UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

LAURA AMAYA TORRES

Automatic Generation of Patient-Specific 3D Models of Organs Using an Unified Deep Learning Approach

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Prof. Dr. Anderson Maciel
Coadvisor: Prof. Dr. Mariana Recamonde Mendoza

Porto Alegre
February 2020

CIP — CATALOGING-IN-PUBLICATION

Amaya Torres, Laura

Automatic Generation of Patient-Specific 3D Models of Organs Using an Unified Deep Learning Approach / Laura Amaya Torres. – Porto Alegre: PPGC da UFRGS, 2020.

71 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2020. Advisor: Anderson Maciel; Coadvisor: Mariana Recamonde Mendoza.

1. Mesh generation. 2. Organ reconstruction. 3. Free-form deformation. 4. Deep learning. I. Maciel, Anderson. II. Recamonde Mendoza, Mariana. III. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitor: Prof. Rui Vicente Oppermann
Vice-Reitora: Prof[a]. Jane Fraga Tutikian
Pró-Reitor de Pós-Graduação: Prof. Celso Giannetti Loureiro Chaves
Diretora do Instituto de Informática: Prof[a]. Carla Maria Dal Sasso Freitas
Coordenadora do PPGC: Prof[a]. Luciana Salete Buriol
Bibliotecária-chefe do Instituto de Informática: Beatriz Regina Bastos Haro

To my dear family and friends.

“My fault, my failure, is not in the passions I have,
but in my lack of control of them.”
— JACK KEROUAC

ABSTRACT

Reconstruction of 3D shapes from images using convolutional neural networks (CNN) has become a very studied field in recent years and has demonstrated great performance. Rigid and non-rigid objects have been reconstructed using several types of 3D representations and approaches with single or multiple images. However, the reconstructed objects are part of the outside world (they are visible and can be photographed). The most used imaging techniques to obtain visual information from organs are computerized tomographies (CT) and magnetic resonance imaging (MRI), which do not generate regular 2D images. Our main objective is to evaluate the feasibility of using deep learning approaches to directly reconstruct 3D models of organs from medical images using free-form deformations (FFD). To do this, we combined existing algorithms used for segmentation of medical images in 3D space with 3D object reconstruction techniques in a fully automatic convolutional network model. We tested our proposed method by training models for two different organs with higher and lower shape complexity (liver and prostate, respectively). The reconstructed models generated by our network are coherent with the overall shape of the organs demonstrating that it can be learned. However, more work needs to be done to obtain organ models that truly represent their actual structure.

Keywords: Mesh generation. organ reconstruction. free-form deformation. deep learning.

Geração automática de modelos 3D de órgãos para pacientes específicos usando abordagem unificada de aprendizado profundo

RESUMO

A reconstrução de formas 3D a partir de imagens usando redes neurais convolucionais tornou-se um campo muito estudado nos últimos anos e tem demonstrado ótimo desempenho. Objetos rígidos e não rígidos foram reconstruídos usando vários tipos de representações em 3D, e abordagens com uma ou múltiplas imagens. Contudo, os objetos reconstruídos são parte do mundo exterior (são visíveis e podem ser fotografados). As técnicas de imagiologia mais utilizadas para obter informações visuais a partir de órgãos são as tomografias computadorizadas e a ressonância magnética, as quais não geram imagens 2D regulares. Nosso principal objetivo é avaliar a viabilidade de usar abordagens de aprendizado profundo para reconstruir diretamente modelos 3D de órgãos a partir de imagens médicas usando deformações de forma livre. Para isso, nós combinamos algoritmos existentes usados para segmentação de imagens médicas no espaço 3D com técnicas de reconstrução de objetos 3D em um modelo de rede neural completamente automático. Nós testamos nosso método proposto treinando modelos para dois órgãos diferentes, com maior e menor complexidade de forma (fígado e próstata, respectivamente). Os modelos reconstruídos gerados por nossa rede são coerentes com a forma geral dos órgãos, demonstrando que a mesma pode ser aprendida. Entretanto, mais trabalho precisa ser realizado de forma a obter modelos de órgãos que representem autenticamente a sua estrutura real.

Palavras-chave: geração de malhas, reconstrução de órgaõs, FFD, aprendizado profundo

.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CT	Computed Tomography
MRI	Magnetic Resonance Imaging
AI	Artificial Intelligence
MLP	Multilayer Perceptron
CNN	Convolutional Neural Network
FCN	Fully Convolutional Network
RNN	Recurrent Neural Network
LSTM	Long-Short Term Memory
CD	Chamfer Distance

CONTENTS

1 INTRODUCTION

Medical image analysis aims to extract clinically useful information from medical images. Over the last decade, research in this area has made significant progress focusing on challenging tasks such as image registration and image segmentation (RUECKERT; GLOCKER; KAINZ, 2016). The latter can be used for purposes of computer-aided diagnosis, computer-assisted surgery, surgery simulators, and general surgery planning, allowing clinicians to design preoperative schemes that ensure the patients safety.

Computer-assisted surgical systems can improve the surgeons understanding of the organ being treated. Also, the use of 3D geometric models has shown improvements in tumor localization, precision of surgery planning, and even improved orientation and confidence of the clinician during the surgery (PALOMAR et al., 2016). To accomplish this, it is necessary to have patient-specific models accurately representing the organ geometry. These models can be reconstructed from image data, such as those acquired through computed tomography (CT) or magnetic resonance imaging (MRI). Manual, semi-automatic or automatic reconstruction techniques are used depending on body part and acquisition modality (BUCKI; LOBOS; PAYAN, 2010).

The reconstruction process can be tackled using different approaches. One of them generates the 3D models in two steps: organ segmentation followed by 3D model reconstruction. Data obtained from CT and MRI consist of a series of 2D images with information in their pixels that can be stacked to create a volume. The volumetric information can be used to separate the target organ from the other structures present in the exam, in a process known as segmentation. This is a straightforward task when organs have well-differentiated intensities. However, neighboring soft tissues such as skeletal muscles, heart, liver, and stomach share similar intensities. The low contrast makes the tissues boundaries merge and be difficult to perceive both with the eye and image processing algorithms (LI et al., 2015).

A second approach can be understood as a single step method where usually generic meshes are deformed to the specific target organ shape. This is more often useful when intensities are well differentiated as in bone structures (BUCKI; LOBOS; PAYAN, 2010), but can also be applied to soft organs (Feng Ding et al., 2009).

The image segmentation part of the problem is based on different well-developed methods as Isosurfaces and Level Sets, Markov Random Fields, Region Growing, Atlas-Guided Approaches, among others (CARDENAS et al., 2019; LENCHIK et al., 2019).

In recent years, however, convolutional neural networks (CNN) approaches have become the methodology of choice as they yield outstanding results in imaging analysis tasks (LITJENS et al., 2017). Once the organ segmentation step is completed, the generation of the 3D models uses a reconstruction algorithm. The most widespread being Marching Cubes (LORENSEN; CLINE, 1987) and its variations.

Research in computer graphics and vision have also explored the reconstruction of 3D models from sources different than medical scans. Specifically, it has studied the process of recovering 3D shapes of objects with information obtained from one or multiple regular 2D projection images (photographs or renderings). This is useful in applications such as virtual reality, augmented reality, robotics, and CAD, where the system needs to recognize objects in the real or virtual environment. Several methods have been proposed to accomplish this, including the use of stereo-based techniques that use several views of the same object to triangulate the 3D coordinates of the pixels along the images (HARTLEY; ZISSERMAN, 2003). These methods require the cameras to be accurately calibrated in order to obtain good results and doing so is not always feasible.

Recent approaches explored the use of machine learning algorithms, particularly CNNs, to infer the 3D geometry of the objects using images as the input data. Different kinds of network architectures had been used, as recurrent neural networks (RNN) (CHOY et al., 2016), and variational auto-encoders combined with 3D generative adversarial networks (3D-VAE-GAN) in a supervised manner (WU et al., 2016), as well as different types of representations to model the shapes that the algorithm needs to learn, as point clouds (FAN; SU; GUIBAS, 2017) or voxelized data (XIE et al., 2019). In a basic reconstruction architecture, information is obtained from the images using an encoder to map the inputs into a high-dimensional feature vector, and then a decoder is used to decode the obtained features to the desired output type (HAN; LAGA; BENNAMOUN, 2019). These techniques have been successfully used to generate rigid or man-made shapes, such as planes, chairs, and cars, and non-rigid or organic shapes like hands and faces.

Learning-based solutions for 3D object reconstruction and medical image segmentation are similar because they use features acquired from the images to generate their desired result. Motivated by the capacity of deep neural networks to obtain such understanding that allows them to accomplish difficult segmentation tasks, added to their showed capabilities to generate 3D representation of shapes, we explored their use in generating 3D models of organs in a single step, going from the medical tomographic dataset

(CT or MRI) directly to the surface representation.

1.1 Objective

In this context, the general objective of this thesis is to automatically generate patient-specific 3D models of organs from medical images using deep neural networks in a single step approach. This goal was then delimited in the scope of the following specific objectives.

1.1.1 Specific Objectives

- Gain understanding of the theoretical basis and state-of-the-art techniques to do organ segmentation and mesh generation based on deep neural networks;
- Develop an algorithm using a deep neural network architecture able to learn meaningful representations of organs, accomplishing the organ segmentation and 3D model generation in a single step;
- Generate 3D models that accurately represent the geometry of the organ of interest maintaining characteristic features and satisfying mesh quality requirements and metrics;
- Investigate the applicability of the 3D models in surgical simulators and surgery planning, making them easy to introduce in commercial software pipelines and using standard output file formats.

1.2 Thesis Organization

This thesis is organized as follows: Chapter 2 presents the theoretical background on Machine Learning, representation of anatomy and deformation of meshes; Chapter 3 describes an overview of the most common 3D organ and objects generation techniques based on state-of-the-art literature; Chapter 4 provides details about the proposed method and each of its parts; Chapter 5 presents the data sets used in this work, the performed experiments and the overall results of this research; and Chapter 6 points out our main contributions, faced difficulties and limitations, and possibilities for future work.

2 BACKGROUND

In this chapter we present a compilation of necessary information to better understand the content of this work.

2.1 Machine Learning

The field of artificial intelligence (AI) had its origins solving tasks that were difficult for humans, but easy to solve for computers, as proving theorems (GELERNTER, 1959 apud RUSSELL; NORVIG, 2009) or playing chess (CAMPBELL; HOANE; HSU, 2002). However, when facing problems that were simple for humans, as speech recognition, understanding of images and text, or making medical diagnoses, AI encountered a greater challenge.

A solution developed to solve these intuitive problems was machine learning, which uses algorithms with the ability to learn from examples. These algorithms build knowledge based in simple concepts and their hierarchical relations, and sometimes, the structure connecting the concepts is deep, creating what we know as deep learning (GOODFELLOW; BENGIO; COURVILLE, 2016). It is generally used as a tool to solve complex AI tasks and problems containing data with high dimensionality.

2.1.1 Learning Algorithms

Machine learning algorithms are algorithms that have the capability to learn from data (GOODFELLOW; BENGIO; COURVILLE, 2016). Here, learning is considered as the process by which the algorithm obtains the ability to perform a certain task. To carry out the learning process it is necessary to input the machine learning system data from which it can obtain information. This set of data, usually called the training set, is composed of examples defined by features that are used to tune the algorithm's parameters, allowing it to process new examples in the way it is expected.

There are many problems that can be solved using machine learning, some common tasks defined by Goodfellow, Bengio and Courville (2016) include:

- Classification: In this task the goal is to assign the inputs to one of a finite set of classes. Examples of classification are object recognition from images or handwrit-

ten digit recognition.

- Regression: In this type of task, the output of the training examples is not a class but rather a continuous value. Examples of regression are temperature prediction, or prediction of future prices of securities.
- Transcription: Here, the system receives unstructured representations of data and its goal is to transcribe it into a textual form. The input could be an audio sample or an image containing text and the expected output would be a sequence of characters that represents the words on it.
- Structured prediction: These are tasks where the output consists of a vector with important dependencies between its elements. Capturing these dependencies is as important as capturing the input-output relationship. Pixel-wise segmentation of images is an example of it as well as image captioning and other translation and transcription tasks where the output needs to be a valid sentence.
- Density estimation: In this type of tasks, the computer program needs to find a function that represents the distribution of data within the input space. It is called the probability density function.

Depending of the type of feedback given to the algorithm, Russell and Norvig (2009) define three main types of learning:

- Supervised learning, where the algorithm learns directly from input-output pairs and tries to learn a function that will yield the desired output for a given input. Classification and regression are common supervised tasks.
- Unsupervised learning, where the algorithm learns form the data even though no feedback is given. It can be used to discern multiple categories in a collection of objects, known as clustering, or to project data from a high-dimensional space to two or three dimensions, known as principal component analysis.
- Reinforcement learning, where the algorithm receives rewards or punishments when doing something "right" or "wrong" and uses that knowledge to determine which of the actions prior to the reinforcement were most responsible for it in order to maximize the total reward.

The supervised learning process usually consist of a training phase where the algorithm is shown the training set, and an evaluation phase, where a test set composed of data the algorithm hasn't seen before is used to assess its performance. To do it, is necessary to define quantitative performance metrics, known as cost, loss or error, that will

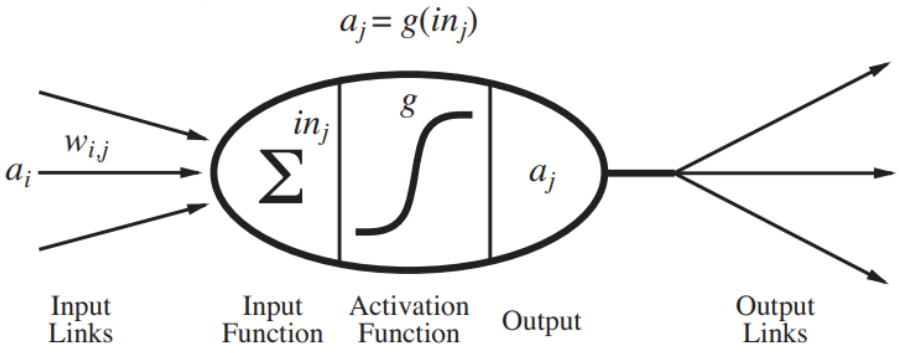
depend of the specific task the system is trying to learn. When the cost is calculated in the training set it is called the training error and by minimizing it during the training process, the learning problem is being optimized. Likewise, when the error is calculated in the test dataset, it is known as the test error or generalization error.

In general terms, an algorithm will perform well if it has a small training error and the gap between the training and test errors is small. If the machine learning system is not being able to achieve a low training error, it is said that it is under-fitting. If the gap between the training and test errors is too big, it is said that the algorithm is over-fitting. The ultimate goal of the learning process must be to perform well on the test set as this will increase the possibility of performing well in new, unseen data.

2.1.2 Artificial Neural Networks

Research about the brain in the neuroscience area inspired some of the first AI works that aimed to create artificial neural networks. The findings that motivated this creation were about mathematical representations of how information is processed in biological systems (BISHOP, 2006), and hypothesis that the base of mental activity resides in electrochemical connections in networks of brain cells called neurons (RUSSELL; NORVIG, 2009). The proposed network model was composed of small units that resembled these cells. The mathematical model of these units (MCCULLOCH; PITTS, 1943 apud RUSSELL; NORVIG, 2009) can be seen in Figure 2.1.

Figure 2.1: Mathematical model of a neuron. The output activation of the network is calculated with the output activation of the previous unit.



Adapted from: Russell and Norvig (2009) (RUSSELL; NORVIG, 2009)

Inside the network, the units are connected so that activation a_i is propagated from unit i to unit j. Each connection has a weight $w_{i,j}$ that determines its strength. First, each

unit j computes the weighted sum of its inputs

$$in_j = \sum_{i=0}^{n} w_{i,j} a_i \tag{2.1}$$

and then transforms it applying a differentiable, non linear activation function g to calculate its output activation

$$a_j = g\left(in_j\right) = g\left(\sum_{i=0}^{n} w_{i,j} a_i\right) \tag{2.2}$$

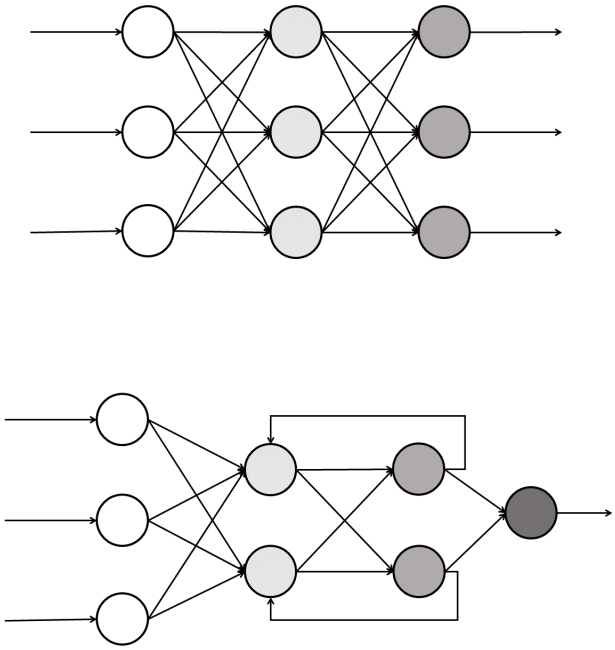
The structure of the network is then formed by stacking units in layers and creating the links between them. The layers that are not directly connected to the output units are called hidden layers. As the activation functions in these layers are non linear, the network becomes a linear combination of non linear functions that maps the input variables to the output variables and is controlled by adjustable parameters or weights that are adapted during the training.

The activation function of the last layer depends of the specific task the algorithm is learning and the nature of the data. Examples of functions that can be used are the identity function for regression problems, logistic sigmoid functions for binary classification problems, or softmax activation functions for multiclass problems, even thought different activation functions can also be used for these tasks.

The way the units are structured determine the type of the network, the two basic definitions being Feedforward Networks and Recurrent Networks (see Figure 2.2). In the first type, the connections go in one direction without loops so the network represents a function of its current input. The second type, recurrent networks, feeds its outputs back to its inputs so the response to a given input depends of its initial state that may depend of past inputs. The feedforward networks, also called Multilayer Perceptron (MLP) are the most relevant model for deep learning.

Once the structure of the network is set, the optimization process starts by initializing the network's parameters and then moving through the weight space in an iterative manner to minimize the error function. By propagating the errors backwards through the network, the gradients of the error with respect to the weights can be calculated in a process known as Backpropagation (RUMELHART; HINTON; WILLIAMS, 1988). These derivatives are then used to compute the adjustments made to the weights, and the process is repeated until any stopping criteria is met.

Figure 2.2: Basic structure of feedforward (above) and recurrent (below) networks. In feedforward networks the information flows in one direction without loops while recurrent networks feed its outputs to its inputs.



Source: Original image

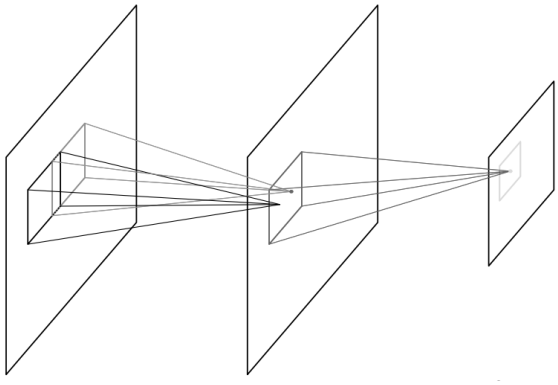
2.1.3 Convolutional Neural Networks

Convolutional networks (LECUN et al., 1989) are a special type of neural network that uses convolution operations in at least one of its layers. They are useful when working with data that has a known grid-like structure as time-series data and images.

The basic structure of a Convolutional Neural Network (CNN) can be seen in Figure 2.3. The convolutional layer is composed of units arranged into planes known as feature maps. All of the units inside a feature map share the same weight, so the whole feature map will detect a specific pattern in different parts of the input image. Using multiple feature maps in each layer allows the model to recognize different patterns in the image.

With the received input, first, the layer performs convolutions in parallel to produce the linear activations than are then transformed by a non linear activation function as with regular neural networks. Then, the output of the convolutional layer is passed to the subsampling layer where the units receive inputs from small regions of the feature maps performing the subsampling action (BISHOP, 2006). The functions used to sub-

Figure 2.3: Basic structure of a convolutional neural network. The input image (left) is followed but a convolutional layer (center) and then a sub-sample layer (right). Networks are composed of several pairs of convolutional + sub-smapling layers.



Adapted from: Bishop (2006) (BISHOP, 2006)

sample the feature map can be max pooling, where the output is the biggest input within the rectangular region, the average, the L^2 norm of weighted averages, among others (GOODFELLOW; BENGIO; COURVILLE, 2016).

The resulting values are then multiplied by adaptive weights and finally transformed using non linear activation functions like in regular layers. This basic structure can be repeated as necessary creating the architecture of the network. Then, the final layer of the network has to be chosen based on the task the algorithm needs to learn.

As with a regular neural network, the training process can be accomplished by minimizing the error metric using backpropagation. The main difference is that when updating the weights, the constraint of the same weight for each feature map must be maintained (BISHOP, 2006).

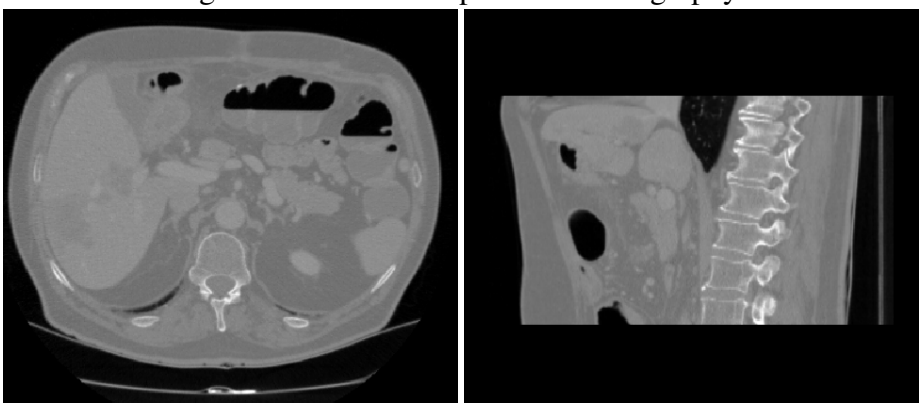
2.2 Representation of Anatomy in Computer Graphics

Medical imaging is a technique that allows the creation of images from anatomy, specifically the structures hidden by the skin and bones, helping clinicians to reliably detect and diagnose diseases. Some of the most common medical imaging modalities include X-Ray, Computed Tomography, Magnetic Resonance Imaging, Ultrasound Imaging, techniques based on nuclear emission (PET, SPECT), among others (BEUTEL; KUNDEL; METTER, 2000).

Images generated by the aforementioned techniques are different depending on which technique was used. CT and MRI techniques yield a series of 2D image slices with

information stored in their pixels (see Figure 2.4). These individual slices can be stacked, providing volumetric information in units of data known as a voxels. Voxel intensities in medical imaging vary according to the physical response of the tissue to the scanning modality.

Figure 2.4: Axial and sagittal slices of a computerized tomography of the abdominal area.



Source: Original Image

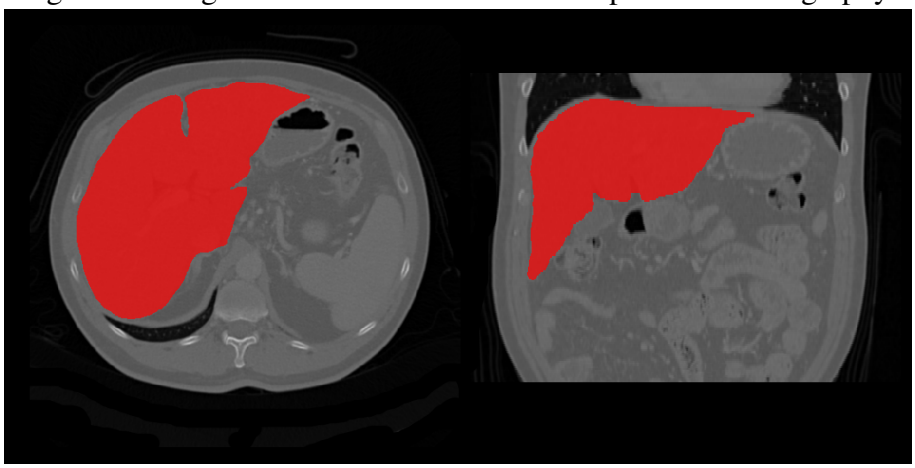
The resulting volume provides information about the whole area that originated the scan (the part of the body that the scan is representing) but in the case of computer-aided diagnosis, computer-assisted surgery, surgery simulators and general surgery planning, the clinicians usually need information about specific organs making it necessary to change this representation. The process of separating a specific organ from the rest of the scan is known as segmentation.

The usual result from a segmentation process is a scan whose voxels contain only binary values indicating if said voxel belongs to the organ or not. Example of a segmented CT scan can be seen in Figure 2.5. To generate a 3D model from that segmented representation, the usual approach is to create a polygonal mesh.

Polygonal meshes are a collection of vertices, edges formed by vertex pairs, and polygons formed by sequences of edges and vertices (SMITH, 2006). A polygonal mesh of the same organ extracted using the marching cubes algorithm can be seen in Figure 2.6 (left).

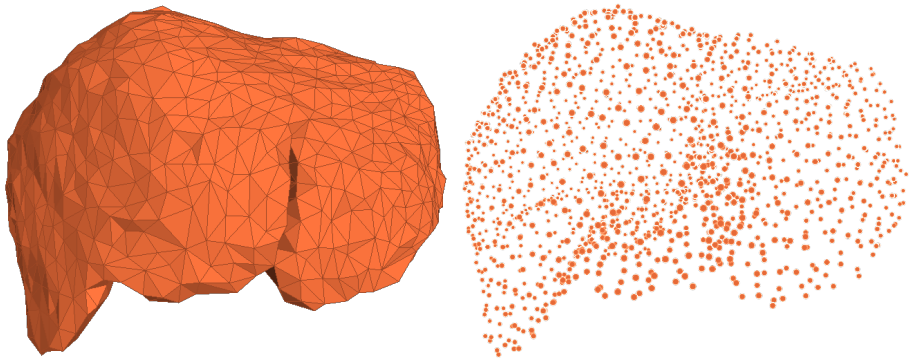
When there is no information about how the vertices of the mesh are connected, the set of points simply becomes a point cloud. The point cloud version of the same organ can be seen in Figure 2.6 (right). In this work, we will represent 3D models with point clouds that can be derived from polygonal meshes.

Figure 2.5: Segmentation of the liver in a computerized tomography.



Source: Original image

Figure 2.6: Polygonal mesh (left) and point cloud (right) representations of reconstructed model of a liver.



Source: Original image

2.3 Free-form Deformation

3D meshes can be deformed using Free-form Deformation (FFD) (SEDERBERG; PARRY, 1986). FFD is a technique that uses a deformation grid placed around a 3D shape and moves it in such a way that deformations made to the grid also deform the object in it.

This technique places a local coordinate system defined by the vectors S, T, U and origin p_0, around the shape in a parallelepiped region. Thus, for any point p in the original shape, its coordinates (s, t, u) can be calculated in the new coordinate system as:

$$s = \frac{T \times U \cdot (p - p_0)}{T \times U \cdot S}, \; t = \frac{S \times U \cdot (p - p_0)}{S \times U \cdot T}, \; u = \frac{S \times T \cdot (p - p_0)}{S \times T \cdot U}. \tag{2.3}$$

Then, the parallelepiped space is divided in $l+1$, $m+1$ and $n+1$ planes in the S, T and U directions respectively, where the intersections of the planes are the control points of the deformation grid and their locations are defined as:

$$P_{ijk} = p_0 + \frac{i}{l}S + \frac{j}{m}T + \frac{k}{n}U. \tag{2.4}$$

By moving the control points, the deformed position p' of a point p can be calculated as:

$$p' = \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} B_{i,l}(s)B_{j,m}(t)B_{k,n}(u)\left(P_{ijk} + \Delta P_{ijk}\right) \tag{2.5}$$

where $B_{n,m}(x) = \binom{n}{m}(1-x)^{n-m}x^m$ is the Bernstein polynomial of degree m, and ΔP_{ijk} is the 3D deformation offset at the grid control point P_{ijk}.

Equation 2.5 can be written in a matrix form as:

$$P' = B(P + \Delta P) \tag{2.6}$$

were $P' \in \mathbb{R}^{N\times 3}$ are the deformed Cartesian coordinates of the 3D shape, $B \in \mathbb{R}^{N\times M}$ is the deformation matrix, $P \in \mathbb{R}^{M\times 3}$ are the coordinates of the control points, N is the number of points in the original shape, M is the number of control points calculated as $(l+1)(m+1)(n+1)$ and ΔP are the control point offsets.

2.4 Comparison of 3D Meshes

To compare two meshes we need to measure how different they are. Relying only on the vertices of the mesh, it can be treated as a point cloud, and metrics designed for point clouds can be used. Next we explain two metrics to measure the difference between two point clouds.

2.4.1 Chamfer Distance

As defined in Fan, Su and Guibas (2017), the Chamfer distance (CD) between two point clouds measures the sum of the distances between each point and its nearest

neighbor in the other cloud. Formally, CD between two point clouds S_1 and $S_2 \subseteq \mathbb{R}^3$ is:

$$d_C(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2 \tag{2.7}$$

where $\|a\|_2$ is the L^2 norm of a, also known as the Euclidean distance.

2.4.2 Mean Squared Distance

In the case where there is one to one correspondence between the two point clouds, the mean of the squared Euclidean distance for every pair of points is defined as follows:

$$d_{MSD}(S_1, S_2) = \frac{1}{n_p} \sum_{x \in S_1} \sum_{y \in S_2} \|x - y\|_2^2 \tag{2.8}$$

where n_p is the number of points in the cloud.

3 RELATED WORK

The generation of 3D models of organs from medical data is a problem that involves inferring a mesh from a volumetric image obtained by a medical imaging technique, as a CT or an MRI. The construction of these models has been a very active research topic along the years and still today. Solutions for multiple, as well as specific organs, have been proposed using all kind of techniques, e.g. deformable models, probability based methods, atlas guided methods, machine learning approaches, among others.

The techniques can be roughly classified in those having two-steps or having a single step. When we refer to two-steps techniques, we are referring to those solutions that do an initial segmentation process of the target organ in the medical scan and then generate a polygonal mesh from the obtained segmentation. Single step methods, on the other hand, can directly generate the shape of the organ using deformable models that adapt the shape of a curve (2D) or mesh (3D) to overlap with the organ of interest. By generating the 3D model of the organ, its segmentation is also implicitly obtained

In recent years the popularity of organ segmentation algorithms based on deep learning has risen and these methods have become the state-of-the-art (CARDENAS et al., 2019). This behavior has also been observed in the generation of 3D models of objects from images, were CNNs have demonstrated impressive performance (HAN; LAGA; BENNAMOUN, 2019). This is a different problem, as the initial data for which one wants to generate the 3D representation only has two dimensions, making it necessary that the machine learning algorithm learns to infer volumetric information from non-volumetric data. Because of that, a lot or research has been conducted including a wide range of reconstruction algorithms and types of 3D representations.

In this chapter, we present an overview of the most common techniques used in each stage of the 3D organ generation process. Then, we summarize the most representative methods used to reconstruct 3D model of objects from images using deep learning approaches, based on the recent review works of Han, Laga and Bennamoun (2019) and Yuniarti and Suciati (2019).

3.1 Organ Reconstruction from Medical Images

In this section we present algorithms used to segment organs from medical data as well as methods used to reconstruct the 3D models from the generated segmentation.

3.1.1 Organ Segmentation Techniques

Initial methodologies to segment organs from medical scans were limited by the low amount of segmented data available. This caused the proposed algorithms to use no prior knowledge about the scans and do the segmentation based only in pixel intensities using simple techniques as intensity thresholding or region growing (CARDENAS et al., 2019).

In time more advanced techniques were proposed, as active contours and level sets. These are considered deformable models as they deform closed surfaces to overlap the shape of the target organs. Deformable models have two main characterizations: parametric and geometric. The former characterization explicitly represents the curves in their parametric form during the deformation. Active contours are examples of it. The latter is based on the theory of curve evolution and implicitly represents curves and surfaces as a level set of a high dimensional scalar function (MESEJO et al., 2016).

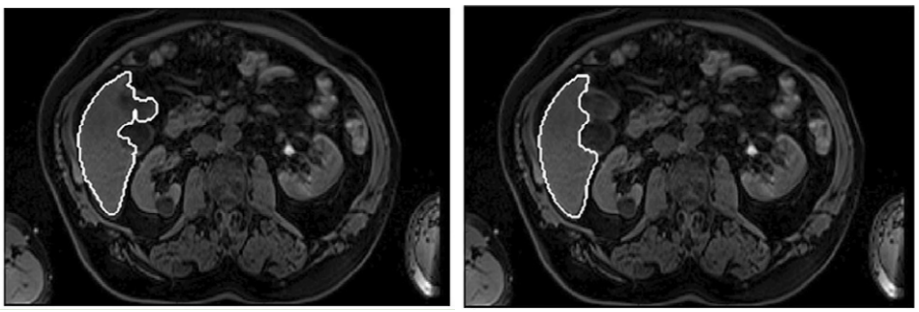
Active contours, as Snakes (KASS; WITKIN; TERZOPOULOS, 1988), are spline curves that deform under the influence of forces that pull them towards features of images such as lines or edges. This is accomplished by minimizing an energy function until the deformable model coincides with the edges of the object being segmented. As these methods often rely only on local information at the control points, the quality of the results severely depends on the contrast of the image intensities. Local minima also undermine the chances of a fully automatic performance. In such way, user intervention is required and semi-automatic approaches obtain better results.

In 2016, BERECIARTUA et al. used 3D active surfaces to segment the liver in multi-sequence MRI images. They did it by using an active contour model designed to work with uniform unconnected regions (liver and not-liver). This contour converged towards the liver boundary with the influence of an energy function and the maximization of the difference between the statistics of the liver and non-liver regions. Even though this method used a 3D active surface, it was used in a local small region of $3 \times 3 \times 3$ voxels to improve the organ segmentation (see Figure 3.1), rather than to build a 3D model of it.

Probabilistic techniques can also be used to predict the possibility of a voxel belonging to a class. Gaussian Mixture models, k-nearest neighbors, and shallow artificial neural networks (ANN) are example methods. However, they ignore region information as they focus on the problem of single voxels.

Atlas guided techniques use reference images where the areas of interest are al-

Figure 3.1: Liver segmentation of an MRI scan using 2D active contours (left) and 3D active contours (right). By using local information about the neighboring voxels a better segmentation can be achieved.



Source: Bereciartua et al. (2016) (BERECIARTUA et al., 2016)

ready segmented, called atlases, as prior knowledge to segment new images (CARDENAS et al., 2019). The use of multiple atlases helps in dealing with inter-subject variability. However, these techniques are prone to misrecognize the surrounding tissues (MOGHBEL et al., 2018). Saxena et al. (2016) proposed an automatic atlas based approach to segment multiple organs in the abdominal region (spine, kidneys, liver, aorta, and spleen) in CT scans. They use the spine as a landmark to extract regions of interest for each organ that are then segmented using an algorithm specifically designed for the target organ. These regions of interest are found at specific distances and angles from the spine. The values were obtained from the observation of 152 scans making the creation of the atlas a time consuming process as the features are handcrafted.

With more segmentation data available, traditional machine learning approaches as Support Vector Machines and Random Forests have also been used. However, deep learning approaches, particularly CNNs, have shown outstanding results in organ segmentation tasks. Methods using CNNs perform well because they use local information instead of single pixel information to produce the segmentation. However this is computationally expensive as the overlapping regions used as input for each pixel cause the same convolutions to be computed several times. To solve this, Long, Shelhamer and Darrell (2015) developed fully convolutional networks (FCN) where the final fully connected layers of regular CNNs were replaced by an up-sampling path to decode the image features to produce accurate segmentation.

The most popular FCN architecture is U-Net (CARDENAS et al., 2019). Developed by Ronneberger, Fischer and Brox (2015), U-Net combined the use of encoder-decoder paths, with skip-connections that concatenate the layers in the the decoder path with their correspondent feature map on the encoder path. The architecture showed impressive results in the segmentation task, with low computational time and also demon-

strated that the training process could be accomplished using a small set or training examples. A 3D version of this model, 3D U-Net, was proposed by Çiçek et al. (2016), making it possible to train with 3D images. It was tested in segmentation of highly variable 3D anatomical structures showing good results.

Another variation of U-Net networks are V-Nets implemented by Milletari, Navab and Ahmadi (2016). Here, the author also developed a 3D version of U-Net, but added a Dice coefficient loss function and the use of residual connections to improve the convergence time. They tested their architecture in prostate segmentation from MRI scans. Recently, an improved version of V-Net was proposed by Shen et al. (2019) allowing the model to deal with multi-region segmentation and to process larger image volumes. The authors included batch-normalization to accelerate the learning process and act as a regularization method. The algorithm was tested for multi-organ segmentation in CT scans and an example of the generated results can be seen in Figure 3.2.

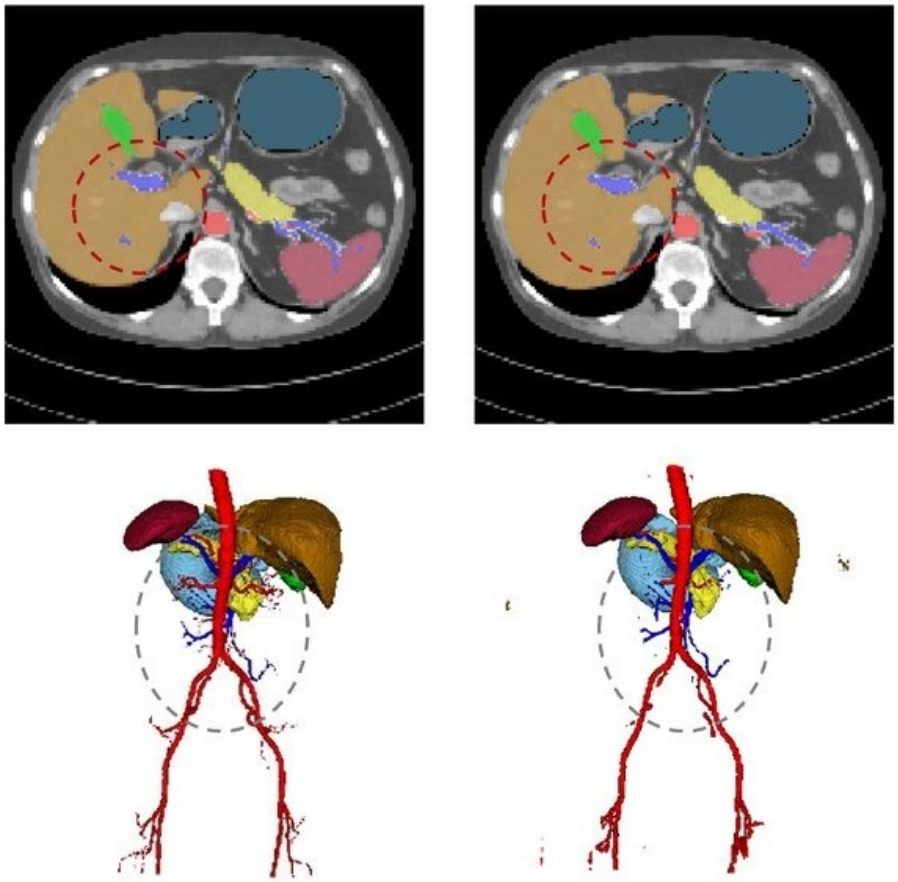
3.1.2 Reconstruction of Organs from Segmentation

When the segmentation is obtained, reconstruction algorithms can be used to create a mesh from it. The traditional algorithm to obtain isosorfaces from voxelized data is Marching Cubes (LORENSEN; CLINE, 1987). Meshes generated with it are usually refined incorporating Laplacian smoothing and decimation.

Deformable model methods have also been applied to the problem of reconstruction from medical data. In 2010, LEDERMAN et al. presented a method to generate tetrahedral meshes from 3D images with multiple region labels from already segmented medical images. They used active surfaces to minimize an energy function with three terms: "a smoothing term to remove the voxelization, a fidelity term to keep the mesh from moving too far away from the image data, and an elasticity term to keep the tetrahedra from becoming flattened or inverted as the mesh deforms." (LEDERMAN et al., 2010). Results applied to an MRI image of the brain can be seen in Figure 3.3.

Lamata et al. (2011) proposed an automatic method to generate patient-specific cubic Hermite meshes and applied it to create bi-ventricular heart models from MRI scans. First, they created binary representations of a template mesh and the patient's anatomy using binarisation and segmentation respectively. Then, they calculated a displacement field between both binary representations and applied it to the template mesh to create the personalized mesh.

Figure 3.2: Ground truth (left) and prediction (right) in multi-organ segmentation in a CT scan using improved deeper V-Nets with batch normalization.



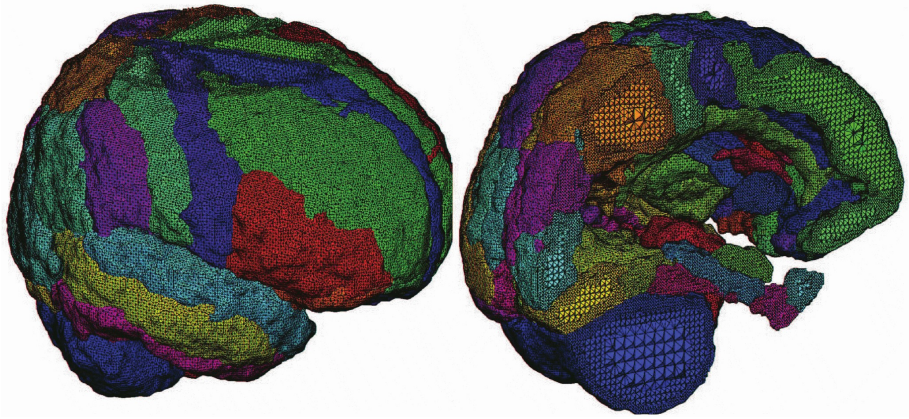
Source: Shen et al. (2019) (SHEN et al., 2019)

Similar to this approach, our method also deforms a template mesh but it does it without having the initial segmentation of the organ. Instead, it uses a V-Net architecture to gain information directly from the medical scan and encodes the obtained features into deformation parameters that modify the mesh using FFD.

3.1.3 Single Step Generation of Organ Models

Deformable models have also been used to directly generate organ meshes. In Lu et al. (2011), the authors used this technique to generate liver meshes in CT scans in an interactive approach. To do this, they created an initial liver mesh from a known segmentation and deformed it towards the best fit of the data using multiple iterations and

Figure 3.3: Single mesh of a segmented brain with 56 region labels created using active surfaces.



Source: Lederman et al. (2010) (LEDERMAN et al., 2010)

a set of user defined parameters for the deformation filter. To refine the obtained mesh, they used attractor points also defined by the user. Although our method also deforms a mesh to create the organ model, intervention of the user is not necessary during the process.

3.2 3D Object Reconstruction from Images using Deep Learning

We saw that deep learning is employed to generate segmentation data to reconstruct the organs. However, studies have also focused on using machine learning to directly generate 3D data. A good example is the reconstruction of 3D objects based on images. Algorithms in this area of research usually can be grouped by the type of representation they give to the 3D data, as it changes the way the problem needs to be approached. The used strategies also depend of the input data type and the number of images used. Single image, multiple images, and RGB-D images are different variations.

Some algorithms generate voxelized data as 3D representations. They are highly used as one can easily perform 3D convolutions on voxel grids (CHOY et al., 2016; WU et al., 2017). Choy et al. (2016) proposed a recurrent neural network called 3D-R2N2 (3D Recurrent Reconstruction Neural Network) to reconstruct objects from both single and multi-view images. In their work, the authors used the ability of Long-Short Term Memory (LSTM) networks to remember their inputs over a period of time. This way, the network learned a suitable representation of the image and adapted it recursively over time when new information about other viewpoints was presented to it. The model handled object self-occlusions by incorporating new information about parts that were hidden in

previous states and maintaining the information it had about the other parts. Examples of the reconstruction can be seen in Figure 3.4.

Figure 3.4: Voxelized reconstruction of objects using 3R-R2N2 network.

Source: Choy et al. (2016) (CHOY et al., 2016)

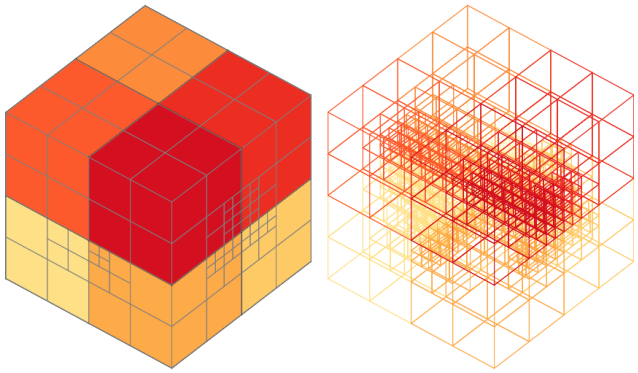
One problem with RNN approaches is that when given the same input images with different orders, the produced results are not consistent. Xie et al. (2019) proposed a method that generated 3D reconstructions by fusing several coarse volumes predicted by a decoder. The predicted coarse volumes were fed to a fusion module that selected high-quality reconstruction for each part creating a fused model that then was refined with skipped connections to create the final result. The coarse volumes were predicted in parallel, making the model faster than LSTM methods and eliminating the effect of the order of input images present in RNN.

Wu et al. (2017) also proposed a method to generate voxelized data using a network with three parts. The first part, a 2.5D sketch estimator, predicted the depth, surface normal and silhouettes of the input images, then, a 3D shape estimator inferred the object shape with the voxel representation, and finally, a re-projection consistency function enforced the alignment between the predicted 3D structure and the inferred 2.5 sketches. A big advantage of this method is that 3D reconstruction from 2.5 sketches can be trained with purely synthetic data because realistic 2.5D sketches are easy to obtain from a graph-

ics engine.

A downside of using voxel data, is that they are considered inefficient and less scalable, as the computation complexity grows cubically with the grid resolution, thus limiting the reconstruction to coarse representations that fail to recover fine details.

Figure 3.5: Hybrid grid-octree structure composed of several shallow octrees with a restricted maximal depth allowing for significant compression ratios.



Source: Riegler, Ulusoy and Geiger (2017) (RIEGLER; ULUSOY; GEIGER, 2017)

A way to increase the efficiency when working with voxels is to use octree representations (see Figure 3.5). An octree is a structure that divides the 3D space recursively into octants in such a way that cells that contain relevant information (as the surface boundary) are subdivided more times, while regions without information are represented by large cells (RIEGLER; ULUSOY; GEIGER, 2017). They have several drawbacks as the structure of the octree depends of the object so ideally the network needs to predict the structure and its content. Also, using convolution operations is easier when the input is a regular grid. Solutions as (WANG et al., 2017) execute the entire training and evaluation process in the GPU by efficiently storing all the network features and octant information in the graphical memory.

Besides from volumetric representations, 3D shapes can also be represented using only the information about the surface of the 3D object, avoiding unnecessarily voluminous data. However, common surface representations as meshes and point clouds are highly unstructured and this is a problem when working with CNN architectures. Han, Laga and Bennamoun (2019) classify the techniques used to work with surface data in three main categories: point-based, parameterization-based, and deformation-based methods.

A type of point-based representation are point clouds. These are a simple and unified structure, which makes them easier to learn (QI et al., 2017a). The leading works to

explore learning from point clouds were PointNet (QI et al., 2017a) and PointNet++ (QI et al., 2017b). Here, the authors developed architectures that processed point clouds and could be applied to object classification, part segmentation, and semantic scene labeling. Fan, Su and Guibas (2017) used them to create a model that directly generates points' coordinates using two main branches, one with flexibility to infer the complicated structures and other to maintain geometric continuity (see Figure 3.6). RealPoint3D by Xia et al. (2018) improved the obtained results by integrating prior shape knowledge into the network, enabling the reconstruction of images with complex backgrounds and captured from arbitrary viewpoints.

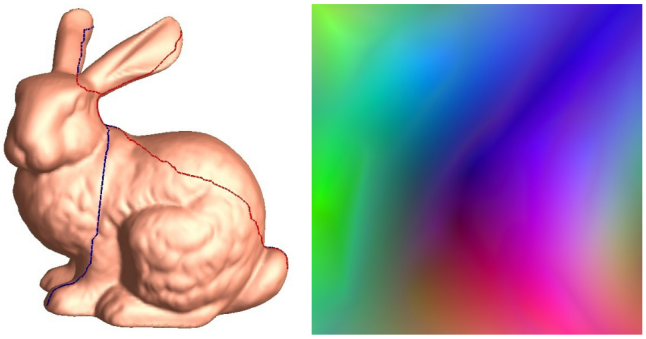
Figure 3.6: Input images and reconstructed 3D point clouds using Point Set Generation network.

Source: Fan, Su and Guibas (2017) (FAN; SU; GUIBAS, 2017)

Parameterization-based reconstruction (PUMAROLA et al., 2018; SINHA et al., 2017; GROUEIX et al., 2018) uses alternative representations of 3D surfaces instead or working directly with triangular meshes. The most commonly used are spherical parameterizations and geometry images (HAN; LAGA; BENNAMOUN, 2019). To obtain a geometry image of a surface it is necessary to cut it into disk-like charts and then unfold them into the 2D domain (see Figure 3.7). However, representing 3D objects within a category using geometry images is a difficult task, as all the surfaces need to be put into correspondence first and then the optimal cuts need to be calculated. Sinha et al. (2017) proposed a method to create consistent representations of objects inside a category and then used a deep learning approach to generate 3D surfaces said objects.

The final type of technique to work with 3D surface data are those based on deformations. Direct deformation of simple meshes using RNNs has been proposed (PAN et al., 2018). Additionally, vertex deformations between the target shape and a template mesh have been used (WANG et al., 2018; KANAZAWA et al., 2018). They assume same topology and one-to-one correspondence between the two meshes. Free-form deformation can also be employed to describe the shape of an object by representing it as

Figure 3.7: Original mesh with cuts and its generated geometry image.

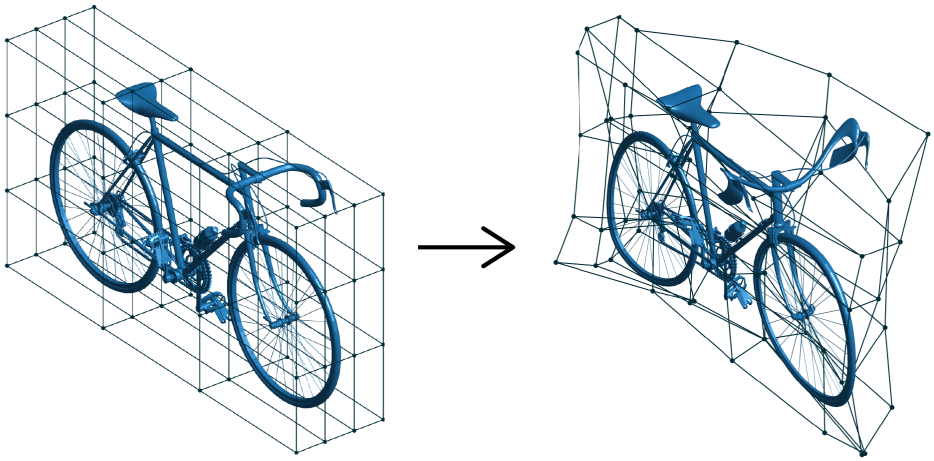


Source: Gu, Gortler and Hoppe (2002) (GU; GORTLER; HOPPE, 2002)

the deformation of a grid structure controlled by a set of points (see Figure 3.8). FFD techniques (JACK et al., 2018; KURENKOV et al., 2018; PONTES et al., 2018) infer the deformation parameters instead of inferring the whole shape. FFD is the type of representation we used in this work.

A great advantage of using this method is that by changing the template model, we can generate meshes with an arbitrary amount of points, as only the deformation parameters are calculated and not the actual point coordinates.

Figure 3.8: Free form deformation of a 3D mesh. To the left the original model and the deformation grid can be seen. To the right, the mesh is transformed by the deformation of the grid structure.



Source: Pontes et al. (2018) (PONTES et al., 2018)

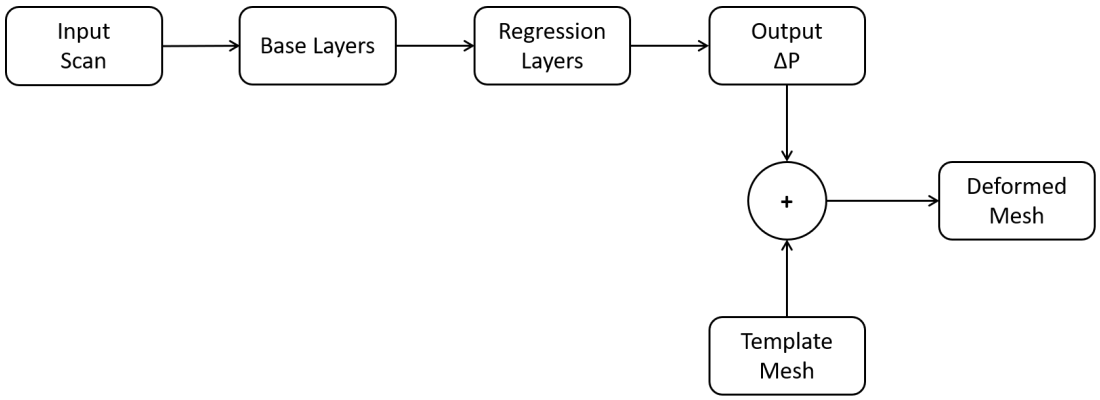
4 AUTOMATIC GENERATION OF PATIENT-SPECIFIC 3D MODELS OF ORGANS USING AN UNIFIED DEEP LEARNING APPROACH

In this section, we explain our technique for generating patient-specific meshes of organs. It is based in the work of Jack et al. (2018) to reconstruct 3D models from 2D images of objects, however, we adapted the method to use volumetric medical data as the input of our system. An overview of the method's steps can be seen in Figure 4.1.

Our problem consists in generating a mesh of an organ having the medical scan of the patient as an input. To achieve this, we trained a deep neural network model to learn the deformation we would need to apply to a template mesh in order to obtain a result that resembles the original shape of the organ.

Each part of the method is explained in more detail in Sections 4.2 - 4.4. Details about different tested approaches that did not produce good results will also be discussed. The final selected methodology can be found in Section 4.5.

Figure 4.1: Overview of our mesh generation method.



Source: Original image

4.1 Method Overview

The input of our system are 3D medical images whose information was mapped to a feature space using a CNN. These features were then mapped to a deformation parameter ΔP. To generate the deformed models, we computed the Bernstein decomposition $P' = BP$ of the template mesh and then, using equation 2.6, we calculated the deformed coordinate points of the template mesh.

To train our models, we used the obtained ΔP to deform the mesh and then com-

puted the error between the predicted and the ground truth meshes.

4.2 Input Data

There were multiple options to feed data into our model. The input representation would define important parts of the algorithm, for instance, the used architecture and preprocessing steps. The necessary definitions were:

- Representation of the medical scans;
- Ground truth meshes computed for each medical scan for the selected representation;
- Template meshes for the selected representation.

The data we worked with are the result of medical imaging techniques, specifically, CT and MRI scans. These scans can be interpreted as regular 2D images or as 3D images by stacking them together. We decided to use the 3D version of the data to benefit from the local information that exists around each voxel when it is analysed along its neighboring region.

Among the several options to represent 3D models presented on Section 2.2, we decided to use point clouds that derived from meshes as our ground truth. Hence, only the point cloud coordinates were processed but information about the original mesh connectivity could be stored and used later to connect the processed point cloud.

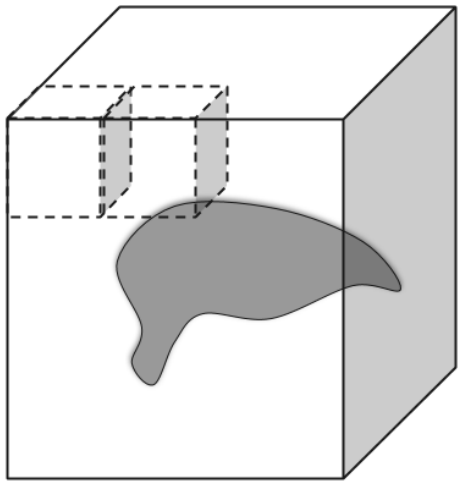
To feed the scan data to the network, the simplest option was to use the raw volumes as a whole. However, this represented a problem in several ways since the amount of available segmented medical images to train the model was low and, as 3D data grows exponentially, feeding the network with it would be expensive memory wise.

4.2.1 Cropped Subvolumes

One way to reduce the size of the input data of the network was to cut the original volume into several smaller cubes in such a way that each cube contains a piece of the original volume (see Figure 4.2). The first input data we used was composed of cropped volumes of size $20 \times 20 \times 20$. Before cropping the cubes, each scan was resampled to an isotropic resolution of $1 \times 1 \times 1\, mm$, so the used cubes had side of $20mm$. The extraction

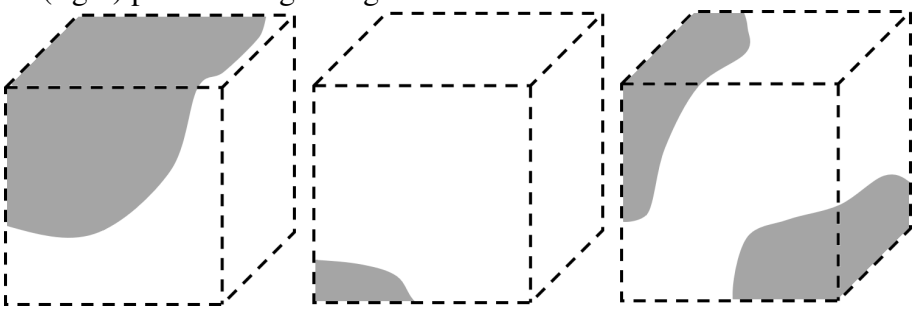
of the subvolumes was made without overlapping, meaning that each voxel of the scan appeared a single time the generated cubes.

Figure 4.2: Cropping subvolume cubes in a volumetric medical scan.



Source: Original image

Figure 4.3: Representation of cropped subvolumes containing big (left), small (center) and several (right) pieces of organ edge.



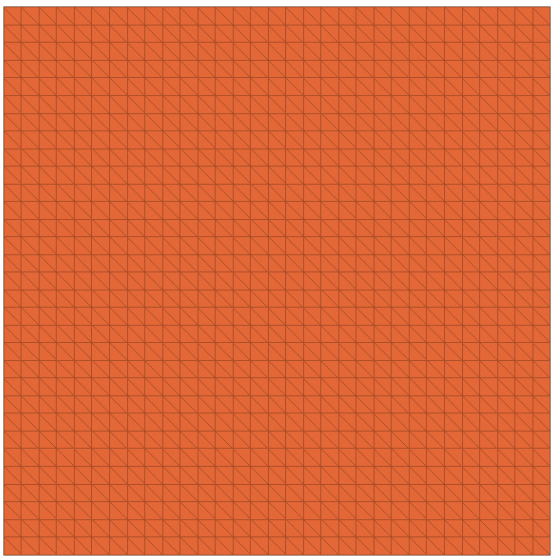
Source: Original image

The target information we wanted to infer from each cropped cube was, in the case of containing a piece of the organ's edge, the shape of that border. So, the selected template to deform for each subvolume was a triangular mesh of a plane represented by a set of vertex coordinates $V \in \mathbb{R}^{n_v \times 3}$ with $n_v = 1024$ as can be seen in Figure 4.4. This plane was placed inside the cube and then was deformed to fit the organ boundary.

The ground truth for each cropped subvolume was obtained by cropping the same region from the segmentation mask and then using Marching Cubes with Laplacian smoothing and decimation to generate a point cloud representing the organ edge. However, using the cropped subvolumes created a series of challenges:

- The cropped subvolumes could contain big or small pieces of organ edges (see

Figure 4.4: Plane template mesh with 1024 points used for the reconstruction of the organs with cropped subvolumes as input data.



Source: Original image

Figure 4.3 - left and center). Cubes with regions of edge much smaller than the template mesh need bigger deformations.

- The cropped subvolumes could contain no edges at all. This represented a problem as there was no direct way of deforming the template mesh to represent "no mesh" for the empty cubes.
- It also affected the ground truth point clouds, making their number of vertices vary greatly. Furthermore, for scan subvolumes without parts of the organ boundary, the number of vertices was zero.
- In organs with complex shapes such as the liver, some cropped cubes could have two or more pieces of edge (see Figure 4.3 - right). However, only one template mesh was used making the network have to learn the deformation to fit only one of the boundaries or maybe stay in the middle of both;
- Subvolumes inside the organ would also be considered as empty, making the amount of examples without edges much higher than those with edges, thus causing our training dataset to be imbalanced.

The imbalance problem was solved using random minority oversampling by copying samples containing edge until the dataset was balanced (BUDA; MAKI; MAZUROWSKI, 2017). Despite still having cubes with no edge, we proposed strategies to solve this during the training stage (see Section 5.2.1).

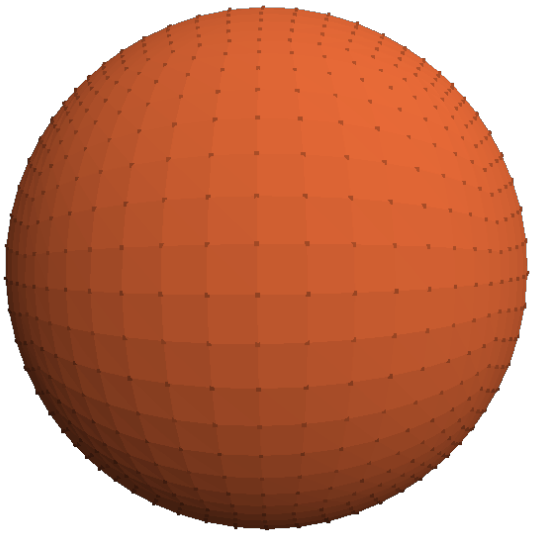
4.2.2 Whole Resized Volumes

To use the whole volumes without facing memory issues, we resampled all the scans to have different resolutions but a fixed size of $256 \times 256 \times 128$ using Nearest-Neighbor interpolation. As this left us with a very low amount of training examples, data augmentation was executed at the training stage as in (MILLETARI; NAVAB; AHMADI, 2016). This was done by performing random deformations in the scans using a deformation field obtained through a $2 \times 2 \times 2$ grid of control points that was modified with random values obtained from a Gaussian distribution with zero mean and a standard deviation of 15 pixels. Also, each scan had its intensity distribution matched to that of another training example in the dataset using histogram matching.

As using the whole volumes allowed us to process the organ without dividing it, the information to infer from the input was the deformation to create the whole organ shape and, with it, the plane template used for cropped cubes was no longer useful.

For this case, we used a spherical template mesh represented by a set of vertex coordinates $V \in \mathbb{R}^{n_v \times 3}$ with $n_v = 1538$, were each vertex v_i was a member of two intersecting orthogonal contours. Thus, each vertex was connected to exactly four vertices creating a quad-sphere structure as can be seen in Figure 4.5.

Figure 4.5: Quad sphere mesh with 1538 vertices used as the template mesh for the reconstruction of the organs with whole resized volumes as input data.



Source: Original image

The ground truth for the whole volumes was obtained from the whole resized segmentation mask using Marching Cubes with Laplacian smoothing and decimation. A sec-

ond ground truth version having correspondence with the template mesh was calculated from the segmentation masks as explained in the next section.

4.2.3 Ground Truth Correspondence Resampling

As we saw before, the point clouds generated as ground truth for the training process may have a variable number of points in them. Chamfer Distance between the ground truth and the deformed template mesh could be calculated without a problem as the metric does not require both point clouds to have the same size. Yet, we wanted to analyse the effect of deforming the points by minimizing the difference between each point and its correspondent point in the ground truth and to do this, we needed to have one to one correspondence between both point clouds.

To achieve it, we generated a second ground truth version for the whole resampled volume by doing the following: first, we generated the mesh from the segmentation mask without using decimation. Then, we found the center of mass of the mesh. Next, we intersected the mesh with rays originating at the center of mass and projecting in the direction of each point in the sphere mesh. Finally, we saved the intersection points of the rays with the mesh.

As the mesh is not convex, it could be intersected several times along one ray. Then, only the furthest point was selected. Notice that this algorithm only works when the center of mass lies inside the mesh.

4.3 Output Data

Defining the format of the output data also defined the final layers of our architecture. In Section 2.3, we saw that FFD can be used to deform any given 3D mesh by modifying the ΔP offsets of some control points. As a result, it was possible to train our network to generate the deformation parameters (offset of the control points) instead of having it generate the raw points.

The output data was then defined by N control points $P_{ijk} \in \mathbb{R}^{N \times 3}$ that could be used to deform the template meshes, where $N = (l+1)(m+1)(n+1)$, with l, m, and n define the grid of control points.

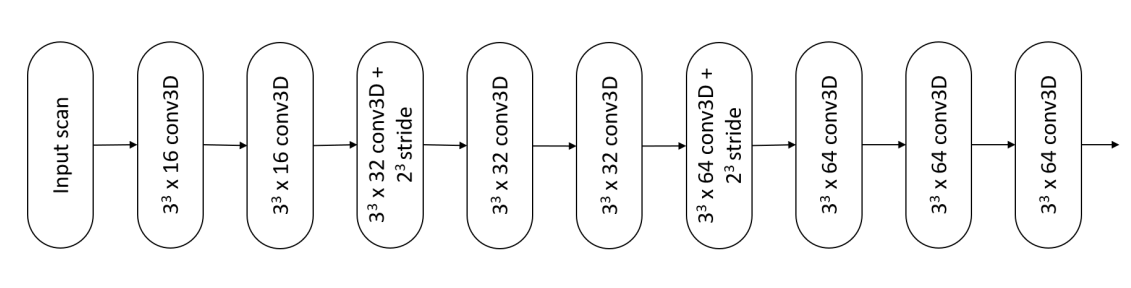
4.4 Network Architecture

As in Jack et al. (2018) work, our network is composed of two main parts, one for obtaining information from the input images and the other to perform the regression of the deformation parameters. We used an existing network architecture to achieve the former and added several fully connected layers at the end of it to achieve the latter.

4.4.1 Base Architecture

First, we used a simple 3D base architecture that was composed of 3 convolutional layers with 3 convolutions each, the last one having stride=2 to act as a pooling layer. The model structure can be seen in Figure 4.6.

Figure 4.6: Structure of our simple base architecture.



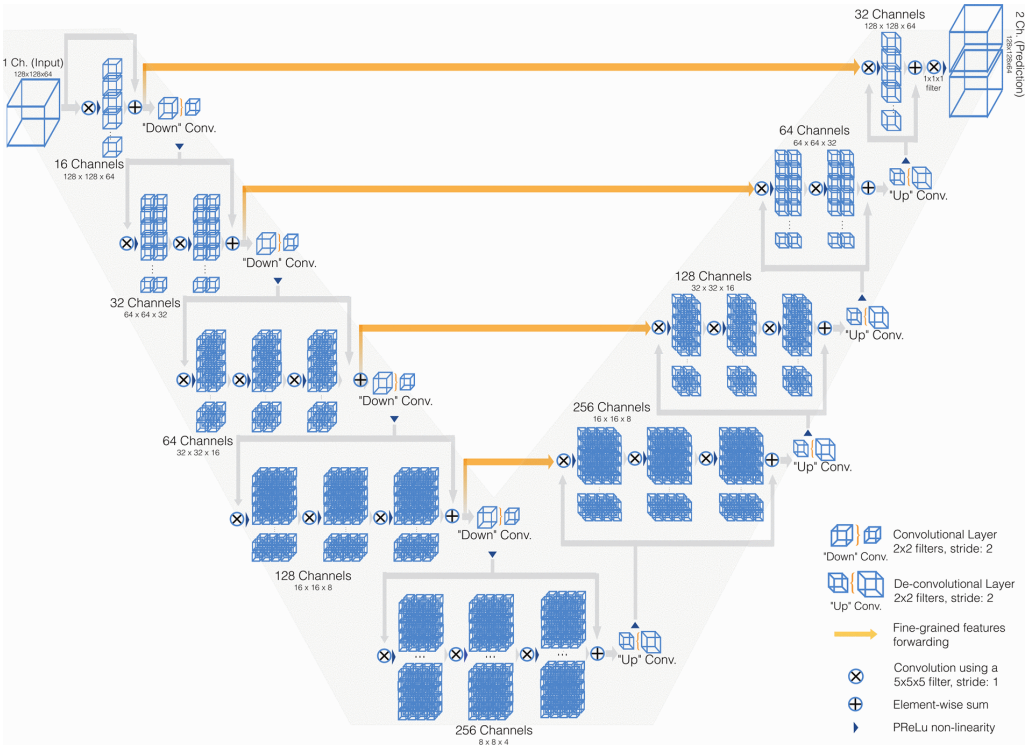
Source: Original image

However, as our input was volumetric medical data, we needed a 3D base architecture that could extract meaningful information about the organs' shape. With this in mind, we chose to use a V-Net model (MILLETARI; NAVAB; AHMADI, 2016) originally meant to segment prostate data from MRIs. The architecture can be seen in Figure 4.7.

4.4.2 Regression Layers

At the end of the base model we added several fully connected layers, the final one outputting the parameters of ΔP. The configuration of the regression layers changed with the different trained models. Details can be seen in Table 5.2.

Figure 4.7: V-Net base architecture structure.

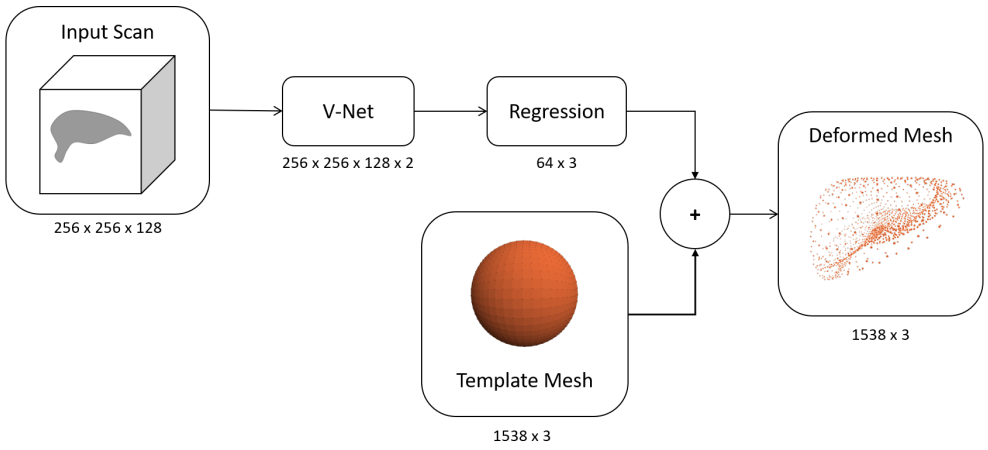


Adapted from: Milletari, Navab and Ahmadi (2016) (MILLETARI; NAVAB; AHMADI, 2016)

4.5 Final Model

The structure and dimensions of our final working model can be seen in Figure 4.8. Its input is composed of whole resampled volumes that are fed to a V-Net model, followed by a regression model that outputs the deformation parameters used to deform the quad-sphere template mesh (Figure 4.5).

Figure 4.8: Structure of the final working model.



Source: Original image

Before feeding the data to the network, we rescaled each medical image intensity

to values between 0 and 1. Then, we augmented the input data as explained in Section 4.2.2, and finally, we performed a standard feature scaling process normalizing each scan by subtracting its mean and dividing by its standard deviation as in (MILLETARI; NAVAB; AHMADI, 2016). The rest of the parameters for each trained model can be found in Chapter 5.

5 EXPERIMENTS AND RESULTS

In this chapter, we will explain the different experiments we performed and present the outcomes from each of them. The final results obtained after refining the methods can be seen in Section 5.3 and Section 5.4. All the experiments shown in this chapter use FFD with $l = m = n = 3$ for a total of 64 control points to deform the template meshes. During all the training processes only the model with the lowest development cost was saved.

The experiments were conducted on a PC with Intel Core i7-8700 CPU @ 3.20 GHz, a 16 GBytes memory, and NVIDIA Quadro P6000. The proposed system was implemented using TensorFlow 1.12.

5.1 Datasets

To evaluate our method, we used datasets of two organs with different levels of shape complexity, the liver and the prostate.

5.1.1 Liver Data

For the liver we used 40 CT scan images, publicly available in the databases Sliver07 [1] (Heimann et al., 2009) and 3Dircadb [2]. From the Sliver07 dataset we used 20 contrast-enhanced images with pixel spacing between 0.57 and 0.8mm, distance between slices varying from 0.7 to 5mm, and number of slices from 64 to 394. The remaining 20 scans came from the 3Dircadb database. The pixel spacing varying from 0.56 to 0.87mm, distance between slices from 1 to 4mm and number of slices from 74 to 260. All the scan images had in-plane resolution of 512x512. Most of the CT images were pathological, including tumors. The segmentation data was created manually by expert radiologists in a slice by slice manner.

We split the data into train, development, and test sets having 32, 4, and 4 liver scans, respectively.

[1] <https://sliver07.grand-challenge.org/>
[2] <https://www.ircad.fr/research/3dircadb/>

5.1.2 Prostate Data

For the prostate we used 80 transversal T2-weighted MR images from the PROMISE12[3] dataset (LITJENS et al., 2014). Data included both benign disease and prostate cancer. Pixel spacing varied between 0.25 and 0.75mm while distance between slices was between 2.2 and 4mm. The data was divided in the same way, resulting in 40, 10, and 30 prostate scans for train, development, and test sets, respectively.

5.2 Generating 3D Models of the Liver Using Cropped Subvolumes

Our first approach to generate liver volumes used the cropped subvolumes input data type (see Section 4.2.1). As discussed before, this approach generates training examples with and without liver boundaries and as our method deforms the template mesh to fit said boundary, it would be necessary to deform it until it represents the "no mesh" status that occurs when there is no organ boundary present. To overcome this, we explored the possibility of dividing the problem into a classification + regression model, where the classification part would determine if the training example contained a part of the boundary, and the regression part would receive only those examples that actually contain organ edges, so the template mesh can be deformed to fit an existing shape. Both networks were trained separately to assess their feasibility.

5.2.1 Classification of cropped subvolumes with and without liver edges

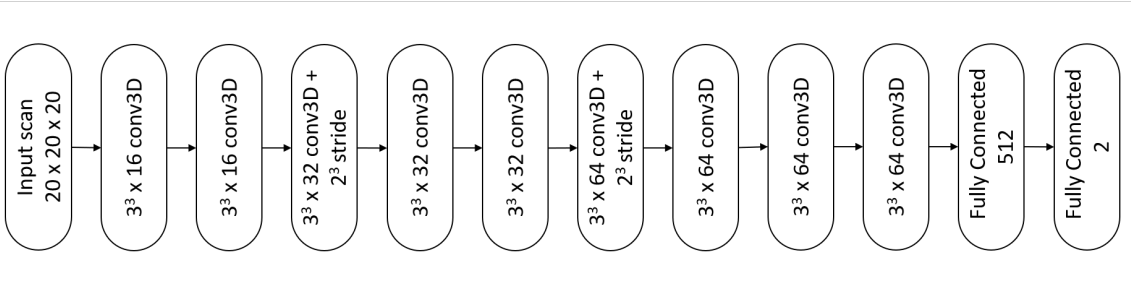
The model selected to classify the liver subvolumes into those containing a boundary of the liver and those not containing it can be seen in Figure 5.1. The data used to train this model were the cropped subvolumes as input with their corresponding binary labels obtained from the original segmentation masks. This model was similar to our simple base architecture having 3 convolutional layers, followed by a fully connected layer and a final layer with 2 neurons, one for each class. All the network activation functions were ReLU and we applied L_2 regularization to our model with a λ penalty term. A batch size of 32 was used. To define the model hyperparameters, we did several exploratory training iterations to define suitable values and then carried out a grid search process with the

[3] <https://promise12.grand-challenge.org/>

following parameters, yielding a total of 12 models, each one trained during 55 epochs:

- Learning rate = [1×10^{-4}, 5×10^{-5}]
- Learning rate decay = [0.2, 0.5]
- λ = [0.01, 0.008, 0.005]

Figure 5.1: Model used to classify cropped subvolumes into those with and without liver boundary.



Source: Original image

To evaluate our models, we calculated accuracy, precision, recall, and F1 score in the test dataset as follows: The resulting values can be seen in Table 5.1.

$$accuracy = \frac{true\,positive + true\,negative}{total} \tag{5.1}$$

$$precision = \frac{true\,positive}{true\,positive + false\,positive} \tag{5.2}$$

$$recall = \frac{true\,positive}{true\,positive + false\,negative} \tag{5.3}$$

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{5.4}$$

5.2.2 Regression of cropped subvolumes containing liver edges

To test the regression process we created a dataset that contained only cropped subvolumes with liver boundaries. This model had the same architecture as the model described in Section 5.2.1, only changing the last layers to predict the ΔP parameters used for the mesh deformation (see Figure 5.2). All the network activation functions were ReLU and the model was trained using batch size of 32 during 1000 epochs. The

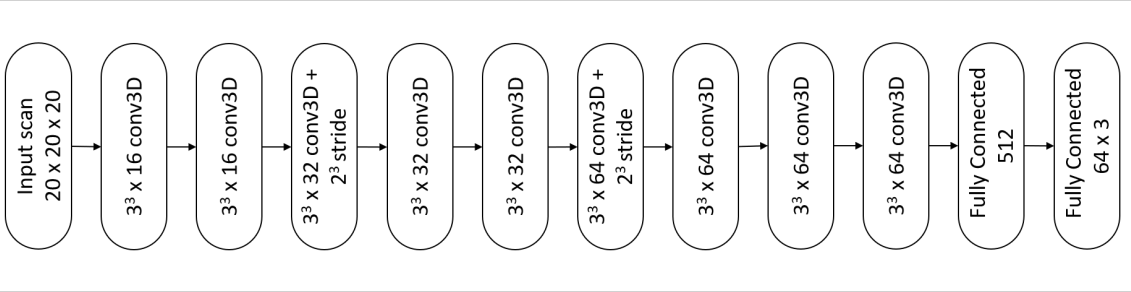
Table 5.1: Results of grid search for hyperparameters optimization in classification of liver subvolumes.

	LR	LR decay	λ	Accuracy	Precision	Recall	F1
1	1×10^{-4}	0.2	0.01	0.88	0.97	0.89	0.93
2	1×10^{-4}	0.2	0.008	0.90	0.96	0.93	0.95
3	1×10^{-4}	0.2	0.005	0.92	0.97	0.94	0.95
4	1×10^{-4}	0.5	0.01	0.91	0.96	0.94	0.95
5	1×10^{-4}	0.5	0.008	0.91	0.97	0.93	0.95
6	1×10^{-4}	0.5	0.005	0.91	0.97	0.94	0.95
7	5×10^{-5}	0.2	0.01	0.91	0.97	0.93	0.95
8	5×10^{-5}	0.2	0.008	0.91	0.97	0.93	0.95
9	5×10^{-5}	0.2	0.005	0.91	0.97	0.93	0.95
10	5×10^{-5}	0.5	0.01	0.90	0.97	0.91	0.94
11	5×10^{-5}	0.5	0.008	0.91	0.97	0.93	0.95
12	5×10^{-5}	0.5	0.005	0.92	0.97	0.94	0.95

learning rate was 0.0001 and we used Adam optimizer with the default values $\beta 1 = 0.9$, $\beta 2 = 0.999$ and $\epsilon = 10^{-8}$.

Even though we trained several models and this one had the lowest cost, the actual results from the regression were very poor as seen in Figure 5.3. These results encouraged us to try a different approach, either by changing the base model or by using a different representation of the input data, as the one proposed in Section 4.2.2.

Figure 5.2: Model used to generate the meshes from cropped subvolumes containing liver edges.



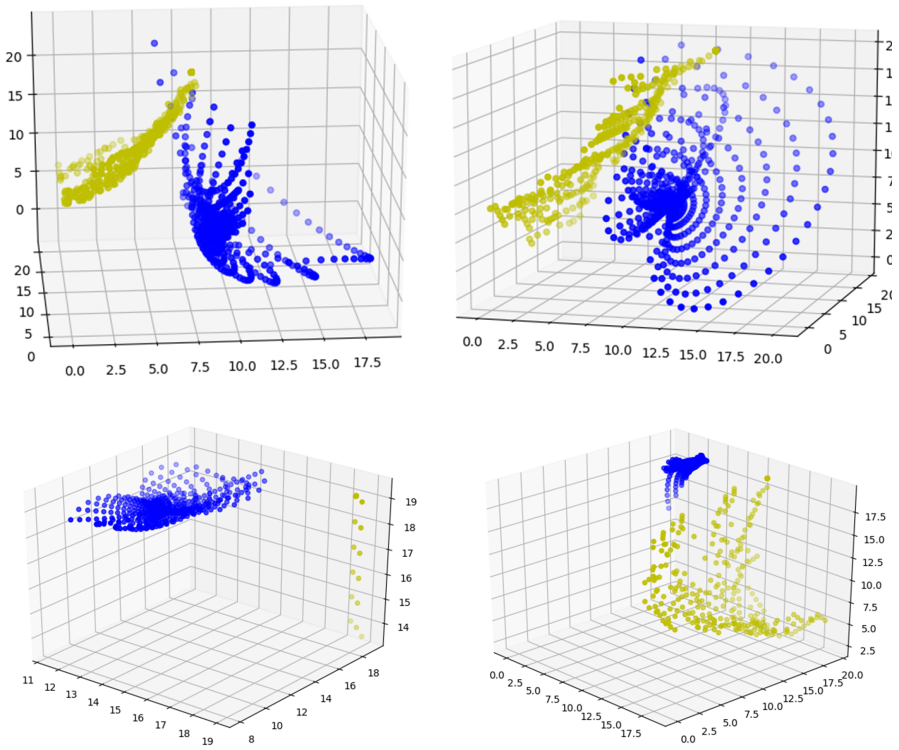
Source: Original image

5.3 Generation of Liver Models Using Whole Volumes

To generate our liver models we followed the approach explained in Section 4.5. However, the base layers of the architecture had its weights initialized from the convolutional layers of a network trained to segment the liver.

Details about the configuration of the regression layers of the model are in Table 5.2. We trained our model using Adam optimizer with $\beta 1 = 0.99$, $\beta 2 = 0.999$, and $\epsilon = 10^{-8}$, and a learning rate of 10^{-4} that decayed by 0.1 every 50 epochs during 200

Figure 5.3: Predictions generated with the regression model using only cropped subvolumes containing liver edges. Yellow points are the ground truth and blue points are the deformed plane.



Source: Original image

epochs with batch size of 1. The cost function we used to measure the error between the generated models was the Chamfer Distance.

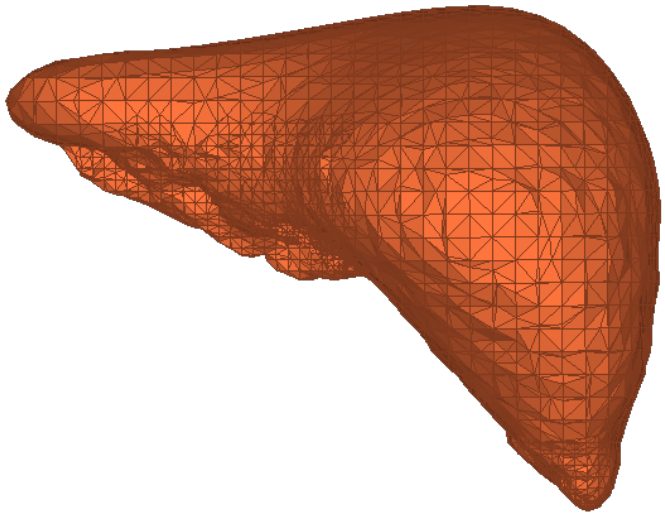
The resulting meshes predicted using this model are in Figure 5.9. The image shows, in the first column, the mesh generated directly from the ground truth segmentation of the original scans. The second and third column show the triangular mesh and point cloud obtained after applying Laplacian decimation and smoothing to the original meshes. This decimation process was made in order to reduce the number of points and the computational cost. The third column shows the point cloud generated by the algorithm. All the shown images have the same scale, reflecting the original liver size.

We can notice that for clearly different input scans, the generated deformations are very similar. The system learnt to create an overall smoothed liver structure without preserving specific shape details.

Figure 5.4 shows the mesh for one of the predicted liver point clouds generated by the algorithm. The mesh was created by obtaining the normals for the point cloud and then applying Screened Poisson surface reconstruction, resulting in a mesh with 5287 vertices and 10574 faces. The original connectivity of the template mesh could not be

used as the deformation caused the mesh to self intersect.

Figure 5.4: Mesh of a liver predicted by the algorithm. The mesh was created applying Screened Poisson surface reconstruction to the point cloud generated by the system.



Source: Original image

5.3.1 Pre-training the Base Architecture for a Segmentation Task

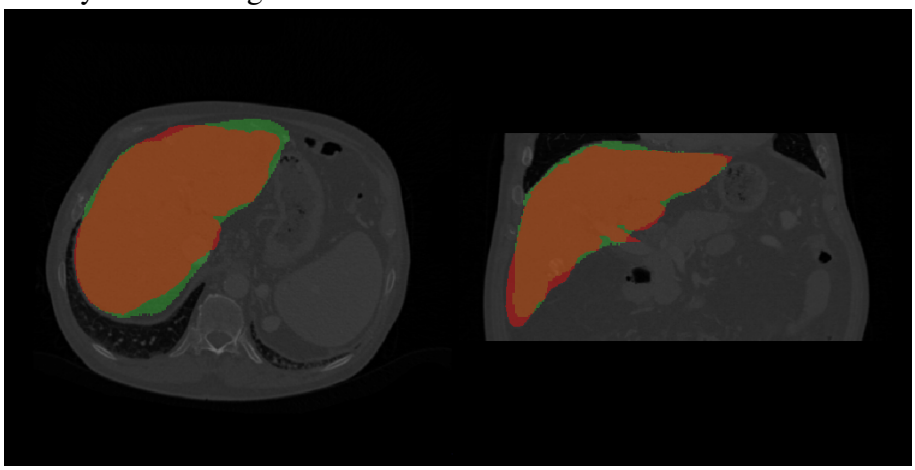
To train a model to segment the liver, we used the V-Net architecture as implemented by Gibson et al. (2018) and followed the same steps described when working with whole resized volumes. Our cost function was the Dice Loss coefficient as described by Milletari, Navab and Ahmadi (2016):

$$D = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2} \tag{5.5}$$

where the sum runs over the N voxels of the predicted volume $p_i \in P$ and ground truth $g_i \in G$.

This model was trained using Adam optimizer with $\beta 1 = 0.99$, $\beta 2 = 0.999$, and $\epsilon = 10^{-8}$, and a learning rate of 10^{-4} that decayed by 0.1 every 25 epochs during 100 epochs with batch size of 1. Although the achieved segmentation results were not completely precise, it was a good starting point for the weights of our regression model. Examples of the segmented results can be seen in Figure 5.5.

Figure 5.5: Results generated by the segmentation model whose weights were used to initialize the layers of the regression architecture.



Source: Original image

5.4 Generation of Prostate 3D Models

The generation of 3D models for the prostate was only tested using the whole resampled volumes as the network input. As mentioned in Section 4.5, the base architecture used was V-Net, and the detailed information about the regression layers can be found in Table 5.2. We used Adam optimizer to carry out the training, with $\beta 1 = 0.99$, $\beta 2 = 0.999$, and $\epsilon = 10^{-8}$. The learning rate was 10^{-4} and decayed by half every 50 epochs during 200 epochs with batch size of 1. This model was trained with and without mesh correspondence, using both Chamfer and MSD as the cost function, and with and without loading pretrained weights. The configuration that gave the best visual results was the one trained with mesh correspondence between the ground truths and the sphere mesh but using Chamfer Distance as the cost function, and trained in an end-to-end manner without using pretrained weights.

Figure 5.10 shows in the first column the mesh generated directly from the ground truth segmentation of the prostate original scans. The second column shows the point clouds sampled from the original meshes. Each one of the points in these clouds have one-to-one correspondence with the points of the template mesh. The third column shows the point cloud generated by the algorithms. All the shown images have the same scale, reflecting the original prostate size.

Results generated for the prostate scans presented the same issues discussed in Section 5.3. However, qualitative results suggest that more details were preserved, leading to a better fit to the overall original shape and size. A possible reason for this, is that as

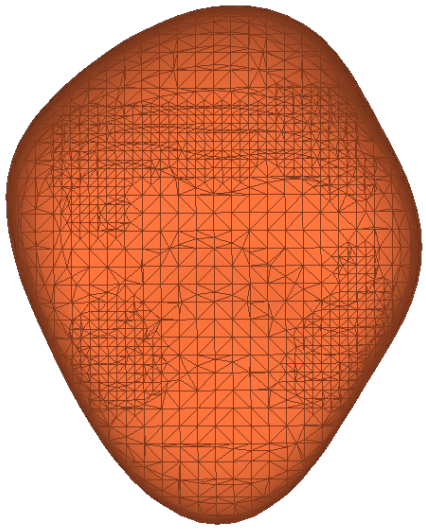
Table 5.2: Summary of layers for our two main liver and prostate models. The "-" sign indicates that the layer was not present in the architecture.

Layer	**Liver**	**Prostate**
Input	256, 256, 128, 1	256, 256, 128, 1
V-Net	256, 256, 128, 2	256, 256, 128, 2
Downsample Convolution	128, 128, 64, 1	64, 64, 32, 1
Downsample Convolution	128, 128, 64, 1	-
Flattened	131072	131072
Fully Connected	1024	1024
Fully Connected x 2	2048	2048
Fully Connected	1024	1024
Fully Connected	192	192
ΔP	64, 3	64, 3

the prostate has a more homogeneous organ shape, the deformation needed to transform a sphere into a prostate is less than that to generate a liver.

Figure 5.6 shows the mesh for one of the predicted prostate point clouds generated by the algorithm. The mesh was created by obtaining the normals for the point cloud and then applying Screened Poisson surface reconstruction, resulting in a mesh that had 4190 vertices and 8376 faces. The original connectivity of the template mesh could not be used as the deformation caused the mesh to self intersect.

Figure 5.6: Prostate mesh for one of the predicted point clouds generated by the algorithm. It was created using Screened Poisson surface reconstruction.



Source: Original image

5.5 Cost Function Analysis

In order to verify if the training was possible, specifically if the amount of selected control points could deform the sphere mesh to resemble a complex organ as the liver, we carried out a test to directly learn the deformations of each sample in the training data. We also used this test to compare the behaviour of different cost functions.

The test can be explained as follows: first we created a neural network with a single fully connected layer which number of neurons was 192, i.e, 3 coordinates for each one of the 64 control points. Using those values as the deformation parameters, we deformed the sphere template. Then, using Adan optimizer we minimized the error metric between the deformed sphere and the liver original point cloud. We trained a model for each sample in the training dataset during 100000 epochs.

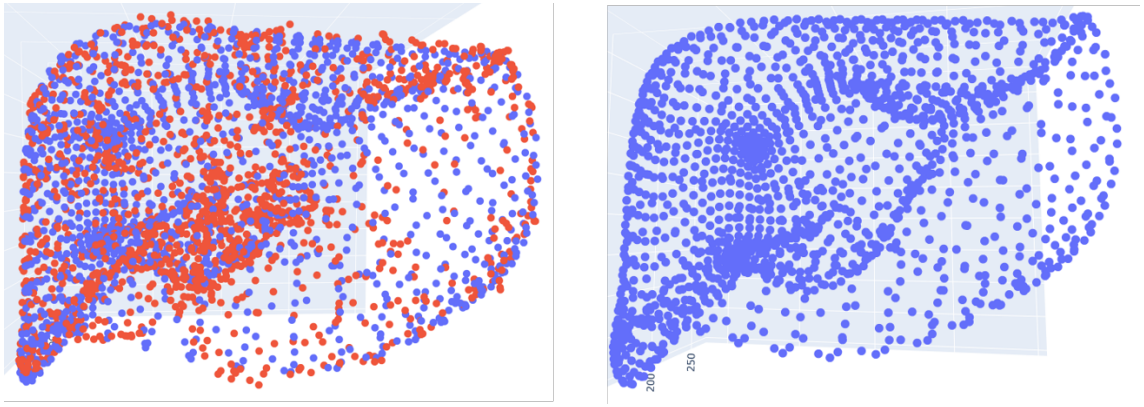
We repeated this process several times under the following conditions:

- Ground truth for the whole volumes as described in Section 4.2.2, and Chamfer Distance as cost function (see Figure 5.7);
- Resampled ground truth (see Section 4.2.3) and Chamfer Distance as cost function (see Figure 5.8 - left);
- Resampled ground truth and the maximum of the squared distance as cost function (see Figure 5.8 - center);
- Resampled ground truth and MSD as cost function (see Figure 5.8 - right).

Visual analysis of the results yielded the following observations:

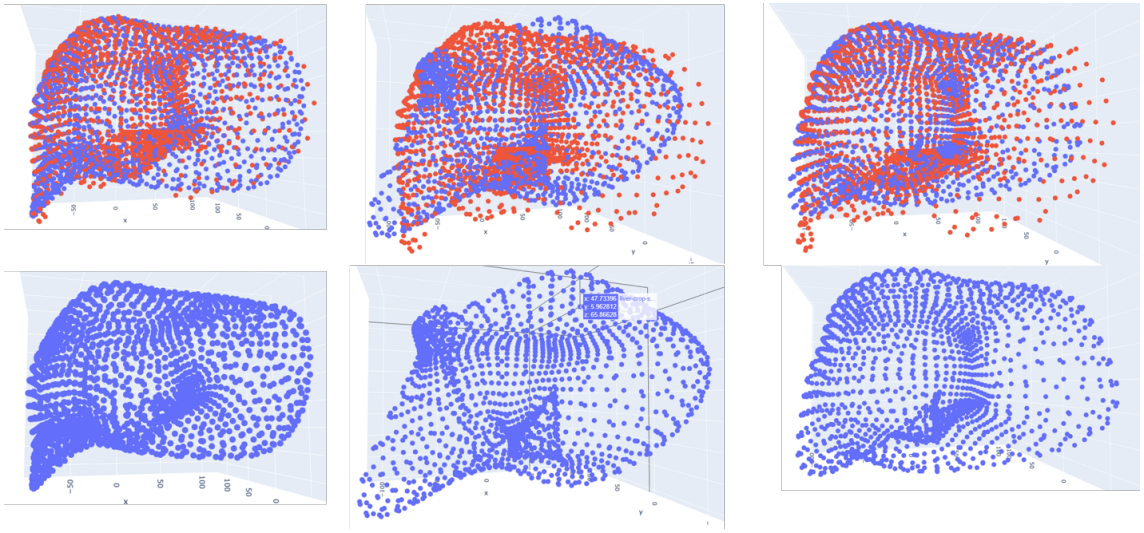
- The number of control points selected for the FFD process (64) deformed the template to fit well the original mesh. However, most of the details were lost.
- Chamfer distance was the metric that better fitted the template to the original shape, preserving well the shape but loosing sharp details.
- MSD created shapes that were too smoothed, maintaining a general shape but loosing the specific characteristics of the organ.
- The maximum of the squared distance failed to properly fit the original shape and created artifacts in the template mesh.

Figure 5.7: To the left, comparison of a liver's ground truth mesh (red) and the deformed template mesh using 64 control points for the FFD (blue). To the right, only the deformed template mesh (blue) is displayed.



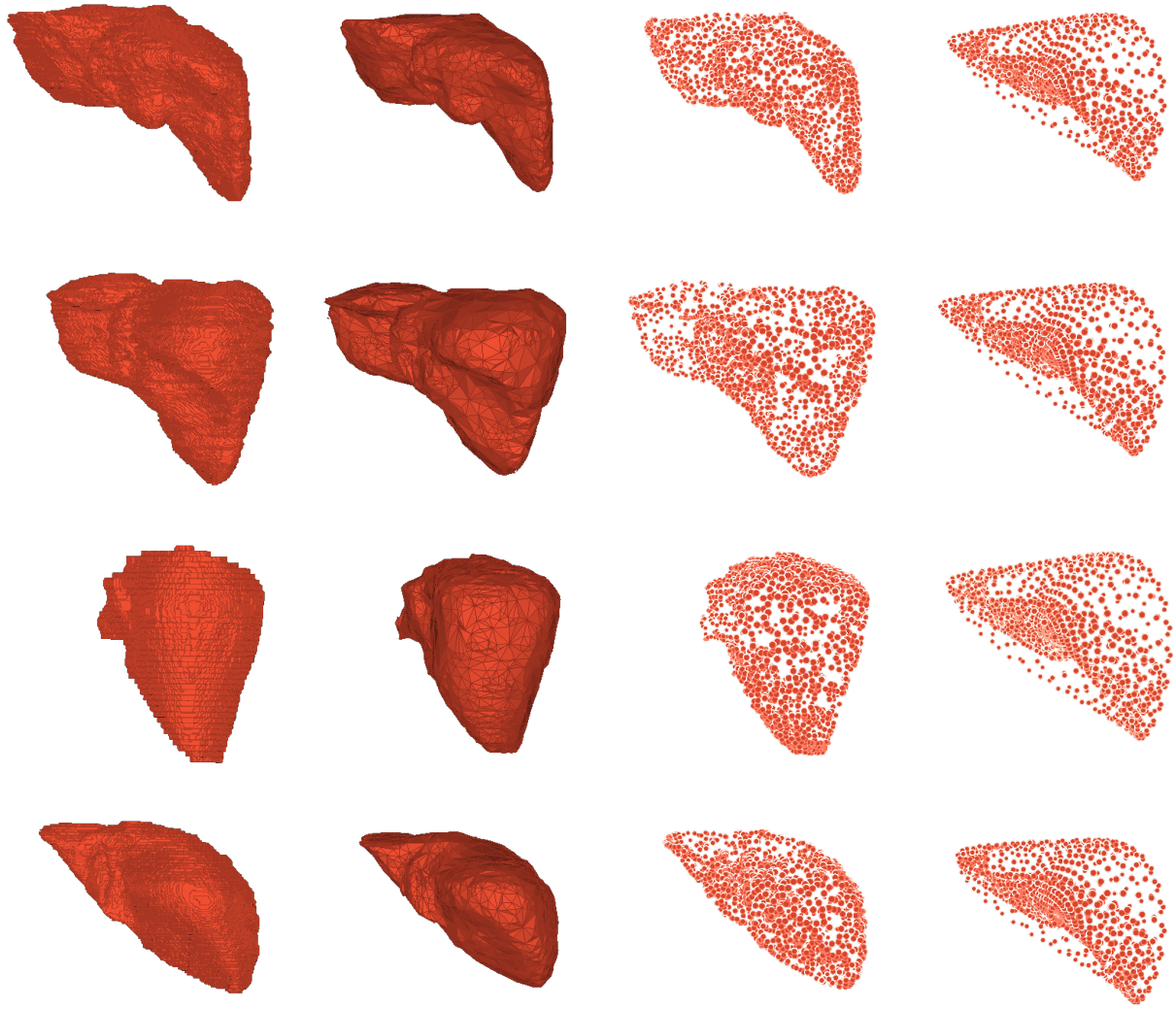
Source: Original image

Figure 5.8: Above, comparison of a liver resampled mesh (red) and the deformed template mesh using 64 control points for the FFD (blue). Below, only the deformed template mesh (blue) is displayed. Columns depict the results obtained when using Chamfer Distance (left), maximum of the squared distance (center), and mean squared distance (right) as cost functions, respectively.



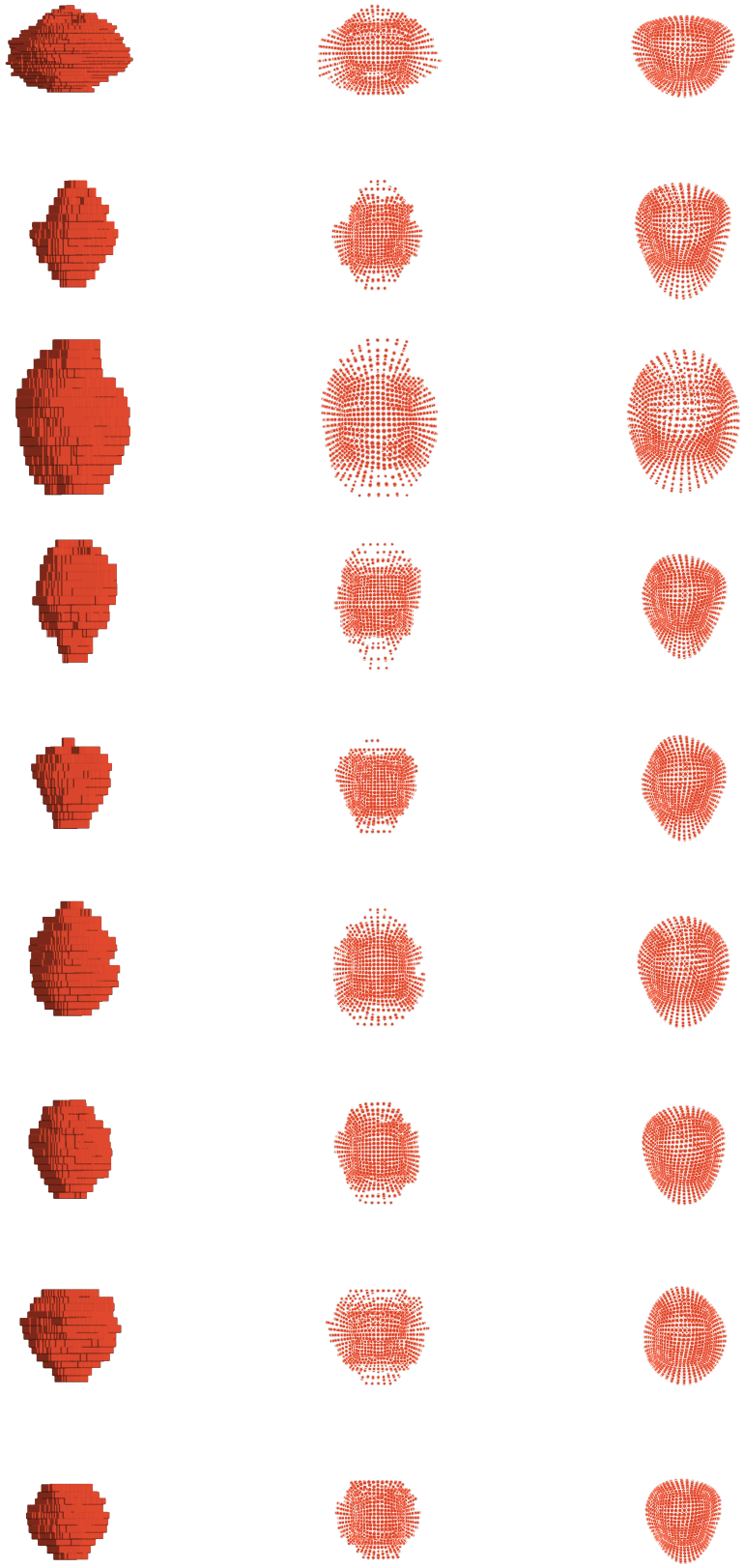
Source: Original image

Figure 5.9: Visual results for the liver: mesh generated with marching cubes from the original segmentation (left), triangular mesh of the decimated and smoothed original mesh (center left), point cloud of the decimated and smoothed original mesh that was used as the ground truth (center right),and predicted point clouds (right).



Source: Original image

Figure 5.10: Visual results for the prostate: shape generated with marching cubes from the original segmentation (left), sampled point cloud correspondence with the template mesh (center), and predicted point clouds (right).



Source: Original image

6 CONCLUSION

In this thesis, we presented a method to generate 3D models of organs from different medical imaging techniques (CT and MRI) using a fully automatic deep neural network architecture. Our algorithm is based on the deformation of template meshes using free-form deformation techniques. The novelty of this work consists in a single step reconstruction algorithm that, for the first time, combines a deformation technique with deep learning architectures to build patient specific anatomical geometry.

Considering the specific objectives of this work (Section 1.1.1) we can state that they were partially achieved. A deep theoretical understanding of the subject was applied throughout the algorithm development. A number of experiments allowed us to learn the strengths and weaknesses of the techniques employed, as well as to better define and understand the problem of anatomy reconstruction. The reconstructed 3D models generated by our network were coherent with the overall shape of the organs demonstrating that it can be learnt, even when details are not preserved.

Several limitations affected the level of detail that could be achieved with our approach. The first one resided in the use of FFD. As this technique has limitations in the scale of the deformations, having a single resolution for the deformation grid does not allow to reach the high levels of detail required for complex organs such as the liver.

The second important limitation was the cost function used to train our network. Even though the Chamfer distance is a common metric used in object reconstruction problems, it does not properly penalizes the lack of sharp details in the shapes. Moreover, Chamfer distance itself as a cost function does not take into account the original connectivity of the template mesh, generating a deformation where it was not possible to directly recover the mesh from the point clouds as the results were often self intersecting meshes.

As shown in Section 5.5, the deformation of the chosen templates using FFD and Chamfer Distance in addition to the use of a learning approach, generated results with a considerable loss of detail, even when the overall shapes were highly maintained.

Future work that might improve these limitations includes the use of more advanced deformation techniques, as hierarchical approaches of FFD where finer deformation grids are used to provide local refinability. Furthermore, it is possible to add mesh quality terms to the cost function to maintain a good structure of the point clouds during the deformation process, avoiding the artifacts generated by the use of Chamfer Distance alone.

Another change that could improve the fidelity of the reconstructed meshes, when compared to the original ones, is to part from a template mesh more similar to the target organ, like a mean model of it instead of a sphere. Is also possible to allow the network to choose the template mesh among different options as proposed by Jack et al. (2018).

Finally, future works include training the networks with different datasets, particularly those of organs that are easier to segment because of their high contrast from the background in the medical scans, as in bone structures. This seems to be a necessary step-back to isolate the two fundamental problems of learning the shapes when they are visible in the images and learning the shapes when they are not visible but can be inferred.

The long term goal of anatomy reconstruction research is to develop cost-effective algorithms that can be seamlessly integrated in the clinical workflow. Arguably, machine learning techniques have an important role to play if they can be combined with graphics algorithms.

REFERENCES

BERECIARTUA, A. et al. 3D active surfaces for liver segmentation in multisequence MRI images. **Computer Methods and Programs in Biomedicine**, Elsevier Ireland Ltd, v. 132, p. 149–160, 2016. ISSN 18727565. Available from Internet: <http://dx.doi.org/10.1016/j.cmpb.2016.04.028>.

BEUTEL, J.; KUNDEL, H. L.; METTER, R. L. V. **Handbook of medical imaging**. [S.l.]: Spie Press, 2000.

BISHOP, C. M. **Pattern recognition and machine learning**. [S.l.: s.n.], 2006.

BUCKI, M.; LOBOS, C.; PAYAN, Y. A fast and robust patient specific Finite Element mesh registration technique: Application to 60 clinical cases. **Medical Image Analysis**, v. 14, n. 3, p. 303–317, jun 2010. ISSN 13618415. Available from Internet: <http://linkinghub.elsevier.com/retrieve/pii/S1361841510000186>.

BUDA, M.; MAKI, A.; MAZUROWSKI, M. A. A systematic study of the class imbalance problem in convolutional neural networks. p. 1–23, 2017. Available from Internet: <http://arxiv.org/abs/1710.05381>.

CAMPBELL, M.; HOANE, A.; HSU, F. hsiung. Deep blue. **Artificial Intelligence**, v. 134, n. 1, p. 57 – 83, 2002. ISSN 0004-3702. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0004370201001291>.

CARDENAS, C. E. et al. Advances in Auto-Segmentation. **Seminars in Radiation Oncology**, v. 29, n. 3, p. 185–197, 2019. ISSN 1053-4296. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S1053429619300104>.

CHOY, C. B. et al. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: SPRINGER. **European conference on computer vision**. [S.l.], 2016. p. 628–644.

ÇIÇEK, Ö. et al. 3d u-net: learning dense volumetric segmentation from sparse annotation. In: SPRINGER. **International conference on medical image computing and computer-assisted intervention**. [S.l.], 2016. p. 424–432.

FAN, H.; SU, H.; GUIBAS, L. J. A point set generation network for 3d object reconstruction from a single image. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2017. p. 605–613.

Feng Ding et al. 3D segmentation of soft organs by flipping-free mesh deformation. In: NATIONAL UNIVERSITY OF SINGAPORE. **2009 Workshop on Applications of Computer Vision (WACV)**. IEEE, 2009. p. 1–7. ISBN 978-1-4244-5497-6. Available from Internet: <http://ieeexplore.ieee.org/document/5403096/>.

GELERNTER, H. Realization of a geometry theorem proving machine. In: **IFIP Congress**. [S.l.: s.n.], 1959.

GIBSON, E. et al. Niftynet: a deep-learning platform for medical imaging. In: . [s.n.], 2018. ISSN 0169-2607. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0169260717311823>.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GROUEIX, T. et al. AtlasNet: A Papier-M\ˆach\’e Approach to Learning 3D Surface Generation. 2018.

GU, X.; GORTLER, S. J.; HOPPE, H. Geometry images. In: ACM. **ACM Transactions on Graphics (TOG)**. [S.l.], 2002. v. 21, n. 3, p. 355–361.

HAN, X.-F.; LAGA, H.; BENNAMOUN, M. Image-based 3d object reconstruction: State-of-the-art and trends in the deep learning era. **arXiv preprint arXiv:1906.06543**, 2019.

HARTLEY, R.; ZISSERMAN, A. **Multiple view geometry in computer vision**. [S.l.]: Cambridge university press, 2003.

Heimann, T. et al. Comparison and evaluation of methods for liver segmentation from ct datasets. **IEEE Transactions on Medical Imaging**, v. 28, n. 8, p. 1251–1265, Aug 2009.

JACK, D. et al. Learning free-form deformations for 3d object reconstruction. In: SPRINGER. **Asian Conference on Computer Vision**. [S.l.], 2018. p. 317–333.

KANAZAWA, A. et al. Learning category-specific mesh reconstruction from image collections. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2018. p. 371–386.

KASS, M.; WITKIN, A.; TERZOPOULOS, D. Snakes: Active contour models. **International journal of computer vision**, Springer, v. 1, n. 4, p. 321–331, 1988.

KURENKOV, A. et al. DeformNet: Free-form deformation network for 3D shape reconstruction from a single image. **Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018**, IEEE, v. 2018-January, n. Figure 1, p. 858–866, 2018.

LAMATA, P. et al. An accurate, fast and robust method to generate patient-specific cubic Hermite meshes. **Medical Image Analysis**, Elsevier, v. 15, n. 6, p. 801–813, dec 2011. ISSN 13618415.

LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. **Neural Computation**, v. 1, n. 4, p. 541–551, 1989. Available from Internet: <https://doi.org/10.1162/neco.1989.1.4.541>.

LEDERMAN, C. et al. Tetrahedral mesh generation for medical images with multiple regions using active surfaces. In: IEEE. **2010 IEEE International Symposium on Biomedical Imaging: From Nano to Macro**. [S.l.], 2010. p. 436–439.

LENCHIK, L. et al. Automated segmentation of tissues using ct and mri: a systematic review. **Academic radiology**, Elsevier, v. 26, n. 12, p. 1695–1706, 2019.

LI, G. et al. Automatic Liver Segmentation Based on Shape Constraints and Deformable Graph Cut in CT Images. **IEEE Transactions on Image Processing**, v. 24, n. 12, p. 5315–5329, dec 2015. ISSN 1057-7149. Available from Internet: <http://ieeexplore.ieee.org/document/7274362/>.

LITJENS, G. et al. A Survey on Deep Learning in Medical Image Analysis. **Medical Image Analysis**, Elsevier B.V., v. 42, n. December 2012, p. 60–88, feb 2017. ISSN 13618415. Available from Internet: <http://linkinghub.elsevier.com/retrieve/pii/S1361841517301135http://arxiv.org/abs/1702.05747http://dx.doi.org/10.1016/j.media.2017.07.005>.

LITJENS, G. et al. Evaluation of prostate segmentation algorithms for mri: The promise12 challenge. **Medical Image Analysis**, v. 18, n. 2, p. 359 – 373, 2014. ISSN 1361-8415.

LONG, J.; SHELHAMER, E.; DARRELL, T. Fully convolutional networks for semantic segmentation. In: **The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2015.

LORENSEN, W. E.; CLINE, H. E. Marching cubes: A high resolution 3D surface construction algorithm. In: **Proceedings of the 14th annual conference on Computer graphics and interactive techniques - SIGGRAPH '87**. New York, New York, USA: ACM Press, 1987. p. 163–169. ISBN 0897912276. ISSN 00978930. Available from Internet: <http://portal.acm.org/citation.cfm?doid=37401.37422>.

LU, J. et al. An interactive approach to liver segmentation in CT based on deformable model integrated with attractor force. In: **2011 International Conference on Machine Learning and Cybernetics**. [S.l.: s.n.], 2011. v. 4, p. 1660–1665. ISSN 2160-133X.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, v. 5, n. 4, p. 115–133, Dec 1943. ISSN 1522-9602. Available from Internet: <https://doi.org/10.1007/BF02478259>.

MESEJO, P. et al. A survey on image segmentation using metaheuristic-based deformable models: state of the art and critical analysis. **Applied Soft Computing**, v. 44, p. 1–29, 2016. ISSN 1568-4946. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S1568494616301053>.

MILLETARI, F.; NAVAB, N.; AHMADI, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: IEEE. **2016 Fourth International Conference on 3D Vision (3DV)**. [S.l.], 2016. p. 565–571.

MOGHBEL, M. et al. Review of liver segmentation and computer assisted detection/diagnosis methods in computed tomography. **Artificial Intelligence Review**, v. 50, n. 4, p. 497–537, Dec 2018. ISSN 1573-7462. Available from Internet: <https://doi.org/10.1007/s10462-017-9550-x>.

PALOMAR, R. et al. Surface reconstruction for planning and navigation of liver resections. **Computerized Medical Imaging and Graphics**, v. 53, p. 30–42, oct 2016. ISSN 08956111. Available from Internet: <http://linkinghub.elsevier.com/retrieve/pii/S0895611116300684>.

PAN, J. et al. Residual meshnet: Learning to deform meshes for single-view 3d reconstruction. In: IEEE. **2018 International Conference on 3D Vision (3DV)**. [S.l.], 2018. p. 719–727.

PONTES, J. K. et al. Image2mesh: A learning framework for single image 3d reconstruction. In: SPRINGER. **Asian Conference on Computer Vision**. [S.l.], 2018. p. 365–381.

PUMAROLA, A. et al. Geometry-aware network for non-rigid shape prediction from a single view. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2018. p. 4681–4690.

QI, C. R. et al. Pointnet: Deep learning on point sets for 3d classification and segmentation. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2017. p. 652–660.

QI, C. R. et al. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2017. p. 5099–5108.

RIEGLER, G.; ULUSOY, A. O.; GEIGER, A. Octnet: Learning deep 3d representations at high resolutions. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2017. p. 3577–3586.

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: SPRINGER. **International Conference on Medical image computing and computer-assisted intervention**. [S.l.], 2015. p. 234–241.

RUECKERT, D.; GLOCKER, B.; KAINZ, B. Learning clinically useful information from images: Past, present and future. **Medical Image Analysis**, Elsevier B.V., v. 33, p. 13–18, oct 2016. ISSN 13618415. Available from Internet: <http://dx.doi.org/10.1016/j.media.2016.06.009http://linkinghub.elsevier.com/retrieve/pii/S1361841516300871>.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Neurocomputing: Foundations of research. In: ANDERSON, J. A.; ROSENFELD, E. (Ed.). Cambridge, MA, USA: MIT Press, 1988. chp. Learning Representations by Back-propagating Errors, p. 696–699. ISBN 0-262-01097-6. Available from Internet: <http://dl.acm.org/citation.cfm?id=65669.104451>.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. 3rd. ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009. ISBN 0136042597, 9780136042594.

SAXENA, S. et al. An automated system for atlas based multiple organ segmentation of abdominal ct images. **Journal of Advances in Mathematics and Computer Science**, p. 1–14, 2016.

SEDERBERG, T. W.; PARRY, S. R. Free-form deformation of solid geometric models. **Proceedings of the 13th annual conference on Computer graphics and interactive techniques - SIGGRAPH '86**, v. 20, n. 4, p. 151–160, 1986. ISSN 00978930.

SHEN, C. et al. Improving v-nets for multi-class abdominal organ segmentation. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. **Medical Imaging 2019: Image Processing**. [S.l.], 2019. v. 10949, p. 109490B.

SINHA, A. et al. SurfNet: Generating 3D Shape Surfaces Using Deep Residual Networks. In: **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. IEEE, 2017. p. 791–800. ISBN 978-1-5386-0457-1. ISSN 18766102. Available from Internet: <http://arxiv.org/abs/1703.04079http://ieeexplore.ieee.org/document/8099574/>.

SMITH, C. **On Vertex-vertex Systems and Their Use in Geometric and Biological Modelling**. Thesis (PhD), Calgary, Alta., Canada, Canada, 2006. AAINR19574.

WANG, N. et al. Pixel2mesh: Generating 3d mesh models from single rgb images. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2018. p. 52–67.

WANG, P.-S. et al. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. **ACM Transactions on Graphics (TOG)**, ACM, v. 36, n. 4, p. 72, 2017.

WU, J. et al. Marrnet: 3d shape reconstruction via 2.5 d sketches. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2017. p. 540–550.

WU, J. et al. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2016. p. 82–90.

XIA, Y. et al. Realpoint3d: Point cloud generation from a single image with complex background. **arXiv preprint arXiv:1809.02743**, 2018.

XIE, H. et al. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. **arXiv preprint arXiv:1901.11153**, 2019.

YUNIARTI, A.; SUCIATI, N. A review of deep learning techniques for 3d reconstruction of 2d images. In: IEEE. **2019 12th International Conference on Information & Communication Technology and System (ICTS)**. [S.l.], 2019. p. 327–331.