Figure 4.31: Application to the cross-section data of the pressure ulcer **(a)** the wound image **(b)** the cross-sections of the wound **(c)** the surface approximated by the proposed approach

The surface in Figure 4.31 is generated in 3.30 s for 44 ruling for 14 curves and a total of 67 curvature points.

4.3.2 Toolpath Planning for Bioprinting

A porous scaffold model can be generated on the approximated surface by using the developed VBScript code on Rhinoceros 5.0 software. A nozzle diameter of 0.5 mm and pore size of 0.25 mm are selected for the wound shown in Figure 4.28 with the longest length of 20.3 mm, greatest width of 15.4 mm, and greatest depth of 4.8 mm. Figure 4.32(d) and Figure 4.33 shows the porous structure generated by the developed script which also generates the G-code for deposition based additive manufacturing by keeping the start and end points of deposited cylindrical fibers.

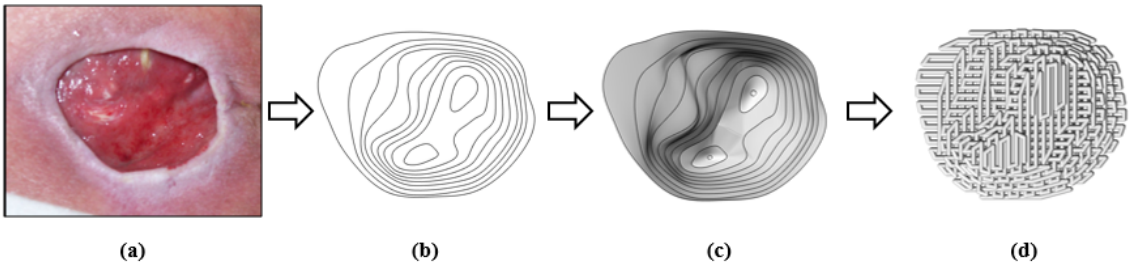

Figure 4.32: Steps of porous scaffold generation **(a)** medical imaging shown in 4.28 **(b)** cross-sectional data generation **(c)** surface reconstruction using the proposed algorithm **(d)** porous scaffold model from the bottom view

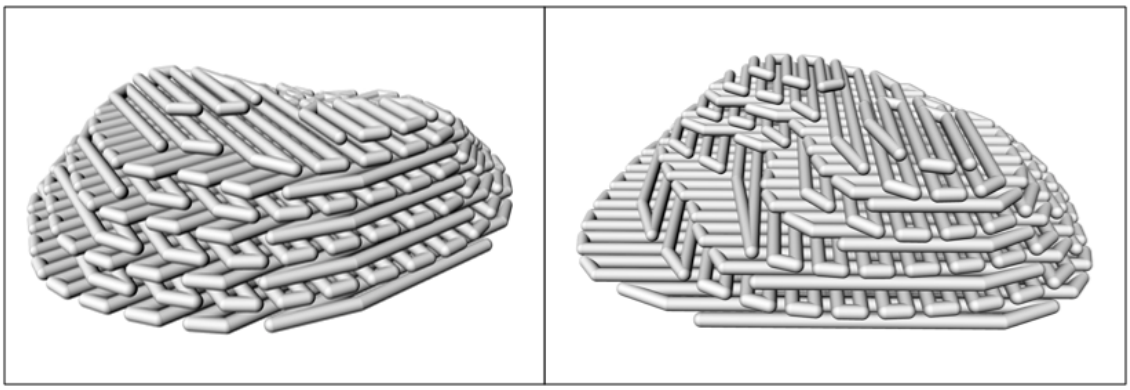

Figure 4.33: The porous scaffold from different view points

Similar procedure is applied to the wound image shown in Figure 4.30 and the steps of the procedure is shown in Figure 4.34. The porous scaffold generated by the developed script can be seen in Figure 4.35.

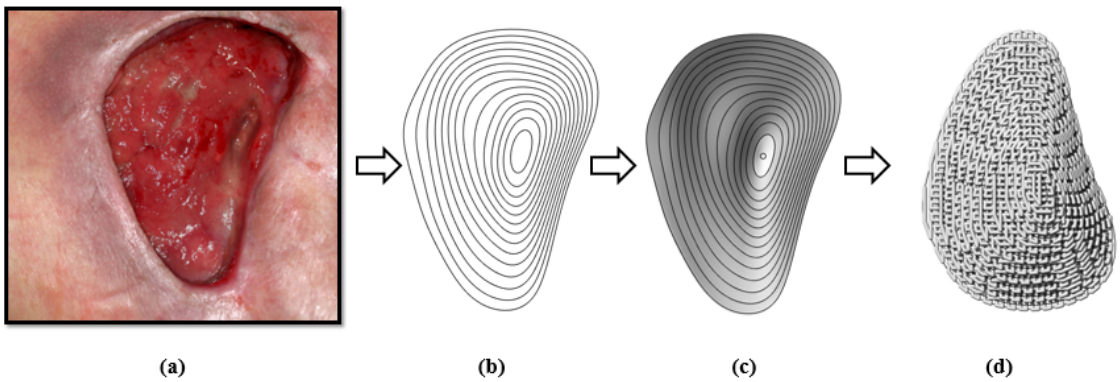

Figure 4.34: Steps of porous scaffold generation **(a)** medical imaging shown in 4.30 **(b)** cross-sectional data generation **(c)** surface reconstruction using the proposed algorithm **(d)** porous scaffold model from the bottom view

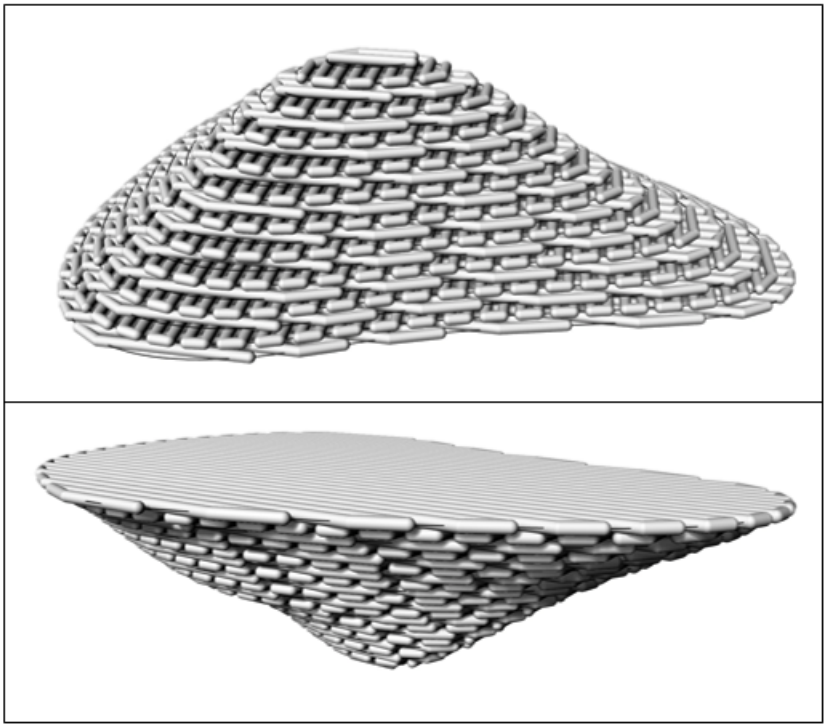

Figure 4.35: The porous scaffold from different view points

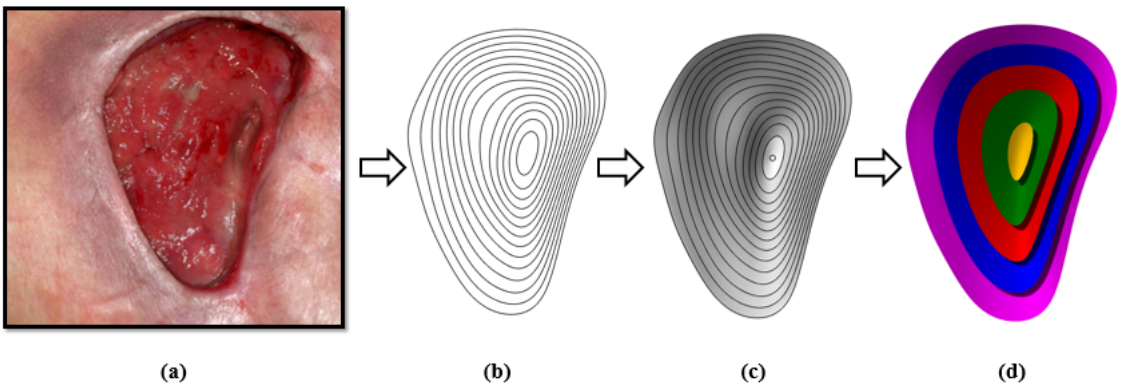

Figure 4.36: Application to the cross-section data of the chronic wound shown in 4.30 **(a)** the chronic wound image from the literature [5] **(b)** the cross-sections of the wound **(c)** the surface approximated by the proposed approach **(d)** the uniform healing region boundaries

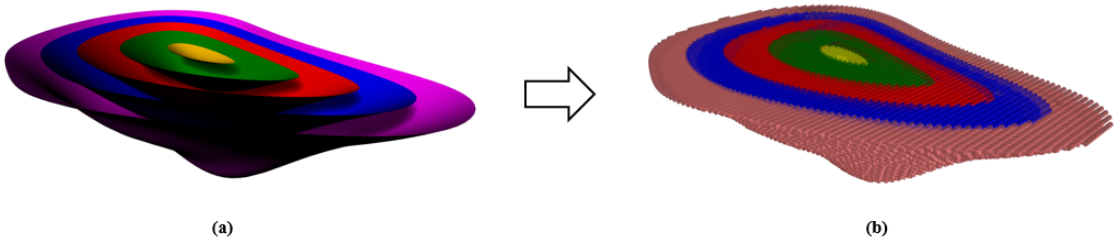

Figure 4.37: Modeling of 3D wound scaffold **(a)** the uniform healing region boundaries **(b)** the functionally gradient uniform region of the scaffold

In addition to tool path planning for single-material porous scaffolds, the approximated surface by our algorithm can be used to model multi-material porous scaffolds.In order to improve healing process for chronic wounds, the methodology proposed by Ozbolat and Koc [3] can be applied on the approximated surface to generate multi-functional porous wound scaffolds. Firstly, the uniform healing region boundaries are generated by using the surface generated by our algorithm, as depicted in Figure 4.36. Then, different bio-material and bio-molecule concentration are assigned to each region, Figure 4.36(d). After that, the porous scaffold is generated based on healing regions with different materials as shown in Figure 4.37. While 3D printing the scaffold, number of materials that are used on each layer changes based on the healing regions covered, Figure 4.38. The multi-material porous scaffold of a patient-specific wound can be fabricated by using multi-chamber single nozzle deposition system [3, 93, 94].

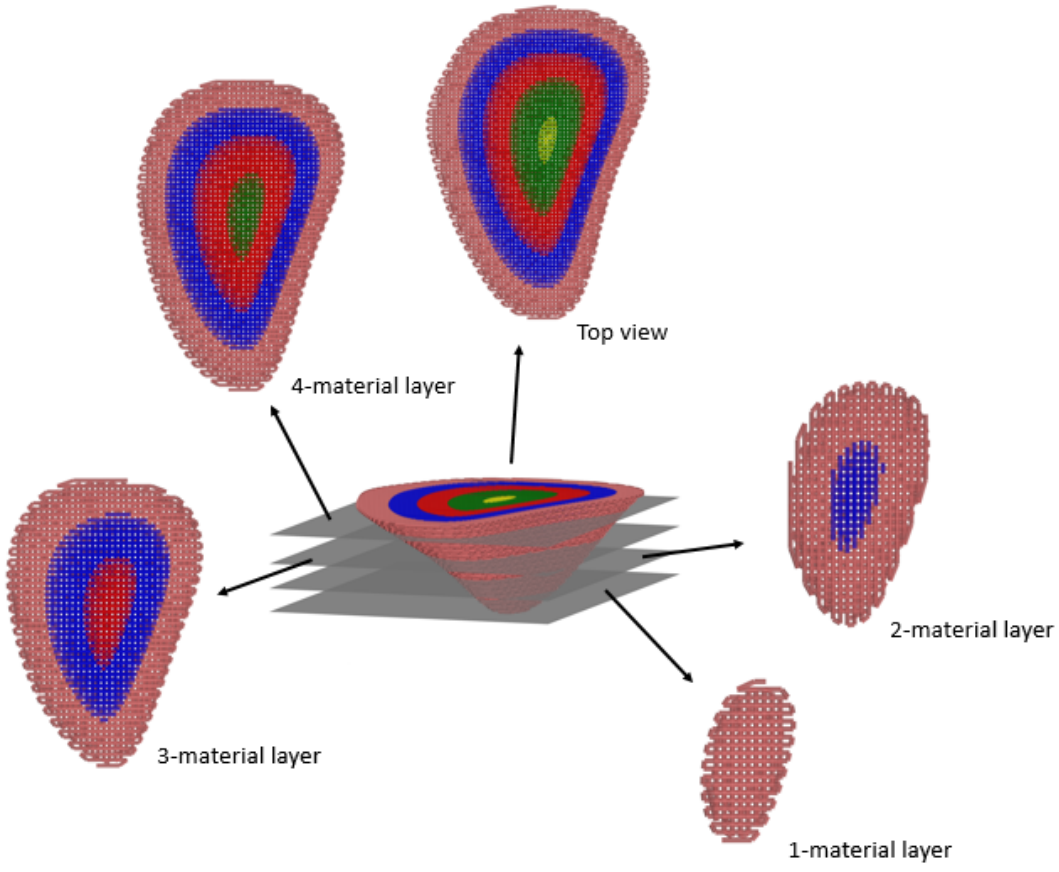

Figure 4.38: Multi-material 3D printing layers of the wound scaffold

We can investigate that the proposed approach in this thesis provides better solutions for CAD modeling of complex shaped wounds, compared to commonly used two methods: NURBS curve lofting and generating swept surface along a base-curve. In addition,

the proposed algorithm can be applied to any set of contours which satisfies the assumption of higher cross-sections involve the lower ones; generally this is case when the outer surface of the object is embedded in another object and cannot be measured. There is a trade off between the number of cross-sections and computational time. As the cross-section numbers increase, the algorithm results in a better surface that reflects all significant properties of the object, but the computational time increases. However, the proposed approach results in good solutions when it is applied to sparse cross-sections with sudden changes in shapes. It also approximates smooth and non-twisted surfaces on the contours with sharp edges and with multiple branches. In addition, a porous scaffold and tool path planning for specified nozzle diameter and pore size are automatically generated by the developed script.

Chapter 5

CONCLUSION AND FUTURE WORK

In this thesis, a new method is proposed to generate a 3D wound surface from planar cross-sections to be used in wound measurement and patient-specific porous scaffold generation for 3D bioprinting. Triangular surface-based approach is commonly used for surface generation from planar contours, which is obtained by stitching triangular patches by connecting the given cross-sections. However, this approach is time consuming and required much storage caused by large number of triangles and it is ineffective due to a lack of topological information. The proposed method generates smooth and non-twisted NURBS surfaces from a set of planar cross-sections satisfying the assumption of higher cross-sections involves the area bounded by lower cross-sections. The input cross-sections can be obtained by 3D scanning techniques or depth sensing cameras. Also, cross-sectional data can be achieved from a point cloud by curve fitting and from the mesh of the object by slicing in certain intervals. The resulting composite surface consists of three kinds of surfaces: skinned, branched and capped.

For the skinned region, the proposed method employs similar objective function and constraints with the *GRLC* method developed by Samanta et al. [83]. While *GRLC* method only works for blending two curves with C^1 continuity, our method generates a 3D surface through cross-sections allowing sudden changes between cross-sections and sharp corners on contours. The proposed method also works with branched cross-sections by using a Voronoi diagram based approach.

The algorithm provides more visually pleasing surfaces for complex wound geometries than curve lofting and sweeping along a base-curve approaches. The proposed method deserves particular attention in multi-functional porous wound scaffold modeling. Also, there are several other application areas to use this method: multi-axis NC machining, morphing and shape blending, toolpath planning for 3D printing, and smooth

transition between curves for animation. Further investigations can be done to remove the assumption of non-intersecting cross-sections on top view, to apply the proposed method to more complex shaped objects.

In addition, even though there are several bioprinting path generator software for 3D printing, there is a need for G-code generator for porous scaffolds which are generated specific to the nozzle diameter and desired porosity. In order to automatically form the porous scaffolds and to generate the tool path, another script is developed in VBScript language using Rhinoceros for porous scaffold modeling and toolpath planning on a given surface. According to entered nozzle diameter and pore size, the script automatically creates a porous scaffold model and a G-code for deposition based additive manufacturing processes.

The future work should focus on improving the surface reconstruction algorithm to apply on cross-sectional data of any biological object, the application area should not be limited to wound modeling. To do so, the assumption of non-intersecting projected contours should be removed. Also, the resulting ruling curved may differ depending on the start point, therefore the ideal way is to start from different curvature points and select the one with maximum objective function value.

In addition, the tool path can be optimized for multi-material bioprinting to generate continuous tool path for each material. Therefore, each healing region can be printed at once by using the same material. This approach will save time by avoiding the time spent on material change.

Bibliography

[1] Undestanding the anatomy. http://headandneckcancerguide.org/adults/introduction-to-head-and-neck-cancer/skin-cancer/anatomy/, note = Accessed: 2017-07-21, .

[2] Wound healing process. http://www.newhealthadvisor.com/wound-healing-process.html, . Accessed: 2017-07-21.

[3] Ibrahim T Ozbolat and Bahattin Koc. Multi-function based modeling of 3d heterogeneous wound scaffolds for improved wound healing. *Computer-aided Design and Applications*, 8(1):43–57, 2011.

[4] Wei Long Ng, Shuai Wang, Wai Yee Yeong, and May Win Naing. Skin bioprinting: impending reality or fantasy? *Trends in biotechnology*, 34(9):689–699, 2016.

[5] Ibrahim T Ozbolat and Bahattin Koc. 3d hybrid wound devices for spatiotemporally controlled release kinetics. *Computer methods and programs in biomedicine*, 108 (3):922–931, 2012.

[6] World Health Organization et al. *Global report on diabetes*. World Health Organization, 2016.

[7] Urban Pavlovčič, Janez Diaci, Janez Možina, and Matija Jezeršek. Wound perimeter, area, and volume measurement based on laser 3d and color acquisition. *Biomedical engineering online*, 14(1):39, 2015.

[8] Cheryl Majeske. Reliability of wound surface area measurements. *Physical Therapy*, 72(2):138–141, 1992.

[9] CJK Bulstrode, AW Goode, and PJ Scott. Stereophotogrammetry for measuring rates of cutaneous healing: a comparison with conventional techniques. *Clinical Science*, 71(4):437–443, 1986.

[10] Alessandro Melo Deana, Sérgio Henrique Costa Jesus, Brunna Pileggi Azevedo Sampaio, Marcelo Tavares Oliveira, Daniela Fátima Teixeira Silva, and Cristiane Miranda França. Fully automated algorithm for wound surface area assessment. *Wound Repair and Regeneration*, 21(5):755–761, 2013.

[11] Habiba A Thawer, Pamela E Houghton, M Gail Woodbury, David Keast, and Karen Campbell. Computer-assisted and manual wound size measurement. *Ostomy/wound management*, 48(10):46–53, 2002.

[12] Nada Kecelj-Leskovec, Matija Jezeršek, Janez Možina, Miloš D Pavlović, and Tomaž Lunder. Measurement of venous leg ulcers with a laser-based three-dimensional method: Comparison to computer planimetry with photography. *Wound repair and regeneration*, 15(5):767–771, 2007.

[13] Line Bisgaard Jørgensen, Jens A Sørensen, Gregor BE Jemec, and Knud B Yderstræde. Methods to assess area and volume of wounds–a systematic review. *International wound journal*, 2015.

[14] David Meyers, Shelley Skinner, and Kenneth Sloan. Surfaces from contours. *ACM Transactions On Graphics (TOG)*, 11(3):228–258, 1992.

[15] Hyungjun Park. A hybrid approach to smooth surface reconstruction from 2-d cross sections. *The International Journal of Advanced Manufacturing Technology*, 25 (11):1130–1136, 2005.

[16] A Catharine Ross, JoAnn E Manson, Steven A Abrams, John F Aloia, Patsy M Brannon, Steven K Clinton, Ramon A Durazo-Arvizu, J Christopher Gallagher, Richard L Gallo, Glenville Jones, et al. The 2011 report on dietary reference intakes for calcium and vitamin d from the institute of medicine: what clinicians need to know. *The Journal of Clinical Endocrinology & Metabolism*, 96(1):53–58, 2011.

[17] Ann Cranney, Tanya Horsley, Siobhan O'Donnell, Hope Weiler, Lorri Puil, Daylily Ooi, Stephanie Atkinson, Leanne Ward, David Moher, David Hanley, et al. Effectiveness and safety of vitamin d in relation to bone health. *Evidence report/technology assessment*, (158):1, 2007.

[18] Richard AF Clark, Kaustabh Ghosh, and Marcia G Tonnesen. Tissue engineering for cutaneous wounds. *Journal of Investigative Dermatology*, 127(5):1018–1029, 2007.

[19] Robert Lanza, Robert Langer, and Joseph P Vacanti. *Principles of tissue engineering*. Academic press, 2011.

[20] American Burn Association et al. Burn incidence and treatment in the us: 2000 fact sheet. *Chicago: ABA*, 2000.

[21] Michael D Peck. Epidemiology of burns throughout the world. part i: Distribution and risk factors. *Burns*, 37(7):1087–1100, 2011.

[22] Centers for Disease Control, Prevention, et al. National diabetes fact sheet: general information and national estimates on diabetes in the united states, 2000. *Atlanta, GA: US Department of Health and Human Services, Centers for Disease Control and Prevention*, 2002.

[23] M Ehrenreich and Z Ruszczak. Update on tissue-engineered biological dressings. *Tissue engineering*, 12(9):2407–2424, 2006.

[24] Dorothy M Supp and Steven T Boyce. Engineered skin substitutes: practices and potentials. *Clinics in dermatology*, 23(4):403–412, 2005.

[25] Jonathan A Sherratt and John C Dallon. Theoretical models of wound healing: past successes and future challenges. *Comptes Rendus Biologies*, 325(5):557–564, 2002.

[26] Jack C Fisher. Amniotic membranes as a temporary wound dressing. *Plastic and Reconstructive Surgery*, 52(5):601, 1973.

[27] Charles Gokoo and Kenneth Burhop. A comparative study of wound dressings on full-thickness wounds in micropigs. *Advances in Skin & Wound Care*, 6(5):42–55, 1993.

[28] Mary Babu et al. Collagen based dressings—a review. *Burns*, 26(1):54–62, 2000.

[29] JG Rheinwald and H Green. Serial cultivation of strains of human epidermal keratinocytes in defined clonal and serum-free culture. *J Invest Dermatol*, 6:331–42, 1975.

[30] Howard Green, Olaniyi Kehinde, and Judith Thomas. Growth of cultured human epidermal cells into multiple epithelia suitable for grafting. *Proceedings of the National Academy of Sciences*, 76(11):5665–5668, 1979.

[31] Loring W Rue, William G Cioffi, William F McManus, and Basil A Pruitt. Wound closure and outcome in extensively burned patients treated with cultured autologous keratinocytes. Technical report, ARMY INST OF SURGICAL RESEARCH FORT SAMHOUSTON TX, 1993.

[32] Andrea Andreassi, Roberta Bilenchi, Maurizio Biagioli, and Carlo D'Aniello. Classification and pathophysiology of skin grafts. *Clinics in dermatology*, 23(4):332–337, 2005.

[33] Robert Sheridan. Closure of the excised burn wound: autografts, semipermanent skin substitutes, and permanent skin substitutes. *Clinics in plastic surgery*, 36(4): 643–651, 2009.

[34] Charles Cuono, Robert Langdon, and Joseph McGuire. Use of cultured epidermal autografts and dermal allografts as skin replacement after burn injury. *The Lancet*, 327(8490):1123–1124, 1986.

[35] Anthony D Metcalfe and Mark WJ Ferguson. Tissue engineering of replacement skin: the crossroads of biomaterials, wound healing, embryonic development, stem cells and regeneration. *Journal of the Royal Society Interface*, 4(14):413–437, 2007.

[36] Richard P Visconti, Vladimir Kasyanov, Carmine Gentile, Jing Zhang, Roger R Markwald, and Vladimir Mironov. Towards organ printing: engineering an intra-organ branched vascular tree. *Expert opinion on biological therapy*, 10(3):409–420, 2010.

[37] Sean V Murphy and Anthony Atala. 3d bioprinting of tissues and organs. *Nature biotechnology*, 32(8):773–785, 2014.

[38] Francoise Marga, Karoly Jakab, Chirag Khatiwala, Benjamin Shepherd, Scott Dorfman, Bradley Hubbard, Stephen Colbert, and Gabor Forgacs. Toward engineering functional organ modules by additive manufacturing. *Biofabrication*, 4(2):022001, 2012.

[39] Vladimir Mironov, Glenn Prestwich, and Gabor Forgacs. Bioprinting living structures. *Journal of Materials Chemistry*, 17(20):2054–2060, 2007.

[40] Murat Guvendiren, Joseph Molde, Rosane MD Soares, and Joachim Kohn. Designing biomaterials for 3d printing. *ACS biomaterials science & engineering*, 2(10): 1679–1693, 2016.

[41] Sean V Murphy, Aleksander Skardal, and Anthony Atala. Evaluation of hydrogels for bio-printing applications. *Journal of Biomedical Materials Research Part A*, 101 (1):272–284, 2013.

[42] Shuai Wang, Jia Min Lee, and Wai Yee Yeong. Smart hydrogels for 3d bioprinting. *International Journal of Bioprinting*, 1(1), 2015.

[43] Susmita Bose, Sahar Vahabzadeh, and Amit Bandyopadhyay. Bone tissue engineering using 3d printing. *Materials Today*, 16(12):496–504, 2013.

[44] Karoly Jakab, Cyrille Norotte, Francoise Marga, Keith Murphy, Gordana Vunjak-Novakovic, and Gabor Forgacs. Tissue engineering by self-assembly and bioprinting of living cells. *Biofabrication*, 2(2):022001, 2010.

[45] Aleksander Skardal, Mahesh Devarasetty, Hyun-Wook Kang, Ivy Mead, Colin Bishop, Thomas Shupe, Sang Jin Lee, John Jackson, James Yoo, Shay Soker, et al. A hydrogel bioink toolkit for mimicking native tissue biochemical and mechanical properties in bioprinted tissue constructs. *Acta biomaterialia*, 25:24–34, 2015.

[46] CHIRAG KHATIWALA, RICHARD LAW, BENJAMIN SHEPHERD, SCOTT DORFMAN, and MARIE CSETE. 3d cell bioprinting for regenerative medicine research and therapies. *Gene Therapy and Regulation*, 7(01):1230004, 2012.

[47] Bernhard O Palsson and Sangeeta N Bhatia. Tissue engineering 2004. *Upper Saddle River, New Jersey*, 7458.

[48] Binil Starly, Wing Lau, Tom Bradbury, and Wei Sun. Internal architecture design and freeform fabrication of tissue replacement structures. *Computer-Aided Design*, 38(2):115–124, 2006.

[49] Cheng Yu Lin, Noboru Kikuchi, and Scott J Hollister. A novel method for biomaterial scaffold internal architecture design to match bone elastic properties with desired porosity. *Journal of biomechanics*, 37(5):623–636, 2004.

[50] JM Taboas, RD Maddox, PH Krebsbach, and SJ Hollister. Indirect solid free form fabrication of local and global porous, biomimetic and composite 3d polymer-ceramic scaffolds. *Biomaterials*, 24(1):181–194, 2003.

[51] Sertug Sogutlu and Bahattin Koc. Stochastic modeling of tissue engineering scaffolds with varying porosity levels. *Computer-Aided Design and Applications*, 4(5): 661–670, 2007.

[52] Damien P Byrne, Damien Lacroix, Josep A Planell, Daniel J Kelly, and Patrick J Prendergast. Simulation of tissue differentiation in a scaffold as a function of porosity, young's modulus and dissolution rate: application of mechanobiological models in tissue engineering. *Biomaterials*, 28(36):5544–5554, 2007.

[53] Tejas Shyam Karande. *Effect of scaffold architecture on diffusion of oxygen in tissue engineering constructs*. The University of Texas at Austin, 2007.

[54] AKM Khoda, Ibrahim T Ozbolat, and Bahattin Koc. Designing heterogeneous porous tissue scaffolds for additive manufacturing processes. *Computer-Aided Design*, 45(12):1507–1523, 2013.

[55] M Gruene, M Pflaum, A Deiwick, L Koch, S Schlie, C Unger, M Wilhelmi, A Haverich, and BN Chichkov. Adipogenic differentiation of laser-printed 3d tissue grafts consisting of human adipose-derived stem cells. *Biofabrication*, 3(1):015005, 2011.

[56] Stefanie Michael, Heiko Sorg, Claas-Tido Peck, Lothar Koch, Andrea Deiwick, Boris Chichkov, Peter M Vogt, and Kerstin Reimers. Tissue engineered skin substitutes created by laser-assisted bioprinting form skin-like structures in the dorsal skin fold chamber in mice. *PloS one*, 8(3):e57741, 2013.

[57] Kyle W Binder, Weixin Zhao, Tamer Aboushwareb, Dennis Dice, Anthony Atala, and James J Yoo. In situ bioprinting of the skin for burns. *Journal of the American College of Surgeons*, 211(3):S76, 2010.

[58] Nieves Cubo, Marta Garcia, Juan F del Cañizo, Diego Velasco, and Jose L Jorcano. 3d bioprinting of functional human skin: production and in vivo analysis. *Biofabrication*, 9(1):015006, 2016.

[59] Sven Van Poucke, Roald Nelissen, Philippe Jorens, and Yves Vander Haeghen. Comparative analysis of two methods for wound bed area measurement. *International wound journal*, 7(5):366–377, 2010.

[60] Tim D Jones and Peter Plassmann. An active contour model for measuring the area of leg ulcers. *IEEE Transactions on Medical Imaging*, 19(12):1202–1210, 2000.

[61] Bryan F Jones and Peter Plassmann. An instrument to measure the dimensions of skin wounds. *IEEE Transactions on Biomedical Engineering*, 42(5):464–470, 1995.

[62] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *In the 12th International Symposium on Experimental Robotics (ISER*, 2010.

[63] H Albert. A survey of optical imaging techniques for assessing wound healing. *Int. J. Intell. Control. Syst*, 17:79–85, 2012.

[64] Dereck W Paul, Pejhman Ghassemi, Jessica C Ramella-Roman, Nicholas J Prindeze, Lauren T Moffatt, Abdulnaser Alkhalil, and Jeffrey W Shupp. Noninvasive imaging technologies for cutaneous wound assessment: a review. *Wound Repair and Regeneration*, 23(2):149–162, 2015.

[65] Fabio Remondino and Sabry El-Hakim. Image-based 3d modelling: A review. *The Photogrammetric Record*, 21(115):269–291, 2006.

[66] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. *Surface reconstruction from unorganized points*, volume 26. ACM, 1992.

[67] AB Ekoule, FC Peyrin, and CL Odet. A triangulation algorithm from arbitrary shaped multiple planar contours. *ACM Transactions on Graphics (TOG)*, 10(2): 182–199, 1991.

[68] Young-Kyu Choi and Kyu Ho Park. A heuristic triangulation algorithm for multiple planar contours using an extended double branching procedure. *The Visual Computer*, 10(7):372–387, 1994.

[69] Eric Keppel. Approximating complex surfaces by triangulation of contour lines. *IBM Journal of research and development*, 19(1):2–11, 1975.

[70] Henry Fuchs, Zvi M Kedem, and Samuel P Uselton. Optimal surface reconstruction from planar contours. *Communications of the ACM*, 20(10):693–702, 1977.

[71] Larry T Cook, P Nong Cook, Kyo Rak Lee, Solomon Batnitzky, Bert YS Wong, Steven L Fritz, Jonathan Ophir, Samuel J Dwyer, Lawrence R Bigongiari, and Arch W Templeton. An algorithm for volume estimation based on polyhedral approxi mation. *IEEE Transactions on Biomedical Engineering*, (9):493–500, 1980.

[72] Henry N Christiansen and Thomas W Sederberg. Conversion of complex contour line definitions into polygonal element mosaics. In *ACM Siggraph Computer Graphics*, volume 12, pages 187–192. ACM, 1978.

[73] Nikolaos C Gabrielides, Alexandros I Ginnis, Panagiotis D Kaklis, and Menelaos I Karavelas. G1-smooth branching surface construction from cross sections. *Computer-Aided Design*, 39(8):639–651, 2007.

[74] Pei Zhou and Wen-Han Qian. Constructive g1 connection of multiple freeform pipes in arbitrary poses. *Computer Aided Geometric Design*, 30(5):462–475, 2013.

[75] Hyungjun Park and Kwangsoo Kim. Smooth surface approximation to serial cross-sections. *Computer-Aided Design*, 28(12):995–1005, 1996.

[76] Hyungjun Park and Kwangsoo Kim. 3-d shape reconstruction from 2-d cross-sections. *Journal of design and manufacturing*, 5:171–185, 1995.

[77] Bruce R Piper. " visually smooth interpolation with triangular bezier patches" ed. g. farin. *Geometric modeling: algorithms and new trends*, pages 221–233, 1987.

[78] Leon A Shirman and Carlo H Séquin. Local surface interpolation with bézier patches. *Computer Aided Geometric Design*, 4(4):279–295, 1987.

[79] Wayne Tiller. Rational b-splines for curve and surface representation. *IEEE Computer Graphics and Applications*, 3(6):61–69, 1983.

[80] Charles D Woodward. Skinning techniques for interactive b-spline surface interpolation. *Computer-Aided Design*, 20(8):441–451, 1988.

[81] Olalekan Solomon Odesanya, Warren N Waggenspack, and David E Thompson. Construction of biological surface models from cross-sections image processing. *IEEE transactions on biomedical engineering*, 40(4):329–334, 1993.

[82] Les Piegl and Wayne Tiller. *The NURBS book*. Springer Science & Business Media, 2012.

[83] Kuntal Samanta, Ibrahim T Ozbolat, and Bahattin Koc. Optimized normal and distance matching for heterogeneous object modeling. *Computers & Industrial Engineering*, 69:1–11, 2014.

[84] Elsa Abbena, Simon Salamon, and Alfred Gray. *Modern differential geometry of curves and surfaces with Mathematica*. CRC press, 2006.

[85] Wolfgang Kühnel. *Differential geometry*, volume 77. American Mathematical Soc., 2015.

[86] Martin Held. Voronoi diagrams and offset curves of curvilinear polygons. *Computer-Aided Design*, 30(4):287–300, 1998.

[87] Martin Held and Stefan Huber. Topology-oriented incremental computation of voronoi diagrams of circular arcs and straight-line segments. *Computer-Aided Design*, 41(5):327–338, 2009.

[88] Arindam Sinharay, Tanusree Roy, Antara Saha, and Vaswar Pande. Algorithm of 2d-voronoi diagram construction with rectangular block. In *Recent Advances in Information Technology (RAIT), 2016 3rd International Conference on*, pages 582–585. IEEE, 2016.

[89] Jiechen Wang, Can Cui, Yikang Rui, Liang Cheng, Yingxia Pu, Wenzhou Wu, and Zhenyu Yuan. A parallel algorithm for constructing voronoi diagrams based on point-set adaptive grouping. *Concurrency and Computation: Practice and Experience*, 26(2):434–446, 2014.

[90] Deok-Soo Kim, Youngsong Cho, Joonghyun Ryu, and Chong-Min Kim. Topologies of surfaces on molecules and their computation in o (n) time. *Computer-Aided Design*, 42(9):795–807, 2010.

[91] Masao Iri, Kazuo Murota, and Takao Ohya. A fast voronoi-diagram algorithm with applications to geographical optimization problems. In *System modelling and optimization*, pages 273–288. Springer, 1984.

[92] Alok Aggarwal, Leonidas J Guibas, James Saxe, and Peter W Shor. A linear-time algorithm for computing the voronoi diagram of a convex polygon. *Discrete & Computational Geometry*, 4(6):591–604, 1989.

[93] Navid Khani, Ali Nadernezhad, Paulo Bartolo, and Bahattin Koc. Hierarchical and spatial modeling and bio-additive manufacturing of multi-material constructs. *CIRP Annals-Manufacturing Technology*, 2017.

[94] Ali Nadernezhad, Navid Khani, Gözde Akdeniz Skvortsov, Burak Toprakhisar, Ezgi Bakirci, Yusuf Menceloglu, Serkan Unal, and Bahattin Koc. Multifunctional 3d printing of heterogeneous hydrogel structures. *Scientific reports*, 6:33178, 2016.