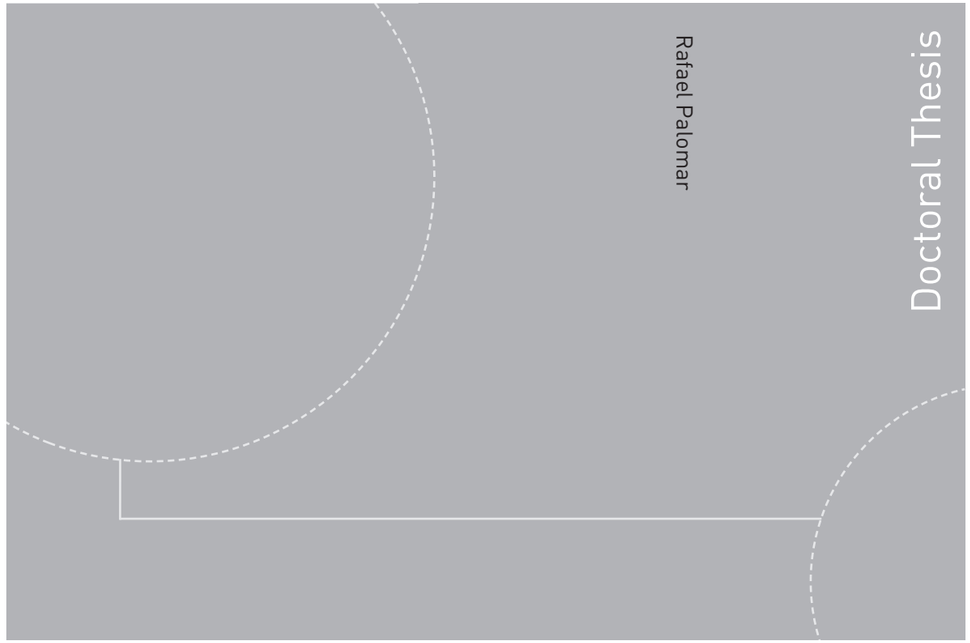
ISBN 978-82-326-3138-4 (printed version)
ISBN 978-82-326-3139-1 (electronic version)
ISSN 1503-8181



Doctoral theses at NTNU, 2018:174

Rafael Palomar

Geometric Modeling for Planning of Liver Resection Procedures

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

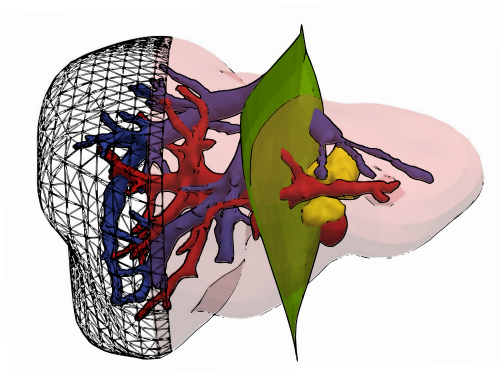
Doctoral theses at NTNU, 2018:174

NTNU

NTNU
Norwegian University of
Science and Technology

Rafael Palomar

Geometric Modeling for Planning of Liver Resection Procedures



Thesis for the degree of Philosophiae Doctor

Gjøvik, June 2018

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Oslo University Hospital
The Intervention Centre

NTNU
Norwegian University of Science and Technology

Oslo University Hospital
The Intervention Centre

NTNU

Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Computer Science

ISBN 978-82-326-3138-4 (printed version)
ISBN 978-82-326-3139-1 (electronic version)
ISSN 1503-8181

Doctoral theses at NTNU, 2018:174

Printed by Skipnes Kommunikasjon as

To my beloved wife Maria Emblem Palomar.
Thank you for lighting the path.

Acknowledgements

I would like to thank my supervisors Ole Jakob Elle, Faouzi A. Cheikh, Bjørn Edwin and Azeddine Beghdadi for all the support, motivation and autonomy provided in this period. Without their engagement, guidance and determination this work would not have been possible. I would also like to thank my co-authors and collaborators Juan Gómez Luna, Joaquín Olivares Bueno, Åsmund A. Fretland and Per K. Hol who greatly contributed to this work with insight, discussions, expert knowledge and thorough reviews of my work. In addition, I would like to thank the Research Council of Norway and Helse Sør-Øst for funding this research.

I have the fortune to count with a large number of colleagues and friends who also left their footprint on this work. I am grateful to Rahul Kumar, Hugues Fontenelle, Kim Mathiassen, Magnus Krogh and Louise Oram for creating a friendly work environment around me, their support reached beyond the boundaries of science. This gratitude should be extended to Cristóbal J. Martín Moral, José Rodríguez Pérez and José M. Palomares Muñoz who once set the seed of science and computing very deep inside me.

Very specially, I would like to thank my beloved wife Maria E. Palomar and son Leandro E. Palomar for their limitless love, support and patience during this period.

Last, but certainly not least, I would like to thank my parents Rafael Palomar Sánchez and María C. Ávalos Carrillo, as well as my sister Carmen Palomar Ávalos for their continuous support in this journey taken few thousand kilometers away.

Contents

List of Tables

List of Figures

Part I

Introduction

CHAPTER 1

INTRODUCTION

LIVER cancer[1] is one of the most common causes of cancer worldwide and its frequency is increasing in geographical areas of traditionally low incidence [1]. In Norway particularly, a remarkable increase in incidence rates have been observed in the last years according to the official statistics [2]. Liver resection, which refers to the surgical removal of a tumor in the liver, is the only curative treatment for large hepatocellular carcinoma[2] (**HCC**) [3], and potentially curative for colorectal metastases [4].

For more than a decade, computer-assisted systems have been helping surgeons in the decision-making process supporting therapy planning and surgery guidance. In the case of liver resections, these systems have made their way into the clinic providing surgeons with tools to plan liver resections [5, 6] and guide surgical interventions [7]. Computer-assisted systems for planning and navigation rely on geometric modeling techniques. These techniques are not only used for computing 3D virtual models representing the patient's anatomy, but also provide the necessary means for the specification of resections, this is, surgical paths dividing

[1] We use the term *liver cancer* to refer to both primary liver cancer (originated in the liver) and secondary liver cancer (originated in other organs and spread to the liver) such as colorectal metastases.

[2] Type of primary liver cancer.

the liver in resected and remnant volumes.

Advances in imaging and computing are changing the way 3D virtual models are computed and utilized in the clinical reality. Improvements in imaging allow not only the more accurate representation of the patient's anatomy, but also the addition of non-anatomical (functional) information; in parallel, new trends in computing allow performing more complex operations (e.g. real-time deformations, mapping of functional information into anatomical models). All these improvements are irreversibly shifting the current current *state-of-the-art*—where pre-operative static 3D models are adapted (i.e. rigid and non-rigid registration techniques)—towards models which can be continuously deformed in real-time to provide a more accurate representation of the target organ [8]. In order to maximize the number of operations performed to these models, optimization of the geometric models (i.e. reducing the number of elements while keeping the accuracy of representation), as well as adaptation of methods and algorithms to the new computing trends, are needed.

Along with the technological developments achieved in the last decade, the medical community has continued improving and developing new surgical techniques. In many cases, these new techniques not only inherited the benefits of the technologies employed by their predecessors, but also established a demand for further technological innovations. In the context of geometric modeling, this demand translates into the development of new systems able to enhance the ability to define and evaluate resection plans, as well as to provide surgeons with more complex models which can include functional information.

To be sure, the adoption of new computing paradigms combined with new geometric modeling tools is a necessary step to advance computer-assisted surgical systems towards a new reality, in which models will not only increase their complexity but also become deformable. Careful consideration of deformations is important for navigation applications where the soft tissue organs are continuously deforming due to motion, patient positioning (e.g., gravity) and physical interaction during surgery.

Aim of This Thesis

The work presented in this thesis was conceived with two major objectives in mind. The first objective was to investigate geometric modeling techniques able to generate 3D models of anatomical structures (particularly the liver surface). The application of these techniques should lead to an improvement over *state-of-the-art* geometric modeling techniques[3] in terms of smoothness and complexity of the

[3] *Marching Cubes* [9] is currently considered the *state-of-the-art* for modeling of liver surfaces.

models, without compromising the visual quality. These properties will contribute to the real-time manipulation of these models (e.g. performing computational operations such as contour slicing and computation of distance maps).

The second objective of this work was to investigate new techniques to model resections (regardless of their type) in an effective and efficient way. This research should lead to computerized tools that are usable and able to be integrated in clinical work-flows. Flexibility of representation for different types of resections is of paramount importance, since it can support not only classic surgical resection techniques, but also emerging surgical techniques such as *parenchymal-sparing* [10] and *associating liver partition and portal vein ligation for staged hepatectomy* [11]. The underlying mathematical and computational methods should be subject to adaptation to new computing paradigms, for instance, heterogeneous computing and general purpose graphics processing units (GPGPU) for which its parallelization and optimization is essential.

1.1 Organization of the Thesis

This research work is structured in three articles supporting the objectives established in the previous section:

Paper I: Surface Modeling for Planning and Navigation of Liver Resection Procedures (Chapter 5). This work studies the application of *Poisson surface reconstruction* in the context of automatic and efficient modeling of liver surfaces with applications to planning and navigation of liver resection procedures.

Paper II: A Novel Method for Planning Liver Resections using Bézier Surfaces and Distance Maps (Chapter 6). This work introduces a new method for specification of resection surfaces. The novelty relies on (1) the specification of resection using Bézier surfaces, which can be deformed in real-time by a set of control points, (2) distance maps which provide a means for the visualization of safety margins and (3) new interaction techniques which allow the control points to be moved in groups to reduce the number of interactions. Associated techniques for volumetry computation (resected and remnant) using these surfaces are also described.

Paper III: High-Performance Computation of Bézier Surfaces on Heterogeneous Platforms (Chapter 7). This work performs an extensive analysis on the parallelization and optimization of Bézier surfaces (the core mathematical construction of our resection planning algorithm). This work is presented in an implementation-independent form, so other research areas can benefit from it.

In addition to the three aforementioned works, we explored the translation of some of the techniques investigated, to other organs and clinical applications. Particularly, the use of Poisson surface reconstruction, is employed in modeling of the left atrium for atrial fibrillation procedures.

Paper IV: Intra-Operative Modeling of the Left Atrium: A Simulation Approach Using Poisson Surface Reconstruction (Appendix A). In this work, we analyze the application of Poisson surface reconstruction in the context of intra-operative modeling the left atrium used in the electroanatomic mapping process supporting radio-frequency catheter ablation.

CHAPTER 2

BACKGROUND

THE aim of this chapter is to provide the background needed to understand the motivation and the research work presented in this thesis. First, a medical background on liver anatomy and treatment of liver cancer through surgical resection is presented. Understanding the human liver and how contemporary liver resections are performed provides a better understanding of the design decisions and strategies employed in this work to solve the objectives established in this project (see Section 1). Secondly, we describe the data work-flow for the use of computer-assisted systems for planning and navigation in the clinic. Modeling of the liver surface (parenchyma), as well as planning algorithms will be emphasized since they constitute the core of this research. Then, the focus is moved towards more technical areas which are key to this project: Bézier surfaces, which are a fundamental tool employed for surgery planning in this project, and high-performance computing. Although this thesis work pivots around liver surgery, in Paper IV, some of the approaches presented are extended to the electroanatomic mapping (**EAM**) of the left atrium for the treatment of atrial fibrillation (**AF**); the background needed to understand the motivation and results related to EAM is presented in Section 2.7.

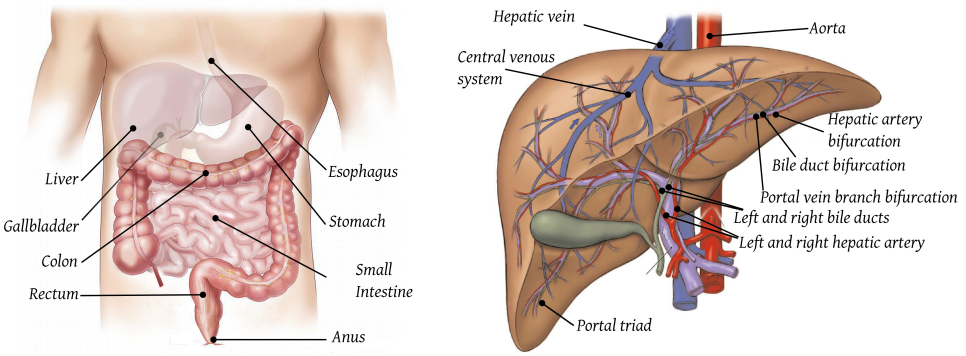


Figure 2.1: Anatomy of the human liver. (left) Position of the liver and its surroundings in the abdomen. (right) Detailed anatomy of the liver. Adapted from the originals http://maxpixel.freegreatpicture.com/Marking-Liver-Intestine-Medical-Offal-1463369 and https://commons.wikimedia.org/wiki/File:Vidin%C4%97_kepen%C5%B3_sandara.png

.

2.1 The Human Liver

The liver (Figure 2.1) is the largest internal organ. Located in the upper-right quadrant of the abdomen, the liver accounts for about 2% to 3% of the average body weight. The metabolic properties of the liver makes it a vital organ. The liver not only serves as a storage for glycogen, fat soluble vitamins and minerals, but also performs over 500 metabolic functions. In addition the liver detoxifies and purifies blood.

Most of the *in-flow* blood to the liver originates from the gastro-intestinal tract, the spleen, the pancreas and the gallbladder, and enters the liver with low pressure (and low levels of O_2) through the portal vein. High-pressure blood is also supplied by the hepatic artery, which branches directly from the descending aorta (with high levels of O_2). Blood from these two vessels (hepatic artery and portal vein) joins in the capillary bed of the liver and leaves the organ through the hepatic vein.

The anatomical division of the liver in segments described by Couinaud [12] (Figure 2.2) is considered the reference and the support for many of the advances in liver surgery in the last 50 years. However, this division of the liver does not pose an absolute consensus in the medical community. Abel-Misih and Bloomston [13] highlight the common misconception that the liver is anatomically divided in two lobes (left and right) separated by the falciform ligament. In the view of the authors, this may be true under a morphological standpoint, however, this is not true under a functional standpoint. In [14], Bismuth sentences that the different

anatomical divisions of the liver and hepatectomies are "causing confusion" among surgeons. In that work, Bismuth, reviews and corrects the original terminology proposed by Couinaud, considering the multiple descriptions made during the decades following the work of Couinaud.

More recently, Majno *et al.* [15] establishes a three-level framework of the anatomical description of the liver according to the purpose of the anatomical description:

Conventional: which corresponds to the classic eight-segment division proposed by Couinaud.

Surgical: to be applied in surgical liver resections and transplantations, and considers the real branching of the major portal pedicles and hepatic veins. As the author highlights, this level requires accepting that the Couinaud model is a simplification model.

Academic: which is the most complex model and considers that the eight-segment models derived from the second-order portal branches are insufficient, provided that some studies point to an average of 20 second-order bifurcations. The authors propose a division based on the "1-2-20" concept and suggest that it fits best the number of anatomical segments.

For simplicity and clarity—as often happens in the literature when referring to liver surgery—we employ the Couinaud model as a base for this chapter, as well as for the description and discussion of our results.

2.2 Liver Cancer and Surgical Resection

Liver cancer can be either primary (originated in liver tissue) or secondary (extended to the liver from tumoral cancer cells located in other organs). Liver cancer is considered one of the leading causes of cancer deaths worldwide [1]. For HCC (primary liver cancer), which accounts for 70% to 80% of the liver cancer cases worldwide [16], surgical resection is the treatment of choice and it is considered as a safe and effective therapy [17]. Selected patients with metastatic liver tumors (secondary)—which develops in 50% of the cases of colorectal cancer—present up to 58% increased 5-year survival rates after liver resection [18].

Liver resection, also referred to as hepatic resection or hepatectomy, can be performed under either open surgery or laparoscopy (minimally invasive surgery) [19]. In open surgery, which represents the traditional approach, a large abdominal incision is performed; the operation takes place under the direct view and direct contact of the surgeon with the organ. In laparoscopic surgery, however, the patient is operated through elongated instruments, which are introduced

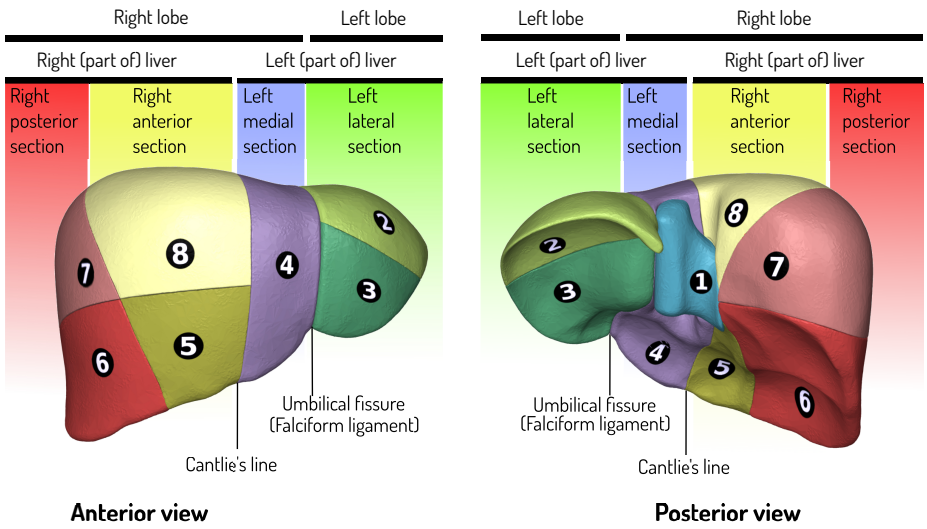


Figure 2.2: Anatomical division of the liver according to Couinaud. Adapted from the original by the Database Center for Life Science (DBCLS) (http://dbcls.rois.ac.jp/). The original image is available at https://commons.wikimedia.org/wiki/File%3ALiver_04_Couinaud_classification.svg

into the body through small incisions—entailing a great challenge for the operator due to the reduced visibility (through an endoscopic camera), and a reduced maneuverability of the instruments.

Laparoscopic surgery, which has been applied successfully to a wide variety of cases such as HCC [20] and colorectal metastases [21], is associated with a reduction on post-operative pain, shorter hospital stays and faster recovery periods. Together with these benefits, laparoscopic liver resection is also associated to technical challenges related to the exploration and mobilization of the liver, the vascular control and parenchymal transection (particularly in cirrotic livers) [22]. Despite the fact that laparoscopic liver surgery has being established for more than a decade now, its comparison to the open approach was (and still is) a subject of research for a large cohort of studies [21, 23, 24].

Regardless of the open/laparoscopic nature of the procedure there exist different types of resections. Classification and typology of resections (Figure 2.4) are still based on the Couinaud eight-segments model. Broadly speaking, could distinguish two groups of resections: *anatomic resections*, where one or more complete segments are removed respecting the boundaries established by the Couinaud division; on the contrary, *non-anatomic resections* do not respect boundaries established by the Couinaud model. According to Strasberg and Phillips [25], anatomic

Figure 2.3: Open and laparoscopic surgery approaches.

resections can be further classified according to a three-orders: hemilivers (order 1), sections (order 2) and segments (order 3); the authors also consider extended resections which can be composed by a combination of a hemiliver and a section.

2.3 Overview on Planning and Navigation of Liver Resection Procedures

For nearly two decades, computer-assisted systems for surgery planning and navigation have been helping physicians in the decision-making processes involved in surgical interventions [26, 27]. Surgery planning refers to the specification of a surgery plan prior to the operation (e.g., a trajectory to be followed during surgery) and often includes indicators of risk and surgical outcome (e.g., distance to anatomical structures and volumetry of resected tissue). Surgical navigation, on other hand, is concerned with the process of monitoring and controlling the movement of surgical instruments to ensure these are positioned correctly and also follow a trajectory required to perform a surgical action (e.g. a resection).

Planning and navigation of liver resection procedures have proven to be beneficial for not only the localization of tumors and the precision of surgery planning [28, 29, 30], but also for the improvement of orientation and confidence of surgeons during the operation [31]. Surgery planning and navigation are often combined and both rely on the generation of three-dimensional (**3D**) models obtained from pre-operative imaging.

As shown in Fig. 2.5, a typical work-flow involving planing and navigation starts from the acquisition of pre-operative (prior to operation) images obtained from either computed tomography (**CT**) or magnetic resonance imaging (**MRI**)—to date, CT is the most widely used imaging modality in clinical liver resection work-flows involving planning and navigation—. Regardless of the imaging modality, a

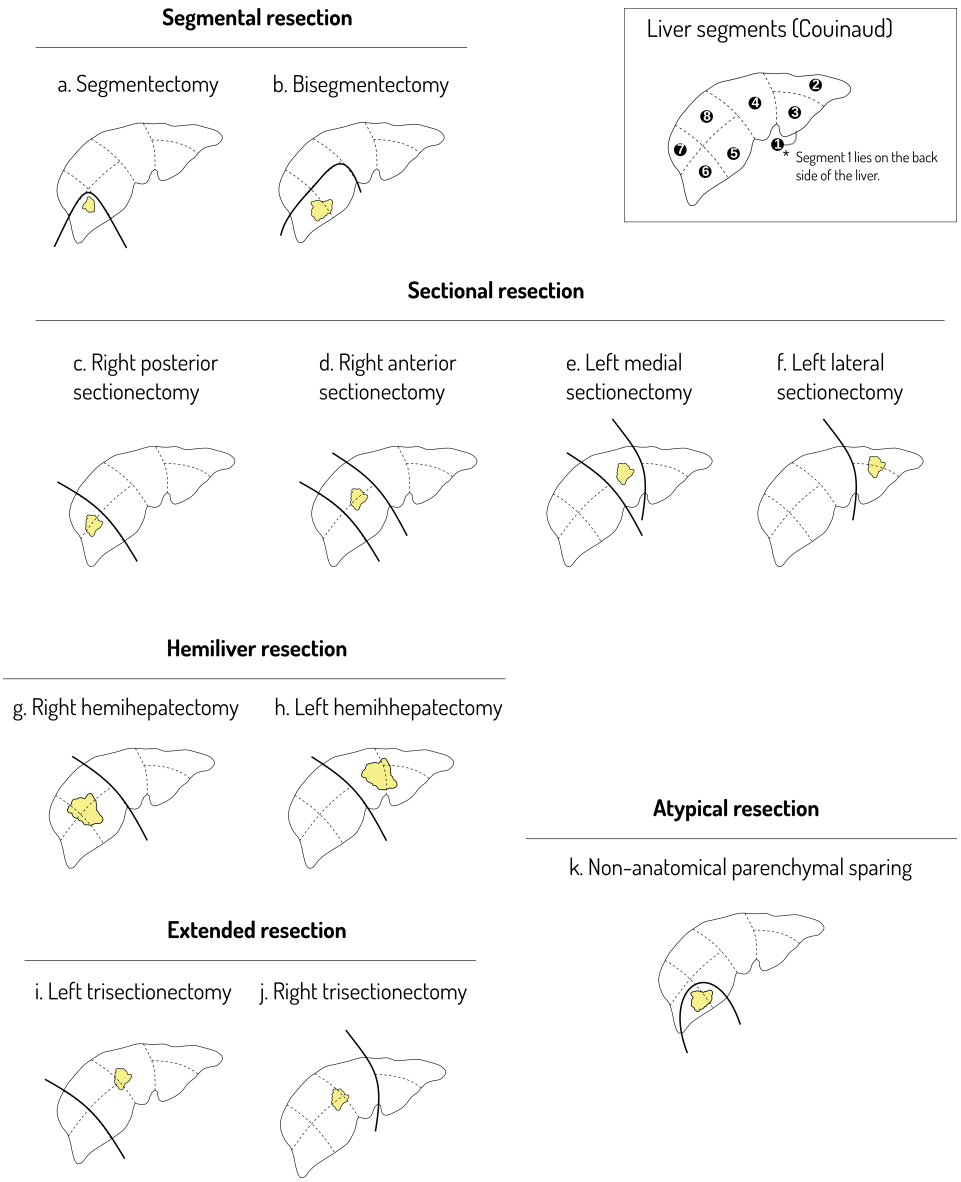


Figure 2.4: Liver resection types: (a-i) anatomical division of the liver in segments according to the three-orders organization of Strasberg and Phillips [25]; (j) non-anatomical resection.

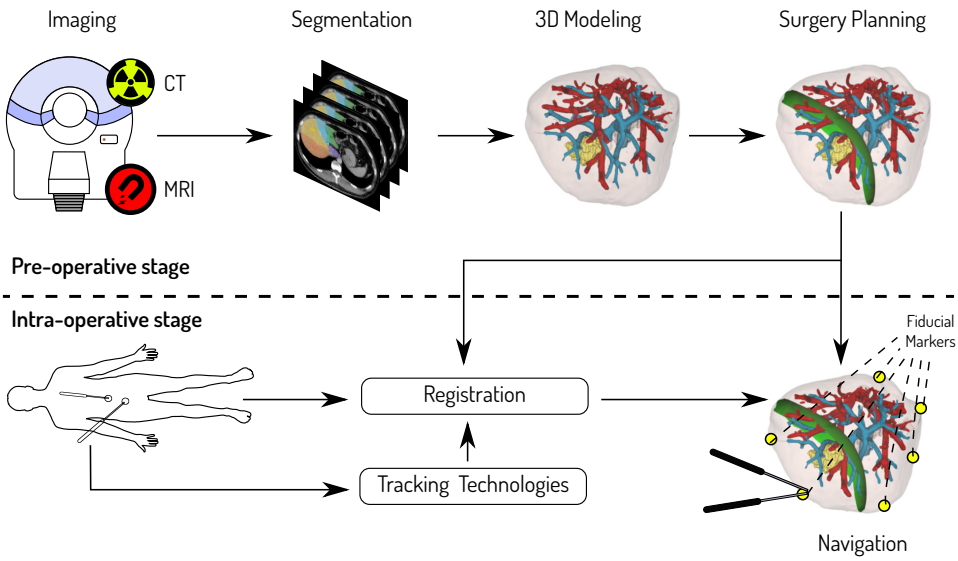


Figure 2.5: Work-flow for pre-operative planning of liver surgery and intra-operative navigation.

medical image (volumetric) is represented by a scalar field defined as $F : \mathbb{R}^3 \rightarrow \mathbb{R}$ where the point $\mathbf{p}_i = (x_i, y_i, z_i)$ with $i = 1, 2, ..., N$ is associated to a scalar value $F(\mathbf{p}_i) = v$ representing an intensity level.

The process of defining (separate) the different anatomical structures in the image space is know as segmentation and produces a new scalar field $S : \mathbb{R}^3 \rightarrow \{l_1, ..., l_k\}$ known as *label map*, where the point $\mathbf{p}_i = (x_i, y_i, z_i)$ with $i = 1, 2, ..., N$ is associated to a value (label) from k different types of tissue. In the case of liver resection procedures, the label map is usually constrained to the set $L = \{l_b, l_p, l_h, l_o, l_t\}$ where l_b represents a value indicating tissue not relevant for the procedure (image background), and the rest represents values for *parenchyma*, *hepatic venous system*, *portal venous system* and *tumor* tissue respectively. In some cases, this set of labels can be extended to consider multiple tumors[1] or other anatomical structures such as the bile duct. Due to the difficulties for its automation, segmentation is considered to be the major bottleneck in a work-flow including planning and navigation of liver resection procedures [32]. Medical image segmentation continues to be subject to extensive research in the the fields of computer vision and bio-medicine and it is considered as a challenging task which to date, requires some degree of human interaction.

[1]The occurrence of multiple different tumors can be indicated by extending the set L to include different tumor values and not just one (l_t).

The separation of image voxels into different categories or labels inherently defines the geometry of the underlying anatomical structures. However, visualization of this geometry in computer graphics, is established through surface models structured as triangular meshes. The process of transforming voxel-based geometries into 3D models is know as 3D modeling, which will be covered in more detail in Section 2.4 since it is one of the core topics of this thesis.

Geometrically speaking, 3D models together with segmentation models contain all the information needed to perform surgery planning, this is, the definition of *virtual resections* [33]. Visual resections help clinicians to visualize the cutting path separating healthy tissue from resected tissue as well as other anatomical structures affected by the resection such as vessels. The geometric information hold by the 3D models and the virtual resection enables computer-assisted systems to calculate volumetric data which can be used as indicator of adequacy of surgery plan. Planning of liver resections is detailed in Section 2.5 since it is one of the relevant topics for this work.

All the processes mentioned above (imaging, segmentation, 3D modeling and surgery planning) happen before the operation takes place, this is, they are pre-operative processes (see Fig. 2.5). The models and images derived from these processes have an intrinsic value and could be employed to guide the surgery without navigation technologies—surgeons could manipulate these data items and use them as a reference during surgery. However, this approach requires that the surgeons align—at least, mentally—the information presented in the screen with the surgical reality using features and unequivocal landmarks and features surrounding the surgical site.

The approach described above, can be further improved by means of navigation. Navigation combines pre-operative 3D models of anatomical structures with 3D models representing the operating tools which moves in real-time while the surgeons perform the surgical actions. This helps the surgeons visualize the surgical reality in the 3D virtual space, with the possibility of presenting additional information such as distances and risk assessments.

As described in Terry and Peters [34] image-guided surgery (navigation) utilizes a localizer to track the surgical instruments; to date, optical and electromagnetic tracking systems are the two main technologies employed in the clinical routine, where optical tracking is the most used. To mitigate the geometric discrepancies between the pre-operative reality (medical images, segmentations and 3D models) and the intra-operative reality (the patient on the operating table) produced by differences in coordinate systems and deformations, registration (the process of aligning two coordinate systems) is performed. Registration can be broadly

classified in rigid and non-rigid, for a detailed review of registration methods, we refer to the extensive work in Markelj *et al.* [35].

To date, there is no single computer-assisted system covering all the stages in the work-flow described above (Fig. 2.5). However, there exist software solutions covering different individual stages which can be combined. Zygomalas et al. [32] highlight the following tools for liver segmentation: *Myrian® XP-Liver* (Intrasense, Montpellier, France), *MeVisLab* (MeVis Medical Solutions AG, Bremen, Germany), *Mint Liver™* (Mint Medical GmbH, Heidelberg, Germany), *Synapse 3D* (FUJIFILM Medical Systems Inc., Stamford, Connecticut, USA), *Scout™* (Pathfinder technologies, Nashvill, USA) and *IQQA®-Liver* (EDDA Technology, Inc., NJ, USA), as well as the open-source solutions ITK-SNAP and 3D Slicer [36]. Some of these computer systems are also able to perform 3D modeling and surgery planning; software platforms like *CAS-ONE Liver* (CAScination, Bern, Switzerland) or the open-source NorMIT[2].

2.4 3D Modeling of Anatomical Structures

The aim of 3D modeling is to derive surface models (∂M_i with $i = 1, ..., k$) from the label-map (segmented image) S containing k different tissues encoded as labels. In computer graphics, the most common representation form of surface models is triangular meshes, which can be denoted as sets $\mathcal{M} = \{V, T\}$ with the set of vertices $V = \{0, 1, 2, ...\}$. Vertices have associated positions in space $\mathbf{p}_i \in \mathbb{R}^3$, edges $E = \{(i, j)|i, j \in V\}$ depicting connectivity, and triangles $T = \{(i, j, k)|(i, j), (j, k), (k, i) \in E\}$. Due to the nature of the underlying data (medical images taken from the patient) the models resemble the anatomical structures of the patient, this is, the models are patient-specific.

A complete 3D patient-specific model (Fig. 2.6) to be used for planning and navigation of liver resection procedures consist of the following geometric items:

- the parenchyma, or the functional tissue of the liver which in 3D is represented as a shell (surface model) containing the rest of the anatomical structures;
- the vessels, which typically are separated into portal vessel system and hepatic vessel system according to a in-flow/out-flow criteria;
- the tumor(s), which might intersect some of the vessels.

[2] http://normit.no

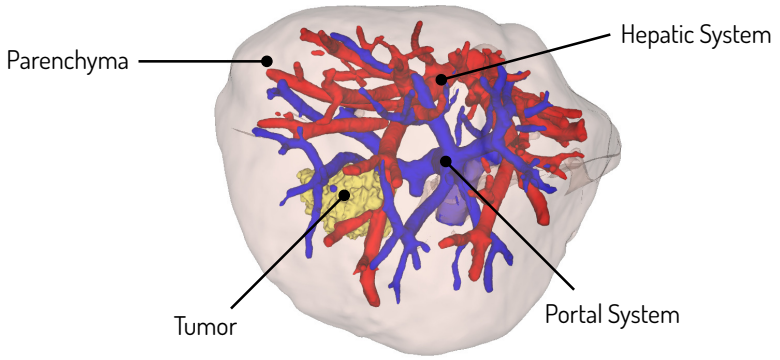


Figure 2.6: Complete 3D patient-specific model obtained from segmented CT. It includes parenchyma, portal and hepatic vessels systems and tumor.

Under a surgical point of view, liver arteries are considered of less importance than veins since the latter carry most of the blood supply, therefore arterial models are not constructed. Modeling of other anatomical structures such as the bile ducts can be beneficial in some cases, but these structures generally present low contrast in CT, and therefore are difficult to segment and reconstruct into a 3D model; models originated from segmented MRI are more promising for reconstruction of these anatomical structures.

Due to their geometric structure—tubular structures with high curvature and convex features—vessels are the most complex geometric anatomical structures involved in the reconstruction of 3D patient-specific models and therefore the topic has been subject of extensive research. Preim *et al.* [33] which provides a comprehensive overview of available reconstruction methods, separates the reconstruction strategies into two categories:

- *model-based* methods which work on the basis of assumptions (e.g, vessels present circular cross-sections) and either generate parametric or implicit surfaces or fit primitives such as cylinders or truncated cones;
- *model-free* methods which are not based in any assumption. Marching cubes [9], multi-level partition of unity (MPU) implicits [37] and subdivision surfaces [38] are evaluated and compared in [39].

Model-based provides smooth models which are easy to interpret but might produce inadequate results for the analysis and representation of certain pathologies (e.g. aneurysms). On the other hand, model-based methods adheres to the under-

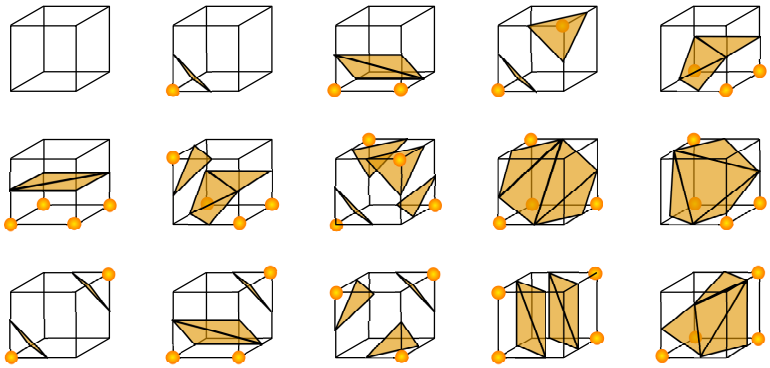


Figure 2.7: The 15 possible cube configurations in Marching Cubes according to the feature points present in the neighborhood of the voxel of interest. Extracted from https://commons.wikimedia.org/wiki/File:MarchingCubes.svg

lying data, and therefore, in some cases they might lead to lower mesh quality resulting in poor visualization (e.g, original image with low resolution or noise).

While reconstruction of vascular structures have been subject to extensive research, reconstruction of parenchyma and tumors have been given very little attention in the literature. For the reconstruction of these anatomical structures, marching cubes emerges as the *de-facto* standard.

In the following section more details about two reconstruction methods which are key for understanding some of the works in this thesis.

2.4.1 Marching Cubes

Marching cubes was originally proposed by Lorensen and Cline [40]. Its simplicity, automaticity, parallelization possibilities [41], and the strict adherence to the underlying data, makes marching cubes the most used method to extract isosurfaces from segmentation images regardless of the type of anatomical structure.

The main idea of marching cubes is the establishment of a correspondence between the neighborhood around a voxel and a specific pattern of triangles in 3D space. Given a segmentation image S containing k different classes of tissue encoded as labels with values L=$\{l_1, ..., l_k\}$, and a particular $l_s \in L$ corresponding to the structure of interest (e.g., parenchyma, vessels or tumor), the neighborhood $(3 \times 3 \times 3)$ of a given voxel can be considered to be either the tissue of interest or a different tissue; therefore there exist $2^8 = 256$ possible neighborhoods. Marching cubes creates a pre-defined pattern of 3D triangles for every one of the 256 possible neighborhoods. Exploiting the symmetry of patterns, this number of possibilities can be reduced to 15 (see Fig. 2.7).

Marching cubes has been subject to a number of adaptations and improvement over the last decades. Newman and Yi [9] provides a comprehensive survey of the extensions to the original proposal by Lorensen and Cline. These extensions are related to the traversal through the data, algorithm's isosurface assembly component, reduction of number of triangles and solving triangulation ambiguities.

In the medical domain, marching cubes is often followed by decimation [42] (reduction of triangles) and smoothing [43] processes. These processes not only introduce additional operations—which translates into either longer processing times or the need for more computing power—, but also additional parameters in the process. The introduction of additional parameters is a great disadvantage in the medical domain, where unsupervised (automatic) algorithms are of paramount importance for the feasibility of methods to be integrated into clinical work-flows. One solution to this problem is the choosing of a set of parameters—which we could call conservative— which guarantee reconstructions with reduced risk of reconstruction degenerations due to smoothing and decimation, but not exploiting the possibilities of smoothing and decimation stages to the maximum.

2.4.2 Poisson Surface Reconstruction

Poisson surface reconstruction (**PSR**) was proposed in Kazhdan *et al.* [44] as a method for reconstruction of watertight surfaces models ∂M (i.e., absence of holes in the mesh) from oriented clouds of points. PSR combines the advantages of global and local fitting schemes and it is know for its noise resiliency.

The idea behind PSR is the reconstruction of an indicator function (Fig. 2.8) $\mathcal{X} : \mathbb{R}^3 \rightarrow \mathbb{R} \in [0, 1]$ which gradient $\nabla\mathcal{X}$ resembles the structure of the oriented cloud of points $\vec{V}$ to be reconstructed. Then the problem can be thought of in terms of finding the scalar function $\mathcal{X}$ whose gradient best matches the cloud of points $\vec{V}$:

$$\tilde{\mathcal{X}} = \arg\min_{\mathcal{X}} \|\nabla\mathcal{X} - \vec{V}\| \tag{2.1}$$

with $||.||$ being the Euclidean norm. By making use of the divergence operator, Kazhdan *et al.*[44] transforms this problem into a variational problem optimized by solving the Poisson equation:

$$\Delta\mathcal{X} \equiv \nabla \cdot \nabla\mathcal{X} = \nabla \cdot \vec{V} \tag{2.2}$$

The solution is computed on the basis of an adaptive and multi-resolution structure, more precisely, PSR makes use of the minimal octree $\mathcal{O}$ to which every points sample of the cloud of points falls into a leaf node at a depth d. In an intuitive manner, the depth parameter d can be thought of as a way to control the granularity

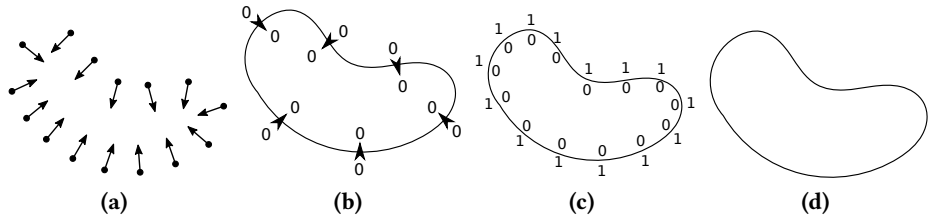


Figure 2.8: Overview of Poisson surface reconstruction in 2D contour of liver parenchyma: (a) oriented set of points $\vec{V}$; (b) gradient of the indicator function $\nabla\mathcal{X}$; (c) indicator function $\mathcal{X}$; (d) surface ∂P.

of the mesh—higher depth values lead to complex models able to represent smaller features and *vice versa*). Finally, to obtain the model ∂M, isosurface extraction is applied to the average value of the approximated indicator function $\tilde{\mathcal{X}}$

2.5 Planning Liver Resection Procedures

The aim of planning liver resections is obtaining the separation between resected tissue and remnant tissue in terms of a virtual resection. Virtual resections can be considered as specifications of cutting trajectories which surgeons will follow during the surgical procedure. Computer-assisted systems for planning liver resections have provided mechanisms for the definition of virtual resections for over a decade. Despite the maturity of these systems, different methods for the specification of virtual resections coexist—which shows a lack of general consensus around strategies for planning liver resections.

Preim and Botha [33] describe the two most used techniques (*drawing on slices* and *using deformable cutting planes*) which have been successfully implemented in computer-assisted systems employed in the clinical routine.

2.5.1 Drawing on slices

The drawing-on-slices paradigm (Fig. 2.9) consist of drawing 2D traces in individual slices (2D) S_i belonging to a medical volume (3D). These traces are then employed to compute the 3D interpolated surface meeting the specified traces T_i. Even though this is considered as a tedious method which, according to Preim and Botha [33], drawing in 50-100 slices is often required.

Ruskó *et al.* [6] proposes a method in which specification of resections can be performed by only drawing in 3 slices. In this work, the authors employ quadratic B-Spline interpolation which requires

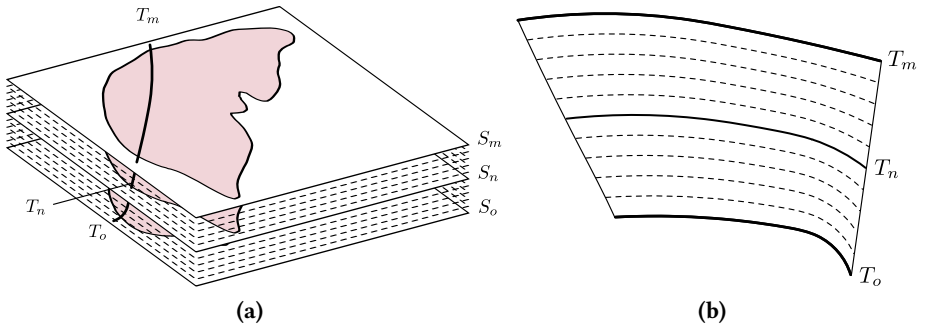


Figure 2.9: Planning liver resection procedures by drawing in slices. (a) Individual traces T_m, T_n and T_o at slices S_m, S_n, S_o respectively; (b) Resection surface obtained by interpolation of traces T_m, T_n and T_o.

- definition of at least 3 traces with 3 points per trace,
- absence of self-intersecting traces,
- traces should be defined in the same type of plane (axial, coronal, sagittal).

2.5.2 Deformable cutting planes

As opposed to drawing-on-slices approaches, in which interactions occur in 2D images, interactions in a deformable-cutting-plane approach take place in virtual 3D environments (e.g. using 3D models) [5].

The method consists of two steps. First, an initial approximation in terms of a plane is computed by defining a set of connected traces on the surface of a 3D liver—these traces often resemble the shape of a ring around the parenchyma. Using these traces, a plane slicing the parenchyma is approximated by means of principal component analysis (**PCA**); this plane will be automatically adjusted (locally) to fit the original traces by translating points in the plane along the normal of the initial plane P_0; Laplacian smoothing of the mesh is then applied . In a second step, the user can refine the resection by applying deformations to the surface. These deformations are performed in terms of a cosine function which applies within a range of influence defined by the user—larger ranges are then translated into more global deformations and *vice-versa*.

Konrad-Verse *et al.* [5] indicate some possible visualization improvements like the continuous visualization of the resection margin and the possibility to set the resection surface transparent, as well as some interaction improvements like the implementation of an “undo” functionality to reverse the deformations performed.

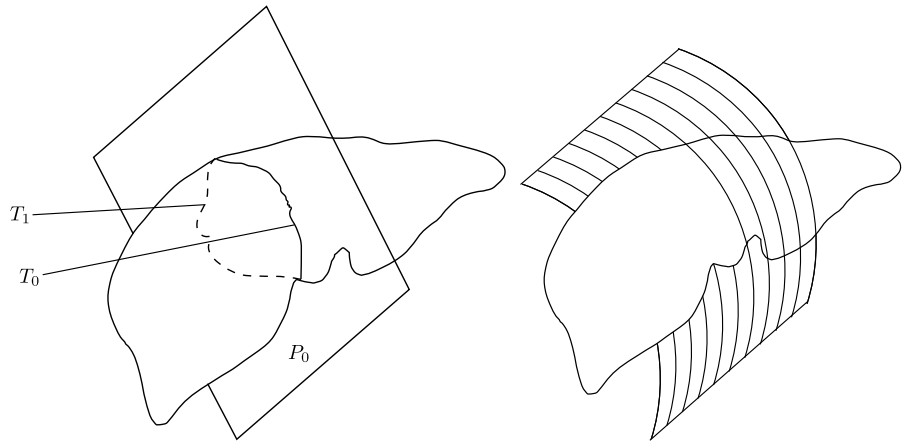


Figure 2.10: Planning liver resection procedures by using a deformable cutting plane. (a) Approximation of the initial plane P_0 through the definition of traces T_0 and T_1. (b) Resection surface obtained by deformation of the initial plane.

2.6 Computation of Bézier Surfaces

Bézier surfaces (also known as Bézier tensor-product surfaces) (Fig. 2.11) are geometric constructions widely employed in the fields of computer graphics and engineering. Mathematically, a non-rational Bézier tensor-product surface $\mathbf{S} : \mathbb{R}^2 \to \mathbb{R}^3$ of degree (m, n) is defined as

$$\mathbf{S}(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{C}_{i,j} B_{i,m}(u) B_{j,n}(v), \tag{2.3}$$

with $u, v \in [0, 1]$. $\mathbf{C}_{i,j}$ are 3D control points and $B_{i,m}$, $B_{j,n}$ are Bernstein polynomials generically defined as

$$B_{i,m}(u) = \binom{m}{i} (1 - u)^{m-i} u^i, \tag{2.4}$$

with $0 \leq i \leq m$.

Bézier surfaces hold multiple properties which make them very useful geometric constructions. For a deeper understanding on Bézier surfaces we refer to [45] which presents an comprehensive description of Bézier surfaces and their underlying properties. In the following lemma, we list the most interesting properties of Bézier surfaces in the context of this work.

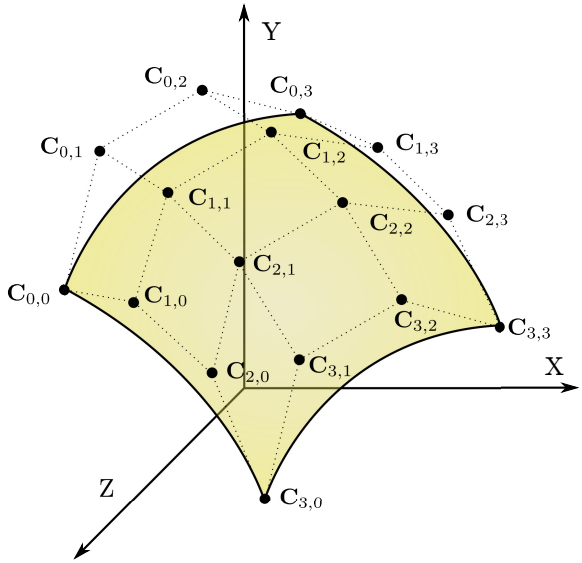


Figure 2.11: Bézier bi-cubic tensor-product surface and its set of control points forming the control polygon.

Lemma 1 *Let* $\mathbf{S}$ *be a parametric bi-linear Bézier surface of degree* (m, n) *as described in Eq. (2.3). Such surface has the following properties:*

(a) Surface contained in the convex hull $\mathbb{CH}$*:*

$$\mathbf{S}(u, v) \in \mathbb{CH}(\mathbf{C}_{0,0}, ..., \mathbf{C}_{m,n}) \; \forall (u, v). \tag{2.5}$$

(b) Affine transformation invariance:

$$\mathbf{T}(\mathbf{S}) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{T}(\mathbf{C}_{i,j}) B_{i,m}(u) B_{j,n}(v), \tag{2.6}$$

where T *is an affine transformation (i.e., rotation, reflection, translation or scaling).*

(c) Polyhedral approximation: under triangulation, the net of control points forms a planar polyhedral approximation of the surface.

Alternative formulations of Bézier surfaces can be derived in order to obtain forms which are more favorable for its parallelization. In this line, Bézier surfaces can be expressed in terms of a product of matrices:

$$\mathbf{S}(u,v) = \mathbf{U}(u)\mathbf{R}(m)\mathbf{C}\mathbf{R}(n)^T\mathbf{V}(v)^T, \tag{2.7}$$

where the $\mathbf{C}$ represents the net of control points

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{0,0} & \mathbf{C}_{0,1} & \dots & \mathbf{C}_{0,n} \\ \mathbf{C}_{1,0} & \mathbf{C}_{1,1} & \dots & \mathbf{C}_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_{m,0} & \mathbf{C}_{m,1} & \dots & \mathbf{C}_{m,n} \end{bmatrix}.$$

.

The vectors $\mathbf{U}$ and $\mathbf{V}^T$ are polynomial spaces of degree m and n, associated to the parameterization directions u and v respectively. Generically, these basis vectors take the form $\mathbf{T}(t) = [t^\alpha, t^{\alpha-1}, ..., t^0]$, where α is the degree of the polynomial space. The matrix of coefficients $\mathbf{R}$ is then defined as:

$$\mathbf{R}(t) = \begin{bmatrix} \binom{t}{0}\binom{t}{t}(-1)^t & \binom{t}{1}\binom{t-1}{t-1}(-1)^{t-1} & \dots & \binom{t}{t}\binom{t-1}{t-1}(-1)^0 \\ \vdots & \vdots & \ddots & \vdots \\ \binom{t}{0}\binom{t}{1}(-1)^1 & \binom{t}{1}\binom{t-1}{0}(-1)^0 & \dots & 0 \\ \binom{t}{0}\binom{t}{0}(-1)^0 & 0 & \dots & 0 \end{bmatrix}.$$

In [46], the authors express this in a more compact form:

$$\mathbf{S}(u,v) = \mathbf{U}(u)\mathbf{G}\mathbf{V}(v)^T \tag{2.8}$$

where $\mathbf{G} = \mathbf{R}(m)\mathbf{C}\mathbf{R}(n)^T$ is constant. Constant values have important implications under the point of view of the implementation and optimization since computations can be replaced by memory accesses which can potentially be faster.

2.6.1 High-performance computing of Bézier surfaces

Computation of Bézier tensor-product constructions like surfaces or volumes is considered to be a computationally expensive task. Certain applications such as shape optimization in aerodynamics [47], flow modeling [48], simulation [49], non-rigid medical registration [50] and real-time computation of high resolution surfaces (concerned with this work) are often limited by the resolution, the complexity of the underlying data and real-time constraints.

Despite the number of applications using Bézier surfaces, the driving force behind their use and development is the computer-graphics community. In this field, surface representation and tessellation[3] [51] are the predominant uses for Bézier surfaces.

[3] Converting surfaces to triangle meshes.

Initially, tessellation was performed by the central processing unit (**CPU**) and then sent to the graphics processing unit (**GPU**) for further processing and visualization of results. The process of transferring data from the CPU memory space to the GPU memory space was considered a bottleneck, particularly for high-resolution surfaces. In order to address this issue, Espino *et al.* [52] proposed an algorithm in combination with a specific hardware architecture for a full integration in GPU.

Initially GPUs were conceived as co-processors dedicated to perform specific tasks related to computer graphics, later these evolved into programmable parallel processors capable of performing a wider range of tasks still in the domain of computer graphics. In order to enable this programmability of the GPUs, two new programmable units (*vertex processing units* and *fragment processing units*) were created. Some works made use of this approach to compute and render Bézier surfaces [53, 54, 55, 56, 57].

To be sure creation of vertex/fragments units was a step forward to increase the programmability and flexibility of GPUs. However, still many algorithms which could not strictly adhere to the particular data structures and mechanisms employed in computer graphics, could not take the advantage of the use of GPUs. In order to fill this gap, a new approach called *general purpose GPU* (**GPGPU**) was developed to allow greater programmability and flexibility using GPUs. GPGPU is currently provided by the CUDA [58] and OpenCL [59] frameworks, which have been used for a wide range of applications [60, 61, 62, 63].

Optimization strategies can be broadly classified into either hardware-specific cor algorithmic—regardless of the underlying implementation mechanisms.

Algorithmic strategies are generally concerned on the reduction of the number of operations to perform. In this line, [53] employs the matrix formulation in Eq. (2.7) which is more efficient than using Eq. (2.3) due to the constant nature of the $\mathbf{R}$ matrices. Later, in [46], the authors make use of Eq. (2.8) which considers also the product $\mathbf{RCR}^T$ as constant, thus reducing the number of operations to perform as well as exploit the spacial coherence of data. Other algorithmic strategies employ numerical approximations, like [61], which is based on forward-differencing [64], however, this methods are subject to error accumulation which might not be appropriate for some applications.

Hardware-specific strategies, on the other hand, are concerned with the efficient mapping of the algorithm on the hardware where the implementation of the algorithm is deployed. Fine-grained mapping of the algorithm is facilitated by frameworks like CUDA or OpenCL over shaders.

2.7 Electroanatomic Mapping of the Left Atrium

Atrial fibrillation the most common heart rhythm disorder, currently affecting around 8 million people. Fibrillation of the atria is produced by the irregular beat (quivering) of the upper chambers of the heart (atria), thus preventing the appropriated movement of blood flow into the ventricles. Although atrial fibrillation might not manifest any symptoms, it is associated with palpitations, fainting, chest pain an congestive heart failure. Furthermore, atrial fibrillation is associated to an increased risk of stroke. The growth in expectancy of life, will increase the incidence of atrial fibrillation up to more than 15 million people by 2050 [65].

The most common treatments for atrial fibrillation is radio-frequency catheter ablation (**RFCA**), together with medication and cardioversion. RFCA is a minimally invasive procedure in which the patient is prepared under local anesthesia. Catheters are inserted into the arteries and elevated to the left atrium chamber, where ablation is applied around the pulmonary veins; this prevents the activation of the electrical signals triggering the atrial fibrillation.

Before the ablation process, electrophysiologists need to know where and how ablation should be applied. In order to do this, electrophysiological models (containing both geometric and electrophysiologic information) are constructed—some authors report about the benefit of EAM-based methods [66]. As opposed to most of image-guided therapies, where models are built in a pre-operative stage, in RFCA, models are built in an intra-operative way (in some cases pre-operative models can be fused with intra-operative models). Computer-assisted systems using electromagnetic tracking technologies are employed to determine the position (geometric information) and the electrical properties (electrophysiology) of the atrium chamber.

There are two approaches to perform EAM: *point-to-point* acquisition and triangulation-based reconstructions [67] and *progressive reconstructions* [68]. Some works study the application of PSR in combination with high-resolution pre-acquired data [69, 70]. These approaches present added complexity and require additional resources (pre-acquired data through imaging) which might complicate their integration into clinical work-flows.

CHAPTER 3

SUMMARY OF CONTRIBUTIONS AND RESULTS

3.1 Paper I: Surface Reconstruction for Planning and Navigation of Liver Resection Procedures

Three-dimensional modeling of anatomical structures is a key component in planning and navigation of liver resection procedures. While vessels have been subject to an extensive research in the literature [38, 39, 71, 72, 73, 74], parenchyma has been given very little attention. Marching cubes [9] followed by smoothing [43] and decimation [42] (MCSD) is the more widespread technique used for modeling liver parenchyma.

In this work, we propose the use of Poisson surface reconstruction (PSR) [44] as a method for reconstruction of liver parenchyma. Since Poisson surface reconstruction operates on clouds of points, a simple yet efficient conversion method is proposed. In order to compare this proposal to marching cubes followed by smoothing and decimation, as well as to estimate the optimal reconstruction parameter for PSR, a multi-objective optimization framework where accuracy and complexity (number of polygons) are considered as equally important objectives.

PSR shows better reconstruction performance in terms of accuracy/complexity trade-off. This is supported by the fact that all PSR reconstructions lie in the

Pareto-optimal front; furthermore PSR reconstructions with depth parameter $d = 7$ show the absolute best accuracy/complexity trade-off for all the liver reconstructions in our data-set; this value can be considered as an estimation of the optimal reconstruction parameter. These PSR reconstructions exhibit an average reduction of 79.59% of polygons compared to MCSD models presenting similar smoothness and visual quality, however, efficient MCSD models (low number of polygons) are perceived by expert users as having less quality than PSR models. The median reconstruction error for Poisson reconstructions is 1.03 ± 0.23 mm, which supports that Poisson surface reconstruction is an adequate method for planning and navigation of liver resection procedures.

3.2 Paper II: A Novel Method for Planning Liver Resections Using Deformable Bézier Surfaces and Distance Maps

Specification of virtual resections is a key functionality included in a number of computer-assisted systems for planning and navigation of liver surgery. The most widespread strategies for the definition of virtual resections are *drawing-on-slices* and the use of *deformable cutting planes* [33].

In this work, a novel method based on deformable Bézier surfaces is proposed. Bézier surfaces possess geometric properties which make them an interesting approach for specification of virtual resections. Deformations of Bézier surfaces takes place by manipulation of a reduced set of control points. In this work, an improved interaction method which considers deformations based on groups of control points (instead to traditional one-to-one interaction) is proposed. In addition, this work describes a method for real-time computation and visualization of resection margins (safety margins) which are based on the computation of distance maps (from the virtual resection to the tumors). Finally, a strategy for computation of resected volume from virtual resections using Bézier surfaces is described.

The proposed method (Bézier) has been evaluated, together with a drawing-on slices approach (DS) [6] and a deformable cutting planes approach (CP) [5] by 5 gastro-intestinal surgeons at Oslo University Hospital. The experimental results show that the planning time for the proposed method is as fast as *state-of-the-art* CP and DS, which indicates adequacy for its integration in real clinical work-flows. Planning liver resections with the proposed method shows superior preservation of the resection margin in terms of deviation[1] (0.42 mm) compared to CP (-1.49 mm) and DS (-4.74 mm). Bézier also shows higher reproducibility of

[1] Negative deviation indicates resection was closer to the tumor than the limit established by the margin (violation of margin).

surgery planning results in terms of resected volume deviation[2] (-0.4%) compared to DS (-1.39%) and CP (-2.40%). Furthermore, Bézier provides smooth virtual resections presenting high feasibility to be performed surgically.

3.3 Paper III: High-Performance Computation of Bézier Surfaces on Parallel and Heterogeneous Platforms

Bézier surfaces are mathematical constructions employed in a wide variety of applications. These surfaces are known to be computationally expensive, which may limit their application. In the literature, some works propose parallelization strategies to improve the performance for the computation of Bézier surfaces, however, these approaches are mainly focused on applications related to visualization and computer graphics.

In this work a new method for computing Bézier surfaces (MLE), together with parallelization strategies to efficiently map the proposed method onto different platforms hardware platforms (CPUs and GPUs). In addition, and in the line of the latest computing trends, we propose computing mechanisms which exploit CPU-GPU cooperation mechanisms in heterogeneous CPU-GPU systems with different hardware architecture. An exhaustive evaluation—including different data-types different surface degrees and resolution—of the method and comparison with a *"brute-force"* (BRF) approach and an efficient approach employed in [46] (MAT).

The experimental results show that MLE achieves speedups up to $3.12\times$ (for double precision) and $2.47\times$ (single-precision) running on a CPU compared to BRF and MAT. While running on GPU the speedups are even higher with up to $3.69\times$ (for double precision) and $13.14\times$ (single precision) using modern GPUs. The results also reveal a clear benefit for CPU-GPU cooperation schemes which can improve the performance up to $2.09\times$ with respect to GPU-only approaches. The proposed method, as well as the CPU-GPU cooperation mechanisms are easily generalizable to higher order Bézier constructions (e.g, volumes or hyper-volumes) as well as other Bézier formulations (e.g., rational Bézier) thus exhibiting great potential to reach applications beyond the Bézier surfaces employed in this work.

[2] Negative deviations indicate less resected volume respect to the reference resection plan.

3.4 Paper IV: Intra-Operative Modeling of the Left Atrium: A Simulation Approach Using Poisson Surface Reconstruction

In this work the use of Poisson surface reconstruction (PSR) beyond reconstruction of smooth organs (like the liver), and under low number of sparse samples is explored. Three-dimensional reconstruction of the atrium chamber in the heart for electroanatomic mapping (EAM) during radio-frequency catheter ablation presents both conditions. On one hand, the atrium chamber together with the pulmonary veins exhibit geometries with higher curvature and local features than the liver. On the other hand, the process of EAM implies the intra-operative sampling of a reduced number of points from the patient's atrium.

In the paper, a simulation framework is employed to evaluate and characterize PSR in terms of accuracy of the reconstruction. To this end, a data-set of 57 segmented left atria is used; this isolates from errors introduced by breathing and cardiac motion. In the simulation process, marching cubes with smoothing and decimation is employed as ground truth which is then used to compute the error generated by PSR. PSR reconstructions are obtained at different sampling rates ([100-600]) and at different sampling rates for different parts of the atrium (50%-90% of samples taken from the upper part containing the pulmonary veins). Gaussian noise, characterizing the error introduced by the use of electro-magnetic tracking technologies (NDI Aurora $\mu = 0.76\ mm, \rho = 0.67\ mm$), is added.

Results derived from the simulation show that, under low number of samples a median error of $4.52 \pm 0.22\ mm$ is expected. This error can be reduced down to $2.61 \pm 0.65\ mm$ by elevating the number of samples ($n = 600$). Obtaining more samples from geometrically complex areas than from smooth areas shows to be beneficial, however, results do not show any clear trend in terms of a preferred sampling distribution. Obtaining a *moderate-to-high* number of samples is convenient to capture the geometry of the pulmonary veins' ostia. The median error is not significantly affected by the introduction of Gaussian noise.

CHAPTER 4

Discussion

For nearly two decades, computer-assisted for surgery planning and navigation have made their way into the clinical reality. Liver surgery is one of the areas benefited by this technological development, however, the adoption of computer-assisted sytems for planning and navigation of liver resectios is still very low. Planning and navigation technologies require the generation of patient-specific models in different processing stages (see Section 2.3) wich, to date, are difficult to perform in an unsupervised manner with sufficient quality and reliability.

To be sure, segmentation of anatomical structures is currently the major bottleneck to increase the adoption of computer-assisted systems for planning and navigation in the clinical reality [32]. While segmentation is still considered a challenging task which requires some degree of human interaction, the emergence of new approaches such as convolutional neural networks in deep learning [75] show promising results for segmentation tasks—not only in the medical domain.

After segmentation of the anatomical structures is performed, the segmenation models are reconstructed into 3D surface models (mesh models). Despite the fact that automated 3D modeling is less challenging than segmentation, complexity of the models, quality and accuracy is sitll a subject of research. In Section 4.1 we discuss the contribution of this thesis concerning 3D modeling of liver parenchyma as well as the most relevant findings and their significance.

Three-dimensional models are then utilized for planning of resections. While there are some works pointing in that direction [31, 76], to the best of our knowledge, automatic planning of resections has not been yet achieved. In the transition from manual definition of the resection towards automated resection planning researchers have been focusing on providing tools to make resection planning as simple, fast and reliable as possible. In this context Section 4.2 discusses the approach presented in this thesis (use of Bézier surfaces and distance maps)—see Paper II (Chapter 7). While Paper II focuses on the methodological aspects of resection planning using Bézier surfaces, Paper III targets the high-performance computing aspects. Section 4.3 discusses the findings of this thesis in that matter.

Finally, the results on the application of PSR to intra-operative reconstruction of the left atrium are discussed in Section 4.4.

4.1 3D Modeling for Planning and Navigation of Liver Resection Procedures

Three-dimensional modeling of anatomical structures has segmented images as a primary input, and therefore, results of the modeling are tightly coupled to the accuracy provided by the segmented models. Artifacts, noise and errors present in the segmentation models will be directly propagated to the 3D reconstruction—this is particularly true for methods such as marching cubes [9] which strictly adhere to the underlying data. Hence, accurate surface models require, in first place, accurate segmentation models.

To be sure, vessels are the most complex (geometrically speaking) anatomical structures in the liver. As opposed to parenchyma and tumors, which can be represented with relatively smooth and simple meshes, vessels present high curvature and features of interest which can be of relatively small size. Unlike vessels, which have been subject to extensive study in the scientific literature, modeling of liver and tumors has not been given much attention.

Modeling of the parenchyma has traditionally been carried out by marching cubes, sometimes followed by smoothing and decimation stages. This approach does not only circunscribe to modeling of liver parenchyma, but is arguably the most used approach for modeling of anatomical structures. The two major disadvantages of this approach are:

- at least two parameters which have to be adjusted in order to obtain optimal results,
- and the fact that noise and artifacts (e.g. staircases) from segmentation models will be translated to the final 3D surface reconstruction.

The aim of the first work in this thesis (Paper I) is to investigate methods which can provide computer-assisted systems with good quality models in an automatic way.

One pertinent question at this point is: which characteristics make good quality models in the context of computer-assisted systems? The literature related to vessels modeling seem to provide a wide consensus on this question. Complexity of the resulting models (number of polygons), accuracy of the reconstruction, visual quality as perceived by end-users and smoothness of results emerge as the preferred indicators of quality [38, 39, 71, 72, 73, 74] in this context.

The motivations for choosing PSR are its smoothness properties and its reduced parametric space—if there is an optimal reconstruction parameter, the low number of possible values can potentially reduce the complexity of its estimation. Despite PSR is a very well known technique in the computer-graphics community, in the medical domain has not been applied very much, perhaps due to the type of data required (oriented clouds of points, rather than medical image semgentations).

While in the literature, accuracy of the reconstruction and complexity of the models are considered separately, there is an intrinsic relationship between the two parameters. Hence, a model presenting a low number of polygons would have less representation capacity to capture, for instance, small features and *vice-versa*. This idea is used in Paper I to establish a multi-objective optimization framework which is employed to find the best accuracy/complexity trade-off and therefore, the estimation of the best parameter (depth d). An assumption in this idea is that both accuracy and complexity are equally important objectives, which makes sense under the point of view of the application: accuracy allows precise planning and the use of surface models for registration during navigation while low complexity helps performing operations such as deformations or cuts in real-time.

In the evaluation of PSR and MCSD the experimental results indicate a better performance of PSR as a method since all the reconstructions lie in in the pareto-optimal front[1]. Furthermore, PSR reconstructions with parameter $d = 7$ are established as optimal solutions not only for all other PSR but also for all MCSD. Therefore $d = 7$ can be considered as a good estimation of optimal reconstruction parameter, which can be used to automate the process. Furthermore computing time of PSR is similar to MCSD.

Under a visual quality standpoint, the experimental results show that for reconstruction of parenchyma models, PSR achieves similar visual quality than MCSD reconstructions with an average of 79.59% reduced number of polygons. Efficient

[1]The pareto-optimal front is the set of all reconstructions for which no other solution improves simultaneously both accuracy and complexity (dominant solutions).

MCSD models are perceived by expert as having a degraded quality with respect to PSR with $d = 7$.

The results obtained in this work can not be directly applied to other anatomical structures, particularly those with high inter-subject variability in shape and size, as well as and local features of interest (e.g, tumors and vessels).

In Paper IV (see Sections 3.4, 4.4 and Chapter 9) the application of PSR is extended to modeling of more complex antomical structures, particularly the left atrium.

4.2 Planning Liver Resection Procedures

Planning liver resection procedures is supported in segmented medical images, 3D models or the combination of both. As highlighted in Section 2.5 the two most common approaches are: *drawing-on-slices* [6] (DS) where the interaction takes place in the segmented images (2D); and the use of *deformable cutting planes* [5] (CP) where the interaction takes place in the 3D virtual space.

The use of Bézier surfaces can be considered a type of *deformable cutting planes* strategy, where the mechanics of deformation differs from the proposal of Konrad-Verse *et al.* [5]. In [5], the authors propose a deformation mechanism with a very high degree of freedom—under certain parameters, these surfaces could potentially be deformed vertex by vertex—, as opposed to Bézier surfaces where deformation takes place by moving a reduced set of control points (16 points at most for a bi-cubic surface). This guarantees certain properties of the virtual resection which are not neccesarily preserved by free-form surfaces, namely, smoothness and degree of the surface. Virtual resections defined by a surface presenting small local deformations (non-smooth) or wavy surfaces (high-degree) are associated to resections which are difficult to perform surgically and, sometimes, even impossible. DS presents an intermediate approach, where on one hand, representation of traces is very flexible but the final surface is interpolated. These ideas are supported by the results in Paper III, where DS and Bézier approaches exhibits significantly less curvature than CP; experts' comments seem to point in the same direction (see Table 7.3 in Chapter 7). Some surgeons in our experiments, highlight the difficulty of DS, particularly to obtain resections with high curvature—in some cases DS would not even produce acceptable results. Time for completing the resection plan seem to be similar for all the methods regardless of their interaction mechanisms.

Computer-assisted systems for planning liver resections are tools which allow surgeons to represent virtual resections. In this line, and once the surgeon has imagined the desired resection plan, planning methods should help the surgeon representing that plan in a precise manner. According to our results, Bézier is the

method which presents less inter-subject variability, thus indicating the accuracy of representation of the desired plan.

In the proposed approach, continuous visualization of the resection margin is achieved through the computation of distance maps (surface to tumors). This improvement was already suggested by [5] as further improvement. The experimental results in Paper III show the convenience of visualization of resection margin; DS and CP which did not include this visualization, incurred in an elevated number of resection margin violations (31 out of 40 for CP and 28 out of 40 for DS) compared to Bézier (1 out of 40). Interestingly, the proposed visualization of the resection margin was not enough to prevent all violations of resection margin since small areas violating the resection margin can be hidden behind other 3D structures. A further improvement would then be the addition of a global indicator of violation which guarantees the resection is valid. Both DS and CP could be further improved by adopting the method for visualization of the resection margin proposed.

Despite the good performance of Bézier in all the measured parameters, some users suggest that for certain types of resection other methods can be more adequate, for instance, *quasi-planar* resections, like in hemihepatectomies could easily be represented in DS or CP.

4.3 High-Performance Computation of Bézier Surfaces

Computation of Bézier surfaces is considered to be computationally expensive. In real-time applications such as the use of Bézier surfaces for planning liver resection procedures computational efficiency is of paramount importance. Efficient computation of Bézier surfaces not only allows including subsequent processing stages (such as computing of distance maps), but also may enable the integration of our planning system into less powerful platforms (such as low-power and mobile platforms) which can easily reach the market.

Parallelization and optimization techniques for the computation of Bézier-related constructions—not only surfaces, but also volumes and hyper-volumes—has been subject of study in the literature, however, these works circumscribe mostly to the computer graphics industry. In Paper IV a list of relevant optimization works related to Bézier surfaces is listed.

The approach of this thesis work to improve the efficiency of computation of Bézier surfaces has been two-sided. On one hand we propose an multi-level algorithm (MLE) for computation of Bézier surfaces wich reduces the number of operations to be performed compared to a *"brute force"* (BRF) approach and an efficient approach presented in [46] (MAT). On the other hand, we propose efficient

mapping techniques for, CPU and GPU computing as well as for heterogeneous computing platforms, in which a GPU and a CPU cooperate together to increase the performance. Heterogeneous computing is in the line of the latest trends of computing and it has an increased relevance for platforms where CPU and GPU are tightly coupled, possibly sharing the same memory space.

The proposed algorithm exploits the fact that some computations in Bézier surfaces are repeated. Computing the result of these computations once and storing it for reuse, effectively changes computation by memory accesses. Comparing this approach to other methods like BRF (no reutilization) and MAT (some degree of reutilization), MLE shows a general increase of performance both in CPU and in GPU. In CPU the benefit of using MLE over MAT can be as high as $3.12\times$ for double-precision and as high as $25.47\times$ over BRF. In GPU the gain is even larger, with MLE being $3.69\times$ faster than MAT and up to $42.62\times$ faster than BRF. Interestingly, the trends indicate that as the degree of the surface increases, the speedup of MLE over MAT decreases (until eventual convergence).

In addition to the proposed algorithm and the mapping strategies onto GPUs, in this thesis work, we propose two cooperation strategies, namely SDC and DDC for devices where the memory space is shared by CPU and GPU. The difference between these approaches is that SDC statically assigns a part of the surface to each unit (either GPU or CPU) while in DDC, the processors would process elements as needed on-demand. In order to find the optimal workload for every processor in SDC, profiling of the algorithm is needed beforehand. DDC is only applicable to architectures of shared memory spaces with coherence mechanisms. The experimental results show a speedup as high as $1.22\times$ for SDC approaches and $2.09\times$ for DDC approaches.

The type of Bézier surfaces proposed in Paper II (bi-cubic) are well suited for its parallelization and optimization according to Paper III with better performance than BRF and MAT. One important consideration for the implementation in a planning platform is choosing where to compute the Bézier surface. The computation of the Bézier surface during surgery planning is only the first processing stage; this stage is, at least, followed by the computation of distance maps (safety margin) and the corresponding contours. If one chooses to compute the Bézier surface in GPU, the most reasonable approach is that GPU also performs the subsequent processing stages (which also need to be parallelized and mapped into GPU). This is due to the fact that GPU also computes the visualization and outputs the results to the screen. Therefore, it makes very little sense to transfer the Bézier surface to the CPU for the subsequent processing, and back again to GPU for visualization. Platforms with shared memory and coherence mechanisms, on the other hand, can exploit the memory coherence to do so.

4.4 Application of Poisson Surface Reconstruction to Electroanatomical Mapping

In Paper I, Poisson surface reconstruction (PSR) is applied to the reconstruction of the liver parenchyma with good results in terms of mesh complexity, accuracy, smoothness and visual quality. However, the input data provided to PSR in that work is a dense cloud of points, which can be considered as a favorable condition for PSR to work. Other applications, however, rely on a sparse cloud of points with a low number of samples. An example of these applications can be the intra-operative electroanatomic mapping (EAM) of the left atrium, where samples from the atrium of the patient are collected (3D positions) during the surgical intervention. The nature of the intervention makes difficult the acquisition of an elevated number of samples.

Under the condition of low number of samples, accuracy plays a crucial role and therefore, multi-objective optimization presented in Paper I is not advisable. Instead, one can choose a depth parameter high enough to capture the geometry as accurate as possible.

The approach presented in Paper IV is a simulation process, where a database of 57 left atria are reconstructed under two different condition variables. One variable is the total number of samples acquired which varies from 100 to 600, which are values found in the literature. The second variable is the proportion of the samples acquired from the upper part of the atrium—which is considred geometrically more complex due to the pulmonary veins—which varies from 0.5 to 0.9 (where 0.5 indicates uniform sampling of the atrium). During EAM, the electrophysiologists will naturally take more samples from the upper part of the atriums not only because is the more comples, but also because is the most interesting part for the surgical procedure.

The simulation reveals that even when PSR would not obtain a faithful representation of the pulmonary veins, which geometrically present high curvature features, the areas of interest (pulmonary veins ostia) which present a moderate curvature, together with the rest of the atrium, are reconstructed properly under a *moderate-to-high* number of samples where the error can be as low as $2.61 \pm 0.05\ mm$. PSR will benefit from the acquisition of higher number of samples from the upper part of the atrium. As in the case of the liver, PSR behaves with resiliency to noise, which is very convenient in the case of intra-operative sample acquisitions performed with the use of electromagnetic tracking technologies.

Although the results are encouraging, further evaluation under more realistic conditions would be required to evaluate the real impact of PSR for the reconstruction

of 3D models for EAM. Technologies such as contact force-sensing catheters [77] can be an advantage in order to increase the number of samples acquired.

CHAPTER 5

CONCLUSION

As previously discussed in Section 2.3, from the acquisition of medical images to the use of computational models for surgical planning and navigation in liver resection, there are a few processing stages applied. In this thesis, three main bodies of work have been focused on improving methods in these stages, particularly the generation of 3D models (parenchyma) and the surgical planning. The remaining body was concentrated on the extension of some of the methods to other organs different from the liver.

First, the application of Poisson surface reconstruction (PSR) for the generation of 3D models of the parenchyma is proposed and evaluated. A multi-objective optimization framework which considers complexity of the resulting 3D meshes and accuracy of the reconstruction is proposed to estimate the best reconstruction parameter. The importance of the best parameter estimation is that the method can be applied in an automatic manner. Comparison with *state-of-the-art* marching cubes followed by smoothing and decimation (MCSD) shows that PSR obtains better reconstructions (in terms of accuracy/complexity trade-off). PSR models with similar smoothness and visual quality properties as MCSD models, present up to 79.59% less polygons than MCSD models.

Secondly, a new method for planning liver resection procedures is proposed. The method is based on the use of bi-cubic Bézier tensor-product surfaces, the visual-

ization of resection margin through thresholded distance-maps and an interaction technique which groups control points combining both simplicity and flexibility of interaction. The method was evaluated and compared together with a *drawing-on-slices* (DS) approach and the use of *deformable cutting planes* (CP) which, to date, are the *state-of-the-art*. The evaluation results show the proposed method to be as fast as DS and CP, which indicates the adequacy of the method to be integrated in real clinical work-flows. The proposed method also shows higher rate of preservation of resection margin compared to both CP and DS. In addition, the experimental results show that the reproducibility of results is higher than that of CP and DS. Smoothness of results is as high as DS and higher than CP, which indicates that the resulting resections are easier to perform surgically.

While one of the bodies of work concentrates on the methodological and clinical aspects of the use of Bézier surfaces for surgery planning, another body of work is focused on the high-performance computing aspects. High-performance computing aspects are important since, to some extent, determine the implementation possibilities into real systems. In order to improve the performance of the computation of Bézier surfaces, on one hand, a multi-level evaluation (MLE) of Bézier tensor-product surfaces is proposed. Along with this method, a set of techniques for mapping the method onto different computing architectures (GPU and CPU) are proposed. Additionally CPU-GPU cooperation mechanisms are explored. The method and the mapping techniques are evaluated and compared with a *"brute-force"* (BRF) and an efficient (MAT) approaches. The results obtained show the proposed method outperform BRF and MAT in both GPU and CPU (for very complex surfaces MAT and MLE present similar performance). Proposed CPU-GPU cooperation mechanisms increase this performance even further and open the possibility of efficient computation Bézier surfaces in mobile platforms.

Finally, the last body of work explore the application of PSR to reconstruction of the left atrium for electroanatomic mapping (EAM). This application presents different characteristics to reconstruction of the liver (i.e., sparse low number of samples as input and a more complex geometry to reconstruct). The application of PSR is evaluated through a simulation process using a data-set of 57 segmented left atria; characteristic noise from electromagnetic tracking technologies is also included. The experimental results show the reconstruction error is similar to other works in the literature. Reconstruction of areas of interest such as the pulmonary veins typically require a *moderate-to-high* number of samples. PSR shows noise resiliency which makes it a valid method to consider for the reconstruction of the left atrium during EAM.

Bibliography

[1] Lindsey A Torre, Freddie Bray, Rebecca L Siegel, Jacques Ferlay, Joannie Lortet-tieulent, and Ahmedin Jemal. Global Cancer Statistics, 2012. *CA: a cancer journal of clinicians.*, 65(2):87–108, 2015. ISSN 1542-4863 (Electronic). doi: 10.3322/caac.21262.

[2] P Kolstad. Cancer in Norway. Technical Report 3, 2014.

[3] J Belghiti and R Kianmanesh. Surgical treatment of hepatocellular carcinoma. *HPB : the official journal of the International Hepato Pancreato Biliary Association*, 7(1):42–49, 2005. ISSN 1365-182X; 1365-182X. doi: 10.1080/13651820410024067;10.1080/13651820410024067.

[4] P C Simmonds, J N Primrose, J L Colquitt, O J Garden, G J Poston, and M Rees. Surgical resection of hepatic metastases from colorectal cancer: A systematic review of published studies. *British Journal of Cancer*, 94(7):982–999, 2006. ISSN 0007-0920. doi: 10.1038/sj.bjc.6603033.

[5] O Konrad-Verse, Arne Littmann, and Bernhard Preim. Virtual Resection with a Deformable Cutting Plane. *SimVis*, (1), 2004.

[6] László Ruskó, Ilona Mátéka, and András Kriston. Virtual volume resection using multi-resolution triangular representation of B-spline surfaces. *Computer methods and programs in biomedicine*, 111(2):315–29, aug 2013. ISSN 1872-7565. doi: 10.1016/j.cmpb.2013.04.017.

[7] M Peterhans, A vom Berg, B Dagin, D Interbitzin, C Baur, D Candinas, and S Weber. A navigation system for open liver surgery: design, workflow and

first clinical applications. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 7(1):7–16, 2011. doi: 10.1002/rcs.

[8] Sebastian Rohl, Sebastian Bodenstedt, Stefan Suwelack, Rudiger Dillmann, Stefanie Speidel, Hannes Kenngott, and Beat P Muller-Stich. Dense GPU-enhanced surface reconstruction from stereo endoscopic images for intraoperative registration. *Medical physics*, 39(3):1632–45, mar 2012. ISSN 0094-2405. doi: 10.1118/1.3681017.

[9] Timothy S Newman and Hong Yi. A survey of the marching cubes algorithm. *Computers {&} Graphics*, 30(5):854–879, 2006. ISSN 00978493. doi: 10.1016/j.cag.2006.07.021.

[10] Yoshihiro Mise, Thomas A Aloia, Kristoffer W Brudvik, Lilian Schwarz, Jean-Nicolas Vauthey, and Claudius Conrad. Parenchymal-sparing hepatectomy in colorectal liver metastasis improves salvageability and survival. *Annals of surgery*, 263(1):146–152, 2016.

[11] Giovanni Vennarecci, Andrea Laurenzi, Roberto Santoro, Marco Colasanti, Pasquale Lepiane, and Giuseppe Maria Ettorre. The ALPPS procedure: A surgical option for hepatocellular carcinoma with major vascular invasion. *World Journal of Surgery*, 38(6):1498–1503, 2014. ISSN 14322323. doi: 10.1007/s00268-013-2296-y.

[12] C Couinaud. *Le foie: {é}tudes anatomiques et chirurgicales*. Masson {&} Cie, 1957.

[13] Sherif R Z Abdel-Misih and Mark Bloomston. Liver anatomy. *Surgical Clinics of North America*, 90(4):643–653, 2010.

[14] Henri Bismuth. Revisiting liver anatomy and terminology of hepatectomies. *Annals of surgery*, 257(3):383–6, mar 2013. ISSN 1528-1140. doi: 10.1097/SLA.0b013e31827f171f.

[15] Pietro Majno, Gilles Mentha, Christian Toso, Philippe Morel, Heinz O. Peitgen, and Jean H D Fasel. Anatomy of the liver: An outline with three levels of complexity - A further step towards tailored territorial liver resections. *Journal of Hepatology*, 60(3):654–662, 2014. ISSN 01688278. doi: 10.1016/j.jhep.2013.10.026.

[16] Ester Vanni and Elisabetta Bugianesi. Obesity and liver cancer. *Clinics in liver disease*, 18(1):191–203, feb 2014. ISSN 1557-8224. doi: 10.1016/j.cld.2013.09.001.

[17] J H Zhong, A C Rodriguez, Y Ke, Y Y Wang, L Wang, and L Q Li. Hepatic resection as a safe and effective treatment for hepatocellular carcinoma involving a single large tumor, multiple tumors, or macrovascular invasion. *Medicine (Baltimore)*, 94(3):e396, 2015. ISSN 0025-7974. doi: 10.1097/md.0000000000000396.

[18] Evangelos P Misiakos, Nikolaos P Karidis, and Gregory Kouraklis. Current treatment for colorectal liver metastases. *World journal of gastroenterology : WJG*, 17(36):4067–4075, sep 2011. ISSN 2219-2840. doi: 10.3748/wjg.v17.i36.4067.

[19] E. Vibert, T. Perniceni, H. Levard, C. Denet, N. K. Shahri, and B. Gayet. Laparoscopic liver resection. *British Journal of Surgery*, 93(1):67–72, 2006. ISSN 00071323. doi: 10.1002/bjs.5150.

[20] Hironori Kaneko, Sumito Takagi, Yuichiro Otsuka, Masaru Tsuchiya, Akira Tamura, Toshio Katagiri, Tetsuya Maeda, and Tadaaki Shiba. Laparoscopic liver resection of hepatocellular carcinoma. *American Journal of Surgery*, 189 (2):190–194, 2005. ISSN 00029610. doi: 10.1016/j.amjsurg.2004.09.010.

[21] A D Guerron, S Aliyev, O Agcaoglu, E Aksoy, H E Taskin, F Aucejo, C Miller, J Fung, and E Berber. Laparoscopic versus open resection of colorectal liver metastasis. *Surgical Endoscopy*, 27(4):1138–1143, 2013. ISSN 1432-2218. doi: 10.1007/s00464-012-2563-2.

[22] Irfan Ahmed and Paraskevas Paraskeva. A clinical review of single-incision laparoscopic surgery. *Surgeon*, 9(6):341–351, 2011. ISSN 1479666X. doi: 10.1016/j.surge.2011.06.003.

[23] AAsmund Avdem Fretland, Airazat M Kazaryan, Bjorn Atle Bjornbeth, Kjersti Flatmark, Marit Helen Andersen, Tor Inge Tonnessen, Gudrun Maria Waaler Bjornelv, Morten Wang Fagerland, Ronny Kristiansen, Karl Oyri, and Bjorn Edwin. Open versus laparoscopic liver resection for colorectal liver metastases (the Oslo-CoMet study): study protocol for a randomized controlled trial. *Trials*, 16(1):1–10, 2015. ISSN 1745-6215. doi: 10.1186/s13063-015-0577-5.

[24] Sean P. Cleary, Ho-Seong Han, Masakazu Yamamoto, Go Wakabayashi, and Horacio J. Asbun. The comparative costs of laparoscopic and open liver resection: a report for the 2nd International Consensus Conference on Laparoscopic Liver Resection. *Surgical Endoscopy*, 2016. ISSN 0930-2794. doi: 10.1007/s00464-016-4801-5.

[25] Steven M Strasberg and Carolyn Phillips. Use and dissemination of the brisbane 2000 nomenclature of liver anatomy and resections. *Annals of surgery*, 257(3):377–82, mar 2013. ISSN 1528-1140. doi: 10.1097/SLA.0b013e31825a01f6.

[26] Hartmut K Gumprecht, Darius C Widenka, and Christianto B Lumenta. BrainLab VectorVision Neuronavigation System: technology and clinical experiences in 131 cases. *Neurosurgery*, 44(1):97–104, 1999.

[27] Hinrich Staecker, Bert W O'malley, Howard Eisenberg, and B Emmerich Yoder. Use of the LandmarX™ Surgical Navigation System in Lateral Skull Base and Temporal Bone Surgery. *Skull Base*, 11(04):245–256, 2001.

[28] W Lamadé, G Glombitza, L Fischer, P Chiu, C E Cárdenas, M Thorn, H P Meinzer, L Grenacher, H Bauer, T Lehnert, and C Herfarth. The impact of 3-dimensional reconstructions on operation planning in liver surgery. *Archives of surgery (Chicago, Ill. : 1960)*, 135(11):1256–61, nov 2000. ISSN 0004-0010.

[29] Hauke Lang, Arnold Radtke, Milo Hindennach, Tobias Schroeder, Nils R Frühauf, Massimo Malagó, Holger Bourquain, Heinz-Otto Peitgen, Karl J Oldhafer, and Christoph E Broelsch. Impact of virtual tumor resection and computer-assisted risk analysis on operation planning and intraoperative strategy in major hepatic resection. *Archives of surgery (Chicago, Ill. : 1960)*, 140(7):629–38; discussion 638, jul 2005. ISSN 0004-0010. doi: 10.1001/archsurg.140.7.629.

[30] C Hansen, S Zidowitz, and B Preim. Impact of model-based risk analysis for liver surgery planning. *International Journal of Computer Assisted Radiology and Surgery*, 9(3):473–480, 2014. ISSN 1861-6429. doi: 10.1007/s11548-013-0937-0.

[31] Pablo Lamata, Félix Lamata, Valentin Sojar, Piotr Makowski, Laurent Massoptier, Sergio Casciaro, Wajid Ali, Thomas Stüdeli, Jérôme Declerck, Ole Jakob Elle, and Bjørn Edwin. Use of the Resection Map system as guidance during hepatectomy. *Surgical endoscopy*, 24(9):2327–2337, sep 2010. ISSN 1432-2218. doi: 10.1007/s00464-010-0915-3.

[32] Apollon Zygomalas, Dionissios Karavias, Dimitrios Koutsouris, Ioannis Maroulis, Dimitrios D. Karavias, Konstantinos Giokas, and Vasileios Megalooikonomou. Computer-assisted liver tumor surgery using a novel semiautomatic and a hybrid semiautomatic segmentation algorithm. *Medical and Biological Engineering and Computing*, 2015. ISSN 17410444. doi: 10.1007/s11517-015-1369-5.

[33] Bernhard Preim and Charl P Botha. *Visual Computing for Medicine: Theory, Algorithms, and Applications*. Newnes, 2nd edition, 2013.

[34] Kevin Cleary and Terry M. Peters. Image-Guided Interventions: Technology Review and Clinical Applications. *Annual Review of Biomedical Engineering*, 12(1):119–142, 2010. ISSN 1523-9829. doi: 10.1146/annurev-bioeng-070909-105249.

[35] P Markelj, D Tomaževič, B Likar, and F Pernuš. A review of 3D/2D registration methods for image-guided interventions. *Medical image analysis*, 16(3):642–61, apr 2012. ISSN 1361-8423. doi: 10.1016/j.media.2010.03.005.

[36] S Pieper, M Halle, and R Kikinis. 3D Slicer. In *2004 2nd IEEE International Symposium on Biomedical Imaging: Nano to Macro (IEEE Cat No. 04EX821)*, pages 632–635, 2004. ISBN 0-7803-8388-5. doi: 10.1109/ISBI.2004.1398617.

[37] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Transactions on Graphics*, 22(3):463, 2003. ISSN 07300301. doi: 10.1145/882262.882293.

[38] Jianhuang Wu, Renhui Ma, Xin Ma, Fucang Jia, and Qingmao Hu. Curvature-dependent surface visualization of vascular structures. *Computerized medical imaging and graphics : the official journal of the Computerized Medical Imaging Society*, 34(8):651–8, dec 2010. ISSN 1879-0771. doi: 10.1016/j.compmedimag.2010.07.006.

[39] Jianhuang Wu, Qingmao Hu, and Xin Ma. Comparative study of surface modeling methods for vascular structures. *Computerized medical imaging and graphics : the official journal of the Computerized Medical Imaging Society*, 37 (1):4–14, jan 2013. ISSN 1879-0771. doi: 10.1016/j.compmedimag.2013.01.002.

[40] WE Lorensen and HE Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, 1987.

[41] Erik Smistad, Anne C Elster, and Frank Lindseth. Real-time surface extraction and visualization of medical images using OpenCL and GPUs. *Norsk informatikkonferanse*, pages 141–152, 2012.

[42] W J Schroeder, J A Zarge, and W E Lorensen. Decimation of triangle meshes. *ACM SIGGRAPH Computer ...*, 26(2):65–70, 1992. ISSN 00978930.

[43] Ragnar Bade, Jens Haase, and Bernhard Preim. Comparison of Fundamental Mesh Smoothing Algorithms for Medical Surface Models. *SimVis*, 1(c):1–16, 2006. doi: 10.1.1.60.895.

[44] Michael Kazhdan and Matthew Bolitho. Poisson surface reconstruction. *Proceedings - Eurographics Symposium on Geometry Processing*, 2006.

[45] Les Piegl and Wayne Tiller. *The NURBS Book (2Nd Ed.)*. Springer-Verlag New York, Inc., New York, NY, USA, 1997. ISBN 3-540-61545-8.

[46] Raquel Concheiro, Margarita Amor, Montserrat Bóo, and Emilio J Padron. Free adaptive tessellation strategy of Bézier surfaces. In *Computer Graphics Theory and Applications (GRAPP), 2014 International Conference on*, pages 1–9. IEEE, 2014.

[47] Michele Andreoli, Janka Ales, and Jean-antoine Desideri. Free-form-deformation parameterization for multilevel 3D shape optimization in aerodynamics. Technical report, INRIA, FR, 2006.

[48] Andrea Manzoni, Alfio Quarteroni, and Gianluigi Rozza. Shape optimization for viscous flows by reduced basis methods and free-form deformation. *International Journal for Numerical Methods in Fluids*, 70(5):646–670, 2012. ISSN 1097-0363. doi: 10.1002/fld.2712.

[49] Hugo Casquero, Lei Liu, Carles Bona-Casas, Yongjie Zhang, and Hector Gomez. A hybrid variational-collocation immersed method for fluid-structure interaction using unstructured T-splines. *International Journal for Numerical Methods in Engineering*, 2015.

[50] Grzegorz Soza, Michael Bauer, Peter Hastreiter, Christopher Nimsky, and Günther Greiner. Non-rigid Registration with Use of Hardware-Based 3D Bezier Functions. *Proceedings of the 5th International Conference on Medical Image Computing and Computer-Assisted Intervention-Part II*, pages 549–556, 2002. ISSN 16113349. doi: 10.1007/3-540-45787-9_69.

[51] FJ Espino, M Bóo, M Amor, and JD Bruguera. Adaptive tessellation of NURBS surfaces. *Journal of WSCG*, 11(1-3), 2003.

[52] F. J. Espino, M. Bóo, M. Amor, and J. D. Bruguera. Hardware support for adaptive tessellation of Bézier surfaces based on local tests. *Journal of Systems Architecture*, 53(4):233–250, 2007. ISSN 13837621. doi: 10.1016/j.sysarc.2006.10.011.

[53] Raquel Concheiro, Margarita Amor, Emilio J Padrón, Marisa Gil, and Xavier Martorell. Rendering of Bézier Surfaces on Handheld Devices. In *21st International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 2013)*, 2013.

[54] Bo Li, Alistair A Young, and Brett R Cowan. GPU accelerated non-rigid registration for the evaluation of cardiac function. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2008*, 11(Pt 2):880–887, 2008.

[55] Michael Guthe, Aákos Balázs, and Reinhard Klein. GPU-based trimming and tessellation of NURBS and T-Spline surfaces. *ACM Transactions on Graphics*, 24(3):1016, 2005. ISSN 07300301. doi: 10.1145/1073204.1073305.

[56] Charles Loop and Jim Blinn. Real-time GPU rendering of piecewise algebraic surfaces. *ACM Transactions on Graphics*, 25(3):664, 2006. ISSN 07300301. doi: 10.1145/1141911.1141939.

[57] R Concheiro and M Amor. Synthesis of Bézier Surfaces on the GPU. In *International Conference on Computer Graphics Theory and Applications*, pages 110–115, Angers, France, 2010.

[58] NVIDIA. NVIDIA CUDA C Programming Guide, 2008.

[59] AMD. Introduction to OpenCL Programming, 2010.

[60] C U I Yuan-min. Real-time accurate Free-Form Deformation in terms of triangular Bézier surfaces. *Applied Mathematics*, 29(4):455–467, 2014.

[61] Michael Schwarz and Marc Stamminger. Fast GPU-based adaptive tessellation with CUDA. *Computer Graphics Forum*, 28(2):365–374, 2009. ISSN 01677055. doi: 10.1111/j.1467-8659.2009.01376.x.

[62] C. Dyken, M. Reimers, and J. Seland. Real-time GPU silhouette refinement using adaptively blended Bézier patches. *Computer Graphics Forum*, 27(1): 1–12, 2008. ISSN 01677055. doi: 10.1111/j.1467-8659.2007.01030.x.

[63] Iddo Hanniel, Adarsh Krishnamurthy, and Sara McMains. Computing the Hausdorff distance between NURBS surfaces using numerical iteration on the GPU. *Graphical Models*, 74(4):255–264, 2012.

[64] Bob Wallis. Tutorial on forward differencing. In *Graphics gems*, pages 594–603. Academic Press Professional, Inc., 1990.

[65] Yoko Miyasaka, Marion E Barnes, Bernard J Gersh, Stephen S Cha, Kent R Bailey, Walter P Abhayaratna, James B Seward, and Teresa S M Tsang. Secular trends in incidence of atrial fibrillation in Olmsted County, Minnesota, 1980 to 2000, and implications on the projections for future prevalence. *Circulation*, 114(2):119–125, 2006. ISSN 1524-4539.

[66] Deepak Bhakta and John M Miller. Principles of Electroanatomic Mapping. *Indian Pacing And Electrophysiology Journal*, 8(1):32–50, 2008.

[67] Daniel Reisfeld. Three-Dimensional Reconstruction of Intra-Body Organs., 2001.

[68] Patricia Chiang, Jianmin Zheng, Koon Hou Mak, Nadia Magnenat Thalmann, and Yiyu Cai. Progressive surface reconstruction for heart mapping procedure. *Computer-Aided Design*, 44(4):289–299, apr 2012. ISSN 00104485.

[69] Deyu Sun, Maryam E Rettmann, David R Holmes III, Cristian Linte, Bruce Cameron, Jiquan Liu, Douglas Packer, and Richard A Robb. Anatomic surface reconstruction from sampled point cloud data and prior models. *Studies in health technology and informatics*, 196:387, 2014.

[70] Deyu Sun, Maryam E Rettmann, Douglas Packer, Richard A Robb, and David R Holmes. Simulated evaluation of an intraoperative surface modeling method for catheter ablation by a real phantom simulation experiment. In *SPIE Medical Imaging*, pages 94152N—-94152N. International Society for Optics and Photonics, 2015.

[71] Steffen Oeltze and Bernhard Preim. Visualization of vasculature with convolution surfaces: method, validation and evaluation. *IEEE transactions on medical imaging*, 24(4):540–8, apr 2005. ISSN 0278-0062.

[72] C Schumann, S Oeltze, R Bade, B Preim, and HO Peitgen. Model-free Surface Visualization of Vascular Trees. In *EuroVis*, pages 283–290, 2007.

[73] Bernhard Preim and Steffen Oeltze. 3D visualization of vasculature: an overview. *Visualization in medicine and life sciences*, pages 1–20, 2008.

[74] Qingqi Hong, Qingde Li, and Jie Tian. Implicit reconstruction of vasculatures using bivariate piecewise algebraic splines. *IEEE transactions on medical imaging*, 31(3):543–53, mar 2012. ISSN 1558-254X. doi: 10.1109/TMI.2011.2172455.

[75] Yann LeCun, Bengio Yoshua, and Hinton Geoffrey. Deep learning. *Nature*, 521(7553):436–444, 2015. ISSN 0028-0836. doi: 10.1038/nature14539.

[76] C. Hansen, B. Lindow, S. Zidowitz, A Schenk, and H.-O. Peitgen. Towards Automatic Generation of Resection Surfaces for Liver Surgery Planning. *International Journal of Computer Assisted Radiology and Surgery*, 5(S1):117–121, may 2010. ISSN 1861-6410. doi: 10.1007/s11548-010-0442-7.

[77] Kurt S. Hoffmayer and Edward P. Gerstenfeld. Contact force-sensing catheters. *Current Opinion in Cardiology*, 30(1):74–80, 2015. ISSN 0268-4705. doi: 10.1097/HCO.0000000000000131.

[78] Richard Bryant, Alexis Laurent, Claude Tayar, Jeanne Tran van Nhieu, Alain Luciani, and Daniel Cherqui. Liver resection for hepatocellular carcinoma. *Surgical oncology clinics of North America*, 17(3):607–33, ix, jul 2008. ISSN 1055-3207. doi: 10.1016/j.soc.2008.02.002.

Part II

Publications

List of Publications

- *Rafael Palomar, Faouzi A. Cheikh, Bjørn Edwin, Azeddine Beghdadhi, Ole J. Elle.* **Surface Reconstruction for Planning and Navigation of Liver Resection Procedures**. Computerized Medical Imaging and Graphics. vol 53. pp 30-42. 2016. Elsevier.

- *Rafael Palomar, Faouzi A. Cheikh, Bjørn Edwin, Åsmund A. Fretland, Azeddine Beghdadi, Ole J. Elle.* **A Novel Method for Planning Liver Resections Using Deformable Bézier Surfaces and Distance Maps**. Computer Methods and Programs in Biomedicine. vol 144. pp 135-145. 2017. Elsevier.

- *Rafael Palomar, Juan Gómez-Luna, Faouzi A. Cheikh, Joaquín Olivares-Bueno, Ole J. Elle.* **High-Performance Computation of Bézier Surfaces on Parallel and Heterogeneous Platforms**. International Journal of Parallel Programming. pp 1-28. 2017. Springer.

- *Rafael Palomar, Faouzi A. Cheikh, Azeddine Beghdadi, Ole J. Elle.* **Intra-Operative Modeling of the Left Atrium: A Simulation Approach Using Poisson Surface Reconstruction**. International Conference on Medical Imaging and Virtual Reality. pp 354-365. 2016. Springer.

CHAPTER 6

PAPER I: SURFACE RECONSTRUCTION FOR PLANNING AND NAVIGATION OF LIVER RESECTIONS

Rafael Palomar · Faouzi A. Cheikh · Bjørn Edwin · Azeddine Beghdadhi · Ole J. Elle

Abstract Computer-assisted systems for planning and navigation of liver resection procedures rely on the use of patient-specific 3D geometric models obtained from computed tomography. In this work, we propose the application of Poisson surface reconstruction (PSR) to obtain 3D models of the liver surface with applications to planning and navigation of liver surgery. In order to apply PSR, the introduction of an efficient transformation of the segmentation data, based on computation of gradient fields, is proposed. One of the advantages of PSR is that it requires only one control parameter, allowing the process to be fully automatic once the optimal value is estimated. Validation of our results is performed via comparison with 3D models obtained by *state-of-art* Marching Cubes incorporating Laplacian smoothing and decimation (MCSD). Our results show that PSR provides smooth liver models with better accuracy/complexity trade-off than those obtained by MCSD. After estimating the optimal parameter, automatic reconstruction of liver surfaces using PSR is achieved keeping similar processing time as MCSD. Models from this automatic approach show an average reduction of 79.59% of the polygons compared to the MCSD models presenting similar smoothness properties. Concerning visual quality, on one hand, and despite this

reduction in polygons, clinicians perceive the quality of automatic PSR models to be the same as complex MCSD models. On the other hand, clinicians perceive a significant improvement on visual quality for automatic PSR models compared to optimal (obtained in terms of accuracy/complexity) MCSD models. The median reconstruction error using automatic PSR was as low as $1.03 \pm 0.23\ mm$, which makes the method suitable for clinical applications. Automatic PSR is currently employed at Oslo University Hospital to obtain patient-specific liver models in selected patients undergoing laparoscopic liver resection.

6.1 Introduction

Liver cancer is the second most common cause of cancer death worldwide [1]. Hepatocellular carcinoma (**HCC**), which accounts for 70% to 80% of the cases [2], presents a 5-year survival rate below 12% [3]. Colorectal cancer metastatic to the liver, on the other hand, develops in 50% of the cases of colorectal cancer, and presents 5-year survival rates up to 58% for selected patients undergoing liver resection [4].

Liver resection is the treatment of choice for patients with localized HCC [5] and can potentially be a curative therapy. A successful surgical resection requires the complete removal of the tumoral cells including a safety margin, preserving as much healthy tissue as possible [6]. Resections with adequate tumor-free margins lead to a better prognosis [7].

For more than a decade, computer-assisted surgical systems have been helping surgeons and other clinicians in the decision making process for planning and guiding surgical interventions. In the case of liver resection, these systems have recently found their way into the clinical practice, providing different patient-specific models: geometric, mechanical and functional as well as simulations. These models, ultimately rely on pre-operative computed tomography (**CT**) or magnetic resonance imaging (**MRI**).

The use of patient-specific 3D geometric models improve the capacity of surgeons to understand the liver and the underlying vascular structures. Using three-dimensional (**3D**) reconstructions have shown not only improvements in tumor localization and precision of surgery planning [8, 9, 10], but also an improved orientation and confidence of the surgeon while operating[11].

Marescaux *et al.* [12] were the first in implementing a system for planning and visualization applied to liver resection. In their approach, 3D models of the liver surface and vessels (tumors were artificially introduced) were built from MRI. The 3D models, based on simplex meshes [13], could be manipulated for visualization and allowed deformations for simulation purposes. Later, Selle *et*

al. [14] introduced the analysis of the hepatic vascular structures, which enabled the identification of the liver segments.

More recent approaches allowed the definition of a virtual resection as well as new visualization techniques. Reitinger *et al.* [15] presented a virtual reality environment in which the planning was performed by direct manipulation of 3D structures. Lamata *et al.* [16] also introduced a progressive clipping visualization based on the advancement of resection during surgery. Hansen *et al.* [17] encoded distance to critical structures in the surface representing the virtual resection.

Most of the visualization systems for planning and navigation of liver resection interventions, including all the aforementioned works, rely on the construction of 3D geometric models. These models have traditionally been obtained by isosurface extraction from segmented anatomical structures (parenchyma and venous vasculature) and tumors. Three-dimensional modeling of liver vasculature has been widely studied in the literature [18, 19, 20, 21, 22, 23], however, very little attention has been given to the modeling of parenchyma and tumors.

Marching cubes (**MC**) is the most widespread technique to obtain 3D models of anatomical structures in the liver. The method was originally developed by Lorensen and Cline [24] as an isosurface extraction method from scalar volumetric data. A number of extensions to the original MC have been proposed along the literature [25]. MC provides with accurate isosurfaces which strictly adheres to the underlying data. In the context of isosurface extraction from medical images, MC leads to surfaces with a high number of polygons and vertices, and the appearance of *staircase* artifacts. Different mechanisms like mesh smoothing [26, 27, 28] and decimation [29] have been employed to palliate these effects at the expense of adding more processing stages and control parameters.

Poisson surface reconstruction (**PSR**), originally proposed by Kazdhan *et al.* [30], is a 3D surface reconstruction technique which works on dense clouds of oriented points. In the literature, PSR finds application in reconstructions where clouds of points are inherent to the acquisition (scanning) technology. Examples of such applications can be reconstruction of scenes using structured light sensors [31], laser scanners [32] and stereo cameras [33]. The noise resilience of PSR makes it a remarkable method for coping with the inherent noise acquired by these technologies. In the medical imaging context, PSR has found very little application perhaps due to the fact that, in this domain, the input data consist of 3D scalar fields rather than oriented cloud of points. In order to apply PSR to medical images, Leonardi *et al.* [34] performed an estimation of the oriented clouds of points by fitting a plane to a segmented surface point and its *k-nearest* neighbors. Other approaches like [35] propose an extension to PSR to perform

adaptive polygonization with application to vessel modeling.

6.1.1 Major Contributions

The goal of this work is to address two common problems in 3D modeling techniques in the context of surgery planning and navigation. On one hand, integration of medical imaging, segmentation and 3D modeling methods into clinical workflows requires an elevated degree of automation. As an alternative to automation, expert users can perform manual adjustments of the reconstruction parameters. On the other hand, models employed in planning and navigation are increasingly becoming more complex, both in terms of resolution and dimensionality of the underlying data. Processing these models is very demanding in terms of computing power, especially with the introduction of real-time constraints.

To overcome these problems and with the aim of facilitating the integration of 3D modeling techniques in clinical workflows, we propose the application of PSR as a technique to model the liver surface (parenchyma). In order to adapt the segmented images to clouds of points (as required by PSR), a method based on gradient fields computation is described (Section 6.2.1).

Unlike traditional analysis of 3D modeling techniques, where different reconstruction parameters are considered separately, we propose a multi-objective optimization framework consisting of a bi-dimensional accuracy/complexity optimization space (Section 6.3) . The aim is to obtain optimal parameters which can lead to automatic 3D model reconstructions.

6.2 Materials and Methods

The approach presented in this work is used to obtain a 3D geometric model of the liver surface from a segmented volumetric image. Fig. 6.1 shows the processing stages of both our PSR approach and *state-of-the-art* MCSD.

Isosurface extraction is considered to be an automatic process, however, adding processing stages either after or before, introduces additional parameters which make the process more complicated and thus limiting its applicability and integration into clinical workflows. As opposed to MCSD, our strategy based on PSR requires only one parameter.

In the following, we describe our proposal. We start first by describing the computation of the oriented cloud of points from segmented images. Then we briefly describe the application of PSR to obtain the 3D models of the liver parenchyma.

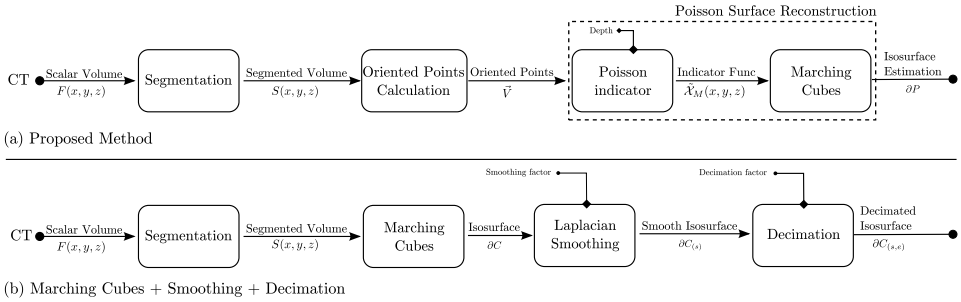


Figure 6.1: Processing stages for the proposed approach, based on PSR (a) and the *state-of-the-art* MCSD (b).

6.2.1 Oriented Cloud of Points from Liver Segmentation

As depicted in Fig. 6.1, medical images, obtained from CT, are represented by the scalar field defined as $F : \mathbb{R}^3 \rightarrow \mathbb{R}$ in which the point $\mathbf{p}_i = (x_i, y_i, z_i)$ with $i = 1, 2, ..., N$ is given a value representing an intensity level $F(\mathbf{p}_i) = v$. Through a segmentation process, the different points of the image, are assigned a class value (label) l_i according to either anatomical or functional criteria. The segmentation is then defined as a new scalar field $S : \mathbb{R}^3 \rightarrow \{l_1, ..., l_k\}$ with k the number of classes. In our case, since we are only interested in the liver parenchyma, our label map is constrained to the set $\{l_p = 0, l_b = 1\}$ which only considers the classes *parenchyma* and *background*. Fig. 6.2 shows an example of a scalar volume and its correspondent liver segmentation overlayed.

In order to find the oriented cloud of points $\vec{V} = \{(\mathbf{p}_i, \vec{n}_i)\}$, with $i = 1, 2, ..., N$, where $\vec{n}_i$ are the inward normals at $\mathbf{p}_i$, we exploit two properties of the gradient of the segmented images. On one hand we use the fact that, the gradient of the segmented image, at any point $\mathbf{p}_i$ belonging to the surface of the parenchyma ∂M, is orthonormal to the surface:

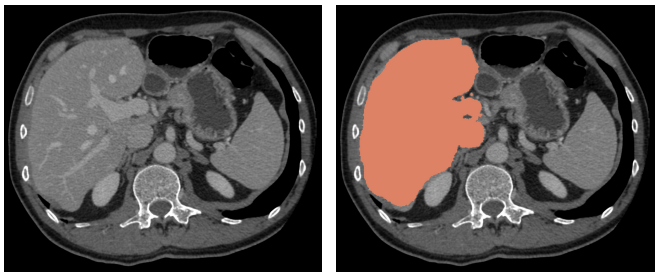
$$\nabla S(\mathbf{p}_i) \perp \partial M|_{\mathbf{p}_i} \Leftrightarrow \mathbf{p}_i \in \partial M. \tag{6.1}$$

On the other hand, the points $\mathbf{p}_i$ laying on the surface ∂M, can be discriminated in terms of the gradient of the segmented image:

$$\mathbf{p}_i \in \partial M \Leftrightarrow \|\nabla S(\mathbf{p}_i)\| > 0. \tag{6.2}$$

Using the properties established in Eq. 6.1 and Eq. 6.2, the definition of a oriented cloud of points $\vec{V}$ can be entirely expressed in terms of the gradient of the binary image:

$$\vec{V} = \{(\mathbf{p}_i, \vec{n}_i)\} = \{\ (\mathbf{p}_i, \nabla S(\mathbf{p}_i))\ \mid\ \|\nabla S(\mathbf{p}_i)\| > 0\ \}, \tag{6.3}$$



(a) (b)

Figure 6.2: Slice from CT imaging of the abdomen: (a) scalar field F; (b) segmentation of the parenchyma S overlayed on F.

with $i = 1, 2, ..., N$.

In contrast to Leonardi *et al.* [34], in which the authors estimate the normals by local plane fitting of the *k-nearest* neighbors, we propose the obtention of oriented clouds of points through the approximation of the gradient of segmented images. Our implementation obtains the gradient of a 3D binary image through the application of the 3D Sobel operator. This operator approximates the partial derivatives individually through convolution of the segmented parenchyma with kernel functions:

$$\nabla S = \left(\frac{\partial S}{\partial x}, \frac{\partial S}{\partial y}, \frac{\partial S}{\partial z} \right) = \left(\mathcal{K}_x * S, \mathcal{K}_y * S, \mathcal{K}_z * S \right), \tag{6.4}$$

where $\mathcal{K}_x, \mathcal{K}_y, \mathcal{K}_z \in \mathbb{R}^{3\times3\times3}$ are the third-order tensors representing the 3D Sobel kernel. The discrete nature of the Sobel operator makes that the condition stated in Eq. 6.2 becomes true for voxels near the surface of the liver (both inside and outside). Therefore, our implementation considers Eq. 6.3 only for those voxels belonging to the liver, this is, $S(\mathbf{p}_i) = l_p$.

6.2.2 Poisson Surface Reconstruction

PSR obtains smooth watertight (i.e. absence of holes in the mesh) triangulated approximations of surfaces from oriented cloud of points. The method utilizes a function fitting strategy which brings the benefit of both local and global fitting approaches together. Furthermore, PSR presents high resilience to noise.

Fig. 6.3 shows the general idea behind PSR. To obtain the surface, PSR uses the oriented cloud of points $\vec{V}$ to reconstruct an indicator function $\mathcal{X}_M : \mathbb{R}^3 \to \mathbb{R}$ of a parenchyma model M. This indicator function resembles the segmented image S, this is, a function in which the values inside the parenchyma are $l_p = 0$ and

the values outside the parenchyma are $l_b = 1$. The oriented cloud of points $\vec{V}$ can be thought of as a set of samples taken from the gradient of the indicator function $\nabla\mathcal{X}_M$. Then the problem can be stated as finding the scalar function $\mathcal{X}_M$ whose gradient best matches the cloud of points $\vec{V}$:

$$\tilde{\mathcal{X}}_M = \arg\min_{\mathcal{X}_M} \|\nabla\mathcal{X}_M - \vec{V}\| \tag{6.5}$$

with $\|.\|$ the Euclidean norm.

Khazdan *et al.* [30] makes use of the divergence operator to transform this problem into a variational problem optimized by solving the Poisson equation:

$$\Delta\mathcal{X} \equiv \nabla \cdot \nabla\mathcal{X} = \nabla \cdot \vec{V} \tag{6.6}$$

where $\mathcal{X}$ is an indicator function and $\vec{V}$ is a vector field.

The solution is represented through an adaptive and multi-resolution basis. More precisely, PSR constructs the minimal octree $\mathscr{O}$ with the property that every point sample falls into a leaf node at depth d. Intuitively, one can think of the depth as a parameter controlling the granularity of the mesh. Higher depth values thus lead to more complex models able to represent smaller features (and *vice versa*). Each octree node $o \in \mathscr{O}$ is associated with a function F_o. Bolitho *et al.* [36] proposes to associate a tri-variate B-spline (translated and scaled by the size of the node) to each node of the octree. The span $\mathcal{F}$ of translated and scaled B-splines defines the function space employed to solve the Poisson equation (Eq. 6.6). The system is solved through a finite elements approach where the system is discretized by using elements F_o as test functions. The solution is given by the function $\tilde{\mathcal{X}} \in \mathcal{F}_{\mathscr{O},F}$ such that:

$$\langle\Delta\tilde{\mathcal{X}}, F_o\rangle = \langle\nabla \cdot \vec{V}, F_o\rangle \mid \forall o \in \mathscr{O} \tag{6.7}$$

Finally, in order to obtain the surface approximation ∂P, isosurface extraction is applied to the average value of $\tilde{\mathcal{X}}$ at the sample positions:

$$\partial P \equiv \{\mathbf{q} \in \mathbb{R}^3 \mid \tilde{\mathcal{X}}(\mathbf{q}) = \gamma\} \text{ with } \gamma = \frac{1}{N} \sum_{(\mathbf{p}_i,\vec{n}_i)\in\vec{V}} \tilde{\mathcal{X}}(\mathbf{p}) \tag{6.8}$$

Different implementations of PSR have been proposed along the literature. The original approach ([30]) extends the method to the case of non-uniformly dis-

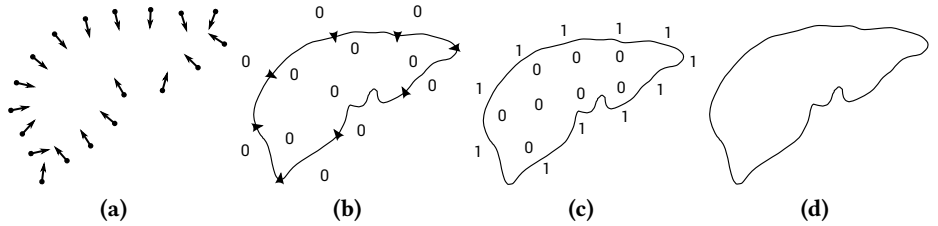


Figure 6.3: Overview of Poisson surface reconstruction in 2D contour of liver parenchyma: (a) oriented set of points $\vec{V}$; (b) gradient of the indicator function $\nabla\mathcal{X}_M$; (c) indicator function $\mathcal{X}_M$; (d) surface ∂P.

tributed cloud of points. PSR it is considered as an efficient method for which parallel implementations have also been proposed [36].

6.2.3 Experimental Setup

In this work, we propose the application of PSR for planning and navigation of liver resection procedures compared to the *state-of-the-art* MCSD. For comparison purposes, we employ the implementation (C++) of MCSD included in 3D Slicer [37] in its version 4.4.0.

Our PSR implementation (C++), which incorporates the computation of oriented cloud of points described in 6.2.1, is based on Doria and Gelas [38], who adapted the original by Kazhdan *et al.*[30].

The experiments are performed on a data set consisting of six CT volumes acquired from the abdomen of patients undergoing liver resection under the Oslo-CoMet study (NCT01516710) [39]. These images were acquired with parameters normally employed for diagnostics and surgical purposes. The images, presenting different image spacing and different liver volumes (Table 6.1), were segmented (parenchyma, vessels, tumors) by biomedical engineers in a semi-automatic way using ITK-Snap [40] and reviewed by two laparoscopic surgeons. All the data sets present anisotropic image spacing, and therefore, are prone to generate *staircase* artifacts which may be distracting and confusing, since these do not have any anatomical foundation.

6.3 Evaluation Criteria

In this section, the evaluation criteria to compare the PSR approach described in the Section 6.2 to MCSD are established. There are different ways to evaluate and compare 3D geometric models, however, in this work, we focus on the desirable properties for planning an navigation of liver resection procedures, this is, accuracy,

Table 6.1: CT data set used in the evaluation.

Name	Spacing (in mm)	Dimensions	Liver Volume (in cm^3)	CT Scanner
P_1	$0.79 \times 0.79 \times 0.62$	$512 \times 512 \times 358$	$2,289.25$	G
P_2	$0.82 \times 0.82 \times 3.00$	$512 \times 512 \times 153$	$2,289.39$	S
P_3	$0.76 \times 0.76 \times 0.60$	$512 \times 512 \times 461$	$1,853.56$	G
P_4	$0.63 \times 0.63 \times 0.62$	$512 \times 512 \times 299$	$1,579.10$	G
P_5	$0.68 \times 0.68 \times 2.50$	$512 \times 512 \times 179$	$1,503.66$	G
P_6	$0.70 \times 0.70 \times 0.62$	$512 \times 512 \times 396$	$2,326.09$	G

G: GE Medical Systems Lightspeed VCT.

S: Siemens Sensation 16.

mesh complexity and smoothness.

6.3.1 Mesh Complexity and Accuracy

Precise planning and navigation of liver resection procedures requires high accuracy of the 3D models. Some works like Wu *et al.* [23] rely on the Hausdorff distance as a metric for accuracy, however, Hausdorff distance is known to be sensitive to noise, which is inherent to medical images. In the same line as Oeltze and Preim [18] and Schumann *et al.*[19], we approximate the 3D reconstruction accuracy using surface distance based metrics, particularly the *point-to-surface* distance $\bar{\delta}(\partial M)$ has been employed. This distance is computed from all points $\mathbf{p}_i$ in a reference cloud of points $\vec{V}$ to all the points $\mathbf{q}$ in the model ∂M (either obtained by PSR or MCSD):

$$\bar{\delta}(\partial M) = \frac{1}{L}\sum_{i=1}^{L} \min_{\mathbf{q}\in\partial M} \|\mathbf{p}_i - \mathbf{q}\| \ , \ \mathbf{p}_i \in \vec{V}, \tag{6.9}$$

where L is the number of points in ∂M.

In mesh modeling, mesh complexity and accuracy are variables related to each other. Decreasing the mesh complexity generally leads to a decreased accuracy (and *vice versa*). Therefore, obtaining an acceptable trade-off between complexity and accuracy can then be established in terms of a multi-objective optimization problem (i.e. obtaining low complexity and high accuracy).

Number of polygons and mean *point-to-surface* distance are expressed in different units on different scales. To bring them into a unit-less objective space on the same scale, linear normalization is applied (Fig. 6.4), thus obtaining the normalized mean *point-to-surface* distance $\hat{\delta} \in [0, 1]$ and the normalized number of polygons

$\hat{N} \in [0,1]$:

$$\begin{cases} \hat{\delta}(\partial M) = \dfrac{\bar{\delta}(\partial M) - \bar{\delta}_{min}}{\bar{\delta}_{max} - \bar{\delta}_{min}} \\ \hat{N}(\partial M) = \dfrac{N - N_{min}}{N_{max} - N_{min}} \end{cases}, \tag{6.10}$$

where N is the number of points in ∂M, and $\bar{\delta}_{min}$, $\bar{\delta}_{max}$, N_{min} and N_{max} refer to the extreme values, considering all models (including MCSD and PSR) for a given parenchyma M.

By using normalization, it is established that both objectives are equally important and so, the optimal reconstruction is obtained by minimizing the objective score σ defined as:

$$\sigma(\partial M) = \sqrt{\hat{\delta}^2(\partial M) + \hat{N}^2(\partial M)}\,. \tag{6.11}$$

Models presenting a score value closer to 0, exhibit better trade-off between accuracy and complexity. Then, the best model among all PSR models ∂P_{best}, the best model among all MCSD models ∂C_{best} and the absolute best model ∂M_{best} can be computed as follows:

$$\begin{cases} \partial P_{best} = \arg\min\limits_{\partial P} \sigma(\partial P)\ ,\ \partial P \in \mathcal{P} \\ \partial C_{best} = \arg\min\limits_{\partial C} \sigma(\partial C)\ ,\ \partial C \in \mathcal{C} \\ \partial M_{best} = \arg\min\limits_{\partial M} \sigma(\partial M)\ ,\ \partial M \in \mathcal{P} \cup \mathcal{C} \end{cases}, \tag{6.12}$$

where $\mathcal{P}$ and $\mathcal{C}$ are the sets of all PSR and MCSD respectively, for a given parenchyma.

6.3.2 Mesh Smoothness

Liver parenchyma is inherently a smooth organ. Smoothness in the 3D model is essential for visualization since it contributes to the natural appearance of the organ. Evaluation of the smoothness is performed through the discrete mean Gaussian curvature as a metric of the roughness of the model ∂M:

$$\bar{K}(\partial M) = \frac{1}{N} \sum_{i=1}^{N} \frac{3(2\pi - \sum_j \gamma_j)}{A(\mathbf{p}_i)}\ ,\ \mathbf{p}_i \in \partial M, \tag{6.13}$$

where γ_j denote the angles between pairs of edges converging at $\mathbf{p}_i$ and $A(\mathbf{p}_i)$ is the sum of the areas of triangles having $\mathbf{p}_i$ as vertex. Values of the mean Gaussian

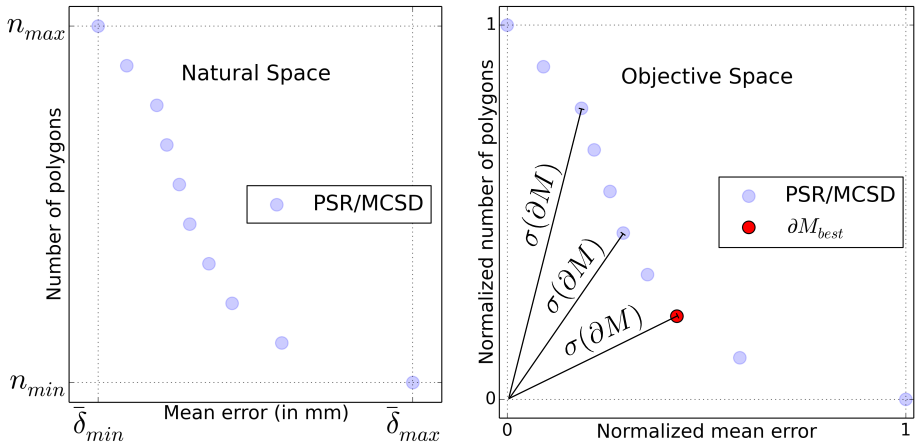


Figure 6.4: Normalization and objective score. (a) PSR and MCSD models represented in natural space; (b) PSR and MCSD in objective space where the scores $\sigma_{\partial M}$ are used to compute the best model (∂M_{best}) in terms of accuracy and complexity.

curvature closer to 0 are then interpreted as smoother appearance of the model.

Despite the fact that other curvature metrics (e.g. mean curvature) are available [41], Gaussian curvature seems to be the most widespread in comparisons similar to the one presented in this work [21, 23, 42].

For further comparisons in terms of accuracy and complexity, it will be useful to consider also the MCSD models ∂C_{sim} which present the most similar smoothness to the best PSR models ∂P_{best}. These models ∂C_{sim} can be computed as:

$$\partial C_{sim} = \arg\min_{\partial C} |\bar{K}(\partial C) - \bar{K}(\partial P_{best})| \ , \ \partial C \in \mathcal{C}. \tag{6.14}$$

6.3.3 Processing Time

3D models supporting planning and navigation of liver resection procedures are computed *pre-operatively* (i.e. before the operation) and therefore, there is no need to attend to real-time constraints. However, it is desirable that new methods involved in surgery planning and navigation, do not alter the clinical workflow significantly in a negative way.

Medical image processing is often performed in terms of a *region of interest* **ROI**, which represents a reduced set of the original data. In general, this greatly improves the performance of algorithms in terms of processing time, since the number of

elements to process decreases dramatically. Therefore, the evaluation of the performance (processing time) presented in this work is performed in the basis of minimal ROIs containing the liver.

6.3.4 Visual Quality

Perceived visual quality of the 3D models is an important evaluation criterion since it can affect not only the interpretation of the anatomy of the patient but also the comfort of the clinicians working with the 3D models. In the literature, visual quality is often discussed in terms of appearance of artifacts in the model and smoothness properties [19, 21, 22, 23].

Some works explicitly introduce visual quality criteria in the evaluation, like in Oeltze *et al.* [18] for 3D modeling or Feng *et al.* [43] for 3D visualization in laparoscopic surgery.

In the same line, we introduce subjective evaluation of visual quality based on the opinion of experts. In our experiment, we recruited 8 laparoscopic liver surgeons ([7-30] years of experience). Each expert is asked the following question: *"Evaluate the quality of the virtual models (parenchyma) for their use in resection planning and surgery guidance"*. In order to evaluate the quality of the 18 liver parenchyma models a 5-levels Likert scale was employed: *Very low* (1), *Low* (2), *Medium* (3), *High* (4), *Very high* (5). The evaluated parenchyma models correspond to the ∂P_{best}, ∂C_{best} and ∂C_{sim} models generated from the dataset described in Table 6.1. To ensure completeness of visualization, parenchyma models are shown together with vessels and tumors obtained by application of MCSD under the same parameters chosen arbitrarily. The experts were enforced to evaluate the quality of the parenchyma regardless of the quality of vessels and tumors. The evaluation was performed in a desktop computer (19 inches screen at 1600x1200@60Hz observed at a approximately 40cm distance), similarly to typical surgery planning stations.

In order to control possible biases, we followed the guidelines presented in [44]. Remarkable question design aspects taking into account are specificity of question (not allowing ambiguities or misinterpretations), simplicity and use of familiar words as well as clarity of correspondence between numeric values and qualitative tags in the answer fields. The models were presented in random order ($<$ 1 minute per model) to avoid patterns in sequential visualizations. Additionally, we allowed the surgeons to repeat the evaluation on the first 6 models to compensate for the initial lack of references in the evaluation.

6.4 Results

6.4.1 Mesh Complexity and Accuracy

PSR was applied to the 6 liver parenchyma of the data set. A total of 24 surface models were obtained considering PSR at depths $d \in \{5, 6, 7, 8\}$. Fig 6.5 shows the impact of this parameter on the reconstructions. On one hand, increasing the depth values, also increases the complexity of the model in number of polygons. On the other hand, increasing the depth values also increases the accuracy of the model. At depth $d = 5$, the median error of the 6 data sets is within the range $3.69 \pm 0.59\ mm$ while the absolute maximum (excluding outliers) is $8.75\ mm$; for $d = 6$, the median error decreases to $1.89 \pm 0.34\ mm$ with absolute maximum of $4.58\ mm$; for $d = 7$ we obtained a median error as low as $1.03 \pm 0.23\ mm$ with absolute maximum of $2.79\ mm$; and finally, for $d = 8$ the median error is within the range $0.82 \pm 0.36\ mm$ with a maximum error of $3.04\ mm$. As it is shown in Fig 6.5, the most prominent errors are located in areas presenting high curvature and concavities. For low depth values (coarser reconstructions), natural formation of concavities and prominent salients form areas of concentration of high errors. However, as the depth d increases (finer reconstructions), errors not only diminish in magnitude, but also distribute uniformly throughout the organ rather than concentrating in any particular areas.

MCSD presents a more complex parameter space than PSR. To represent the extent of this parameter space we obtained the 50 MCSD models $\partial M_{(s,e)}$ resulting from the combination of $s = \{0, 22, 45, 67, 90\}$ and $e = \{0, 0.1, 0.2, ..., 0.9\}$.

From the application of Eq. 6.12, the objective score value for all PSR and MCSD models was computed. The results, presented in Table 6.2, show that PSR with $d = 7$ always present the absolute best score values among all PSR and MCSD models, and therefore better behavior in terms of complexity and accuracy. PSR reconstructions different from $d = 7$ present lower score values than those of the best MCSD.

Another interesting comparison concerning accuracy and mesh complexity can be derived from the best MCSD (∂C_{best}), the best PSR models (∂P_{best}) and those MCSD models similar smoothness to ∂P_{best} (Eq. 6.14). The results reveal that the main contribution to the trade-off difference for ∂C_{sim} is the high number of polygons (460.1 K polygons average) compared to ∂P_{best} (72.4 K polygons average). The relative difference of mean error is low in the comparison of ∂C_{sim} ($0.81\ mm$ average), ∂P_{best} ($1.16\ mm$ average) and ∂C_{best} ($1.02\ mm$ average). Fig. 6.6 illustrates the compared models derived from P_1.

A broader view of the evaluation results is presented in Fig. 6.7. In this figure,

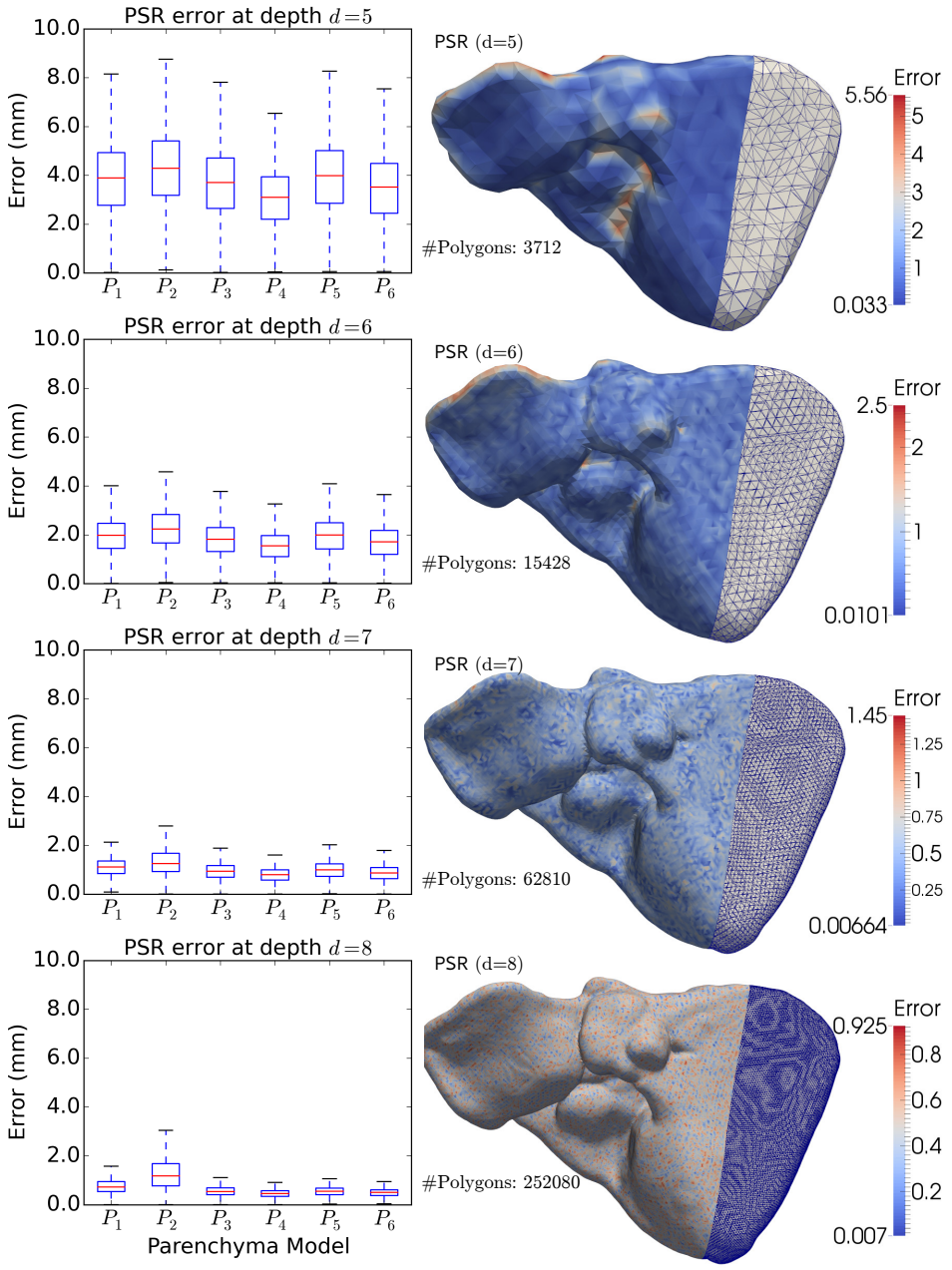


Figure 6.5: (left) Box-plot showing individual error distribution for PSR of the different parenchymas in the dataset ($P_1, ..., P_6$) at different depth values ($d \in \{5, 6, 7, 8\}$). (right) PSR for parenchyma P_1 at different depth values ($d \in \{5, 6, 7, 8\}$), showing density of polygons and color-coded projection of reconstruction error onto the PSR model.

Table 6.2: Objective score σ computed for all PSR and best MCSD models.

Parenchyma	PSR				Best
	$d = 5$	$d = 6$	$d = 7$	$d = 8$	MCSD
P_1	1.000	0.416	**0.184**	0.405	0.228
P_2	1.000	0.341	**0.248**	0.541	0.257
P_3	1.000	0.404	**0.165**	0.439	0.226
P_4	1.000	0.418	**0.161**	0.461	0.210
P_5	1.000	0.416	**0.152**	0.312	0.199
P_6	1.000	0.379	**0.148**	0.406	0.205

Minimum objective score highlighted in boldface.

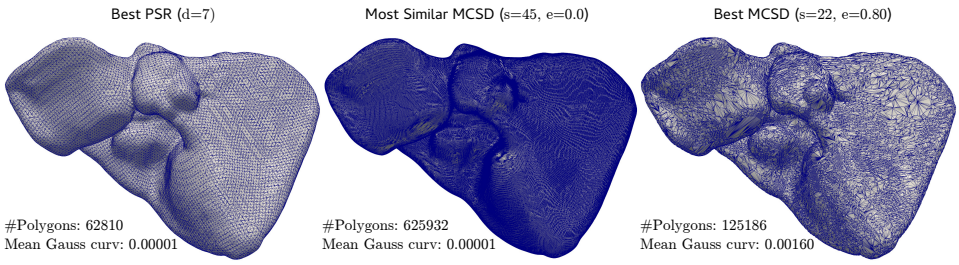


Figure 6.6: Mesh models correspondent to the best MCSD (accuracy/complexity), best PSR (accuracy/complexity) and its most similar MCSD (smoothness) derived from P_1.

the performance of all PSR and MCSD are shown in terms of complexity and accuracy. We computed the Pareto frontier, this is, the set of models which are not dominated by any other model in terms of both better accuracy and complexity. For a given model lying in the Pareto frontier, no other model can improve the two objectives, only one or none of the two. The results in Fig. 6.7 show that all PSR models ∂P, together with some MCSD, including the best MCSD ∂C_{best} models are part of the Pareto frontier. None of the most similar MCSD ∂C_{sim} are part of the Pareto frontier.

6.4.2 Mesh Smoothness

In order to evaluate the smoothness of the models, the mean Gaussian curvature $\bar{K}$ was computed for all PSR and MCSD models. A comparison of smoothness of PSR models with that of the best MCSD ∂C_{best} and most similar MCSD models ∂C_{sim} in terms of accuracy and complexity was performed. The results, presented in Table 6.4, show that, on one hand, the best MCSD ∂C_{best} generally presents much higher curvature values than most of PSR and the most similar MCSD ∂C_{sim}. On the other hand, the most similar MCSD ∂C_{sim} (9.67×10^{-3} average curvature)

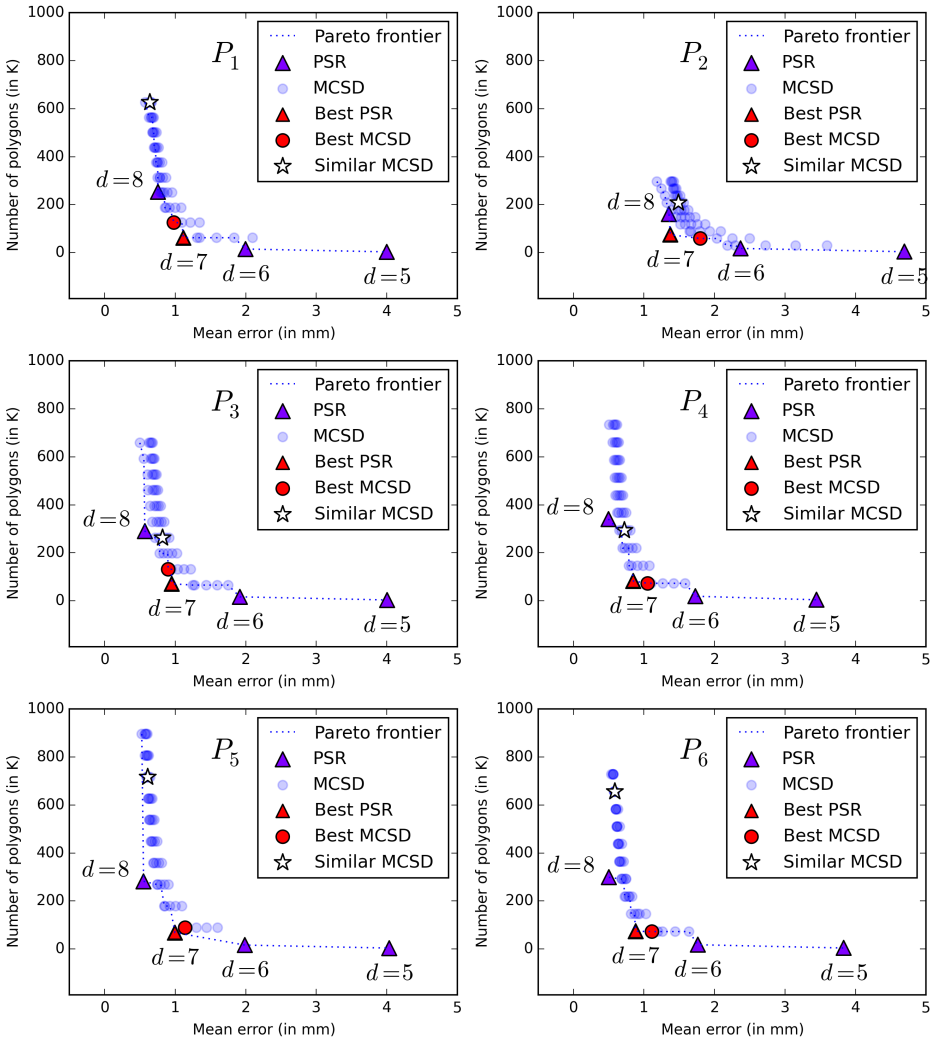


Figure 6.7: Mesh complexity and accuracy for all MCSD and PSR models derived from the evaluation data set. Best MCSD (complexity/accuracy), together with the best PSR (complexity/accuracy), its most similar MCSD (in smoothness) and the Pareto frontier are highlighted.

Table 6.3: Number of polygons, mean error and objective score σ for best PSR ∂P_{best} (accuracy/complexity), their most similar MCSD ∂C_{sim} (smoothness) and the best MCSD ∂C_{best} (accuracy/complexity).

Parenchyma	P_1			P_2			P_3		
Model	I	II	III	I	II	III	I	II	III
Polygons (in K)	**62**	625	125	75	207	**59**	**71**	263	131
Mean Error	1.11	**0.64**	0.98	**1.37**	1.48	1.79	0.94	**0.81**	0.89
Objective score	**0.18**	1.00	0.22	**0.24**	0.70	0.25	**0.16**	0.40	0.22
Parenchyma	P_4			P_5			P_6		
Model	I	II	III	I	II	III	I	II	III
Polygons (in K)	82	293	**73**	**68**	715	89	74	655	**72**
Mean Error	0.84	**0.72**	1.05	0.99	**0.61**	1.13	0.88	**0.58**	1.11
Objective score	**0.16**	0.40	0.21	**0.15**	0.79	0.19	**0.14**	0.89	0.20

(I) Best PSR ∂P_{best} | (II) Most similar MCSD ∂C_{sim} | (III) Best MCSD ∂C_{best}

Minimum values highlighted in boldface.

Table 6.4: Mean Gaussian Curvature $\bar{K}$, excluding outliers ([5-95] percentiles), computed for all PSR ∂P, best MCSD ∂C_{best} and most similar MCSD ∂C_{sim}.

Name	PSR				Similar MCSD	Best MCSD
	$d=5$	$d=6$	$d=7$	$d=8$		
P_1	4.56	0.29	0.19	0.19	0.19	1.60
P_2	11.17	31.96	0.706	14.34	0.72	6186.43
P_3	37.19	52.42	28.28	0.75	17.56	21743.83
P_4	8.22	14.86	151.16	3.15	77.46	20474.71
P_5	2.06	3.27	0.68	0.36	0.68	18494.12
P_6	7.61	5.78	0.32	0.20	0.32	681.94

All values expressed in a scale $\times 10^{-3}$.

are can only produce relatively less smoothness as their correspondent best PSR ∂P_{best} (0.03×10^{-3} average curvature).

6.4.3 Processing time

Our performance results (shown in Table 6.5) were obtained using a CPU Intel® Core™ i7-930 at 2.80GHz. The results were obtained on a basis of *best time of three* executions per model, considering the ROI containing the liver. For PSR, the execution time includes also the computation of the oriented cloud of points.

Mean execution time values show higher performance for similar MCSD models ∂C_{sim} ($\bar{t}_{\partial C_{sim}} = 12.05s$) over best MCSD models ∂C_{best} ($\bar{t}_{\partial C_{best}} = 14.08s$) and best PSR models ∂P_{best} ($\bar{t}_{\partial P_{best}} = 14.50s$), while variability of execution time

Table 6.5: Processing time (best of three executions, in seconds) the best PSR (including computation of oriented cloud of points) ∂P_{best}, best MCSD ∂C_{best} and most similar MCSD ∂C_{sim}.

Name	Liver Dimensions (Region of Interest)	PSR $d = 7$	Similar MCSD	Best MCSD
P_1	$310 \times 279 \times 330$	14.5	**9.4**	13.7
P_2	$279 \times 286 \times 68$	12.4	**5.8**	6.2
P_3	$296 \times 290 \times 264$	13.8	13.8	**13.0**
P_4	$297 \times 284 \times 272$	15.8	**15.5**	15.7
P_5	$330 \times 345 \times 285$	**15.2**	16.6	19.2
P_6	$280 \times 321 \times 323$	15.3	**11.2**	16.7
	Mean	14.5	**12.05**	14.8
	Std. Dev.	**1.24**	4.06	4.45

Minimum values highlighted in boldface.

is significantly lower for ∂P_{best} models ($s_{\partial C_{best}} = 1.24s$) compared to similar MCSD ∂C_{sim} ($s_{\partial C_{sim}} = 4.06s$) and best MCSD ∂C_{best} ($s_{\partial C_{best}} = 4.45s$).

6.4.4 Visual Quality

In this section we present the results of our experiments, where 8 surgeons evaluated the quality of 18 models (6 ∂P_{best}, 6 ∂C_{best} and 6 ∂C_{sim}) using a 5-level Likert scale (*Very low* (1), *Low* (2), *Normal* (3), *High* (4), *Very high* (5)). Subjective data obtained by questionnaires (Fig. 6.8a) show lower mean quality score for ∂C_{best} ($\bar{s}_{C_{best}} = 2.729$) versus ∂C_{sim} ($\bar{s}_{C_{sim}} = 3.291$) and ∂P_{best} ($\bar{s}_{P_{best}} = 3.333$).

In the same line as Feng *et al.* [43], we test the statistical difference between groups of methods regarding the perceived quality. In our case, the comparison between methods is performed by means of Welch two sample t-test using the R statistical environment (Fig. 6.8b). Statistical significance was obtained in the comparison between ∂C_{best} and ∂P_{best} ($p = 0.0006661$), and in the comparison between ∂C_{sim} and ∂C_{best} ($p = 0.001382$). No statistical significance was found in the comparison between ∂P_{best} and ∂C_{sim} ($p = 0.777$).

6.5 Discussion

6.5.1 PSR compared to state-of-the-art MCSD

The application of PSR to reconstruction of segmented images requires the transformation of the binary image (segmentation) to a cloud of oriented points. Perhaps due to this fact, PSR has found very little application in the medical domain. As opposed to Leonardi *et al.* [34], in which PSR is enclosed in a larger method for segmentation and reconstruction, our work focuses on the surface reconstruction

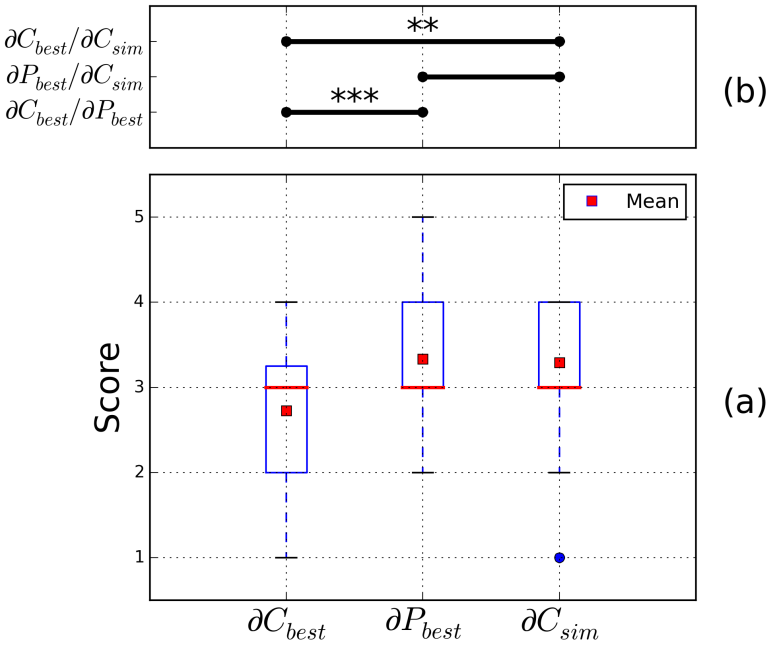


Figure 6.8: (a) Subjective evaluation results grouped by method; and (b) Welch two sample t-test results with ** significant at $p = 0.001382$ and *** significant at $p = 0.0006661$. Results were obtained from the evaluation of 18 models by 8 surgeons. Score values, are expressed in a 5 levels Likert scale: *Very low* (0), *Low* (1), *Medium* (3), *High* (4), *Very high* (5).

process and the comparison of PSR with the *state-of-the-art* MCSD.

The election of PSR as an alternative to be evaluated against *state-of-the-art* MCSD, is justified by the reduced parameter space, the resilience to noise and smoothness properties of PSR. Under the point of view of the application, the reduced parameter space of PSR (not only in number of parameters, but also in possible parameter values), translates into a simpler interaction by clinicians, and hence, a better integration in the clinical workflow. Empirically, we have determined the depth parameter space to be $d \in \{5, 6, 7, 8\}$ in the case of reconstruction of liver parenchyma. This set of values is clearly smaller than all reasonable combinations of smoothing and decimation factors for MCSD. Since the liver is a smooth organ, the good smoothness behavior exhibited by PSR contributes to the natural appearance of the models. Local features and staircase artifacts are reduced, allowing easier interpretation of the internal structures of the organ (vessels and tumors).

Our results suggest that PSR outperforms *state-of-the-art* MCSD for modeling liver parenchyma in different aspects. Overall, ∂P_{best} present reconstructions with better accuracy/complexity trade-off than *state-of-the-art* MCSD. Comparing models with similar smoothness and visual quality (∂C_{sim} and ∂P_{best}), ∂P_{best} models present a dramatic reduction of complexity (73% less triangles on average) at the cost of decreasing the accuracy slightly. Accuracy values for PSR, however, remain within clinically acceptable limits. Despite ∂C_{best} models can achieve similar complexity as ∂P_{best} in some cases, subjective evaluation experiments reveal that perceived visual quality of ∂C_{best} models is lower with statistical significance than those of ∂P_{best} and ∂C_{sim}.

Based on the accuracy/complexity space (Figure 6.5) two important observations can be made. First, PSR as method, defines the Pareto frontier, which suggests the goodness of the method over MCSD. Secondly, parameters to obtain ∂C_{best} depend on the input data while PSR obtains ∂P_{best} using the same parameter ($d = 7$), therefore indicating that PSR, as method, is more stable than MCSD. One implication of the stability of PSR is the possibility to perform optimal parameter estimation which can be used to produce automatic reconstructions (see Section 6.5.3).

As it is shown earlier in Figure 6.1, *state-of-art* MCSD requires two parameters (smoothing and decimation factors). In order to integrate automatic MSCD in clinical workflows, a set of values can be established for these parameters. In this line, the possibilities are either ignoring the application of smoothing and decimation, (i.e. considering $s = 0$ and $d = 0$) or choosing a set of "conservative" values that can slightly improve the reconstructions assuming that the optimal is unknown and unreachable. For smooth structures like the liver parenchyma,

some degree of smoothing and decimation is beneficial, not only under a polygon-reduction standpoint, but also for visualization purposes. As an alternative to automation, experts (biomedical engineers or clinicians) can manually set the parameters in a *try-and-fail* fashion for every patient. This can be a tedious process and does not guarantee optimality of results.

6.5.2 PSR Compared to Other Reconstruction Methods

The wide variety of reconstruction techniques in the literature makes the choice of the method a relevant question. The answer to this question is often driven and limited by the scope of the application. There are a number of surface reconstruction methods other than MCSD or PSR. In particular, multi-level partition of unity implicits (**MPUI**) [45] has been previously adopted for vessel modeling from clouds of points [19, 20, 23, 42, 46] and can be applied to liver modeling as well. This and other reconstruction methods are compared and discussed in [30]. In this comparison, MPUI shows generation of spurious surface sheets under noisy conditions, which are inherent in medical imaging, while PSR exhibits good noise resilience.

More recent approaches like dynamic particles [47, 48] provide high-quality meshes suitable for visualization and simulation purposes. These methods show more flexibility (regularity of triangulation and accuracy of reconstructions can be controlled) at the cost of a more complex parameter space. Although these approaches can be better suited for simulation, widening the parameter space complicates the integration into clinical workflows.

6.5.3 Accuracy, Complexity and Depth Parameter d

The choice of the depth parameter has an impact over the smoothness, accuracy and complexity of the reconstruction. Increasing the depth parameter value generally produces similar or superior smoothness, higher accuracy and higher complexity of the model. Similar works study accuracy and complexity as independent dimensions of the problem [19, 21, 23, 42]. For MCSD and PSR, there is a clear relationship between the complexity and accuracy of reconstruction. Low complexity meshes, for instance, present limited representation power for small features, and hence, decreased accuracy.

Considering the accuracy/complexity relationship in a multi-objective optimization framework present some interesting advantages. First, it provides a basis for comparing reconstructions with similar properties: ∂P_{best} vs. ∂C_{best} for optimal accuracy/complexity trade-off and ∂P_{best} vs. ∂C_{sim} for similar smoothness. Secondly, optimal parameters can be estimated. In this line, and by considering accuracy and complexity as equally important objectives (see normalization in

Section 6.3.1), our study suggests that for liver modeling, PSR models with $d = 7$ are optimal in terms of accuracy/complexity trade-off. Other applications might consider different weights for accuracy and complexity objectives, in which case, the optimal depth parameter may vary. In [34], Leonardi *et al.* employ $d = 5$ for kidney modeling attending to a reduction of polygons criterion, however, the authors do not provide any data related to accuracy of reconstruction nor comparison with other methods.

The accuracy/complexity analysis described in this work, can also support automatic generation of multi-resolution PSR models at different d values. Among other purposes, this approach can support the use of proxy geometries for mesh processing and visualization as in [49].

6.5.4 Integration of PSR in Clinical Workflows

The application domain of our study is planning and navigation of liver resection procedures. In this domain, visual realism and accuracy are of paramount importance. Visual realism is achieved through smoothness, which removes staircase artifacts and small features not needed for the visualization of the organ, thus reducing the complexity of visualization. Accuracy plays an important role since the model can be used as a base for clinical decisions as well as for *model-to-patient* registration (e.g. surgery navigation). Some of these clinical decisions are supported on operations performed directly on the models. Low complexity models can improve the performance of operations like mesh cutting, mesh volume or distance computations, present in computer-assisted systems for planning and navigation. By optimizing the depth parameter ($d = 7$) we obtained a fully automatic reconstruction method able to produce smooth models with better accuracy/complexity trade-off than the models generated by MCSD. The errors presented in the results for PSR at $d = 7$ (median errors within $1.03 \pm 0.23\ mm$) are clinically acceptable. For all this, we consider PSR a suitable candidate to replace *state-of-the-art* MCSD to model 3D for planning and navigation purposes.

The difference in computing time between PSR and MCSD is, for pre-operative purposes, negligible (Table 6.5), however, time is still far from real-time reconstructions, which might be of interest for other clinical workflows. The operations involved in PSR are subject to parallelization strategies using graphics processing units (**GPUs**). Works like [31] and [50] show the feasibility of using GPUs for real-time reconstruction of complex scenes using PSR. As intra-operative systems (e.g. surgical navigation) move towards the use of deformable models, aspects related to algorithm performance and parallelization capabilities will get more relevance in medial systems design.

Process automation during imaging, segmentation and 3D modeling is the key for

improving the adoption of 3D patient-specific models in clinical workflows. To be sure, the major bottleneck is segmentation (currently is subject to extensive research), which automation is still considered as a challenging task [51], thus requiring some degree of human interaction. While segmentation falls out of the scope of this work, the integration ability of PSR and how this can increase the adoption of 3D patient-specific models with the help of *state-of-the-art* tools, like MeVis Distant Services [51, 52, 53] or Fujifilm Synapse VINCENT [54], is a relevant question. PSR can be seamlessly integrated in any medical platform, provided that the platform is able to obtain segmented images (Figure 6.1). Open source software like 3D Slicer [37] or ITK-Snap [40] can also make use of this work not only for the reconstruction of liver surfaces, but also to investigate further applications (e.g. modeling of other anatomical structures).

Liver surgery is moving towards minimally invasive surgery (laparoscopy). The clinical advantages of this approach must take into consideration the increased complexity of the surgical procedure (reduced maneuverability and visual field). In this context, the use of 3D patient-specific models, together with intra-operative imaging (ultrasound) are becoming increasingly relevant. Recent studies highlight the advantages of integrating these models as part of augmented reality guidance systems in laparoscopy [55] and open surgery [53]. The combination of 3D patient-specific model, together with the latest trends in surgery, like robotic surgery, have the potential to make surgical interventions easier, faster and probably safer [56]. Some of these works highlight the importance of accuracy, and reduction of human interaction (automation of processes), topics which are widely studied and discussed in this work.

6.5.5 Application of PSR to Other Anatomical Structures

Our application of PSR to liver modeling (parenchyma) from scalar volumes inevitably raises the question of the adequacy of PSR to obtain 3D models of other anatomical structures. In the context of planning and navigation of liver resection procedures, the question reduces to whether PSR is a suitable method for reconstruction of tumors and vessels.

For tumors, due to their high variability in shape and size, as well as the need of preservation of local features, PSR demands a high d parameter value (associated to a relatively high number of polygons). Though this does not disqualify PSR to be applied to tumors, the advantage of PSR over MCSD is, at the least, not as powerful as for the liver surface. Further investigation is needed to determine the degree of adequacy of PSR for such structures.

In the case of PSR applied to vessels, preservation of high curvature and branches (concavities) demands a high value of the d parameter, resulting in models with

high number of polygons. To cope with this problem, Wu *et al.* [23] evaluates a variant of PSR (in that work referred to as *scale-adaptive* [**SA**]), which includes curvature-dependent polygonization (e.g. increasing/decreasing the size of triangles according to the local curvature) [35]. In [23], other methods including MC (without smoothing and decimation) are evaluated with application to vessel modeling. The authors, point at SA as a suitable method for reconstruction of vessels with applications to surgery planning. The methods evaluated by Wu *et al.* [23] could be also compared with another set of techniques (known as *model-based* methods) [20], widely used in the context of vessel modeling for surgery planning.

6.5.6 Future Work

Leonardi *et al.* [34] suggest the use of PSR to construct geometric models of other organs than kidney. In the same line, and despite the little attention PSR has been given in the medical domain, we believe that its use can be extended to other organs outperforming *state-of-the-art* methods.

Smooth organs absent of sharp features are, in principle, good candidates to undergo PSR. Evaluations similar to Wu *et al.* [23] could also consider the intrinsic relationship between number of polygons and error according to our multi-objective optimization framework. To the best of our knowledge, modeling of tumors has not been subject to an exhaustive evaluation like the one presented in this work or in Wu *et al.* [23], which can be of great interest.

New reconstruction methods that may arise, can be evaluated using this work as guideline. Despite the more complex parameter space of methods based on dynamic particles [47, 48], these can support an interesting comparison with PSR for modeling of anatomical structures for different purposes.

6.6 Conclusion

In this work, we propose the application of PSR to obtain patient-specific models of liver parenchyma for planning and navigation of liver resection procedures. For the application of PSR to medical images, we propose an efficient transformation of the segmented images to oriented cloud of points based on computing gradient fields. In order to make an automatic PSR, we found the PSR parameter obtaining the best accuracy/complexity trade-off ($d = 7$).

Comparing PSR with the *state-of-the-art* (MCSD) in terms of accuracy, complexity and smoothness, PSR shows better reconstruction performance and stability of results. This study also reveals that PSR liver models using the optimal parameter $d = 7$ not only are smooth, but also present better accuracy/complexity trade-off than MCSD models. Reconstructions obtained through automatic PSR ($d = 7$),

presents median errors within $1.03 \pm 0.23\ mm$, which makes them suitable for clinical applications. On average, these models have 79.59% less polygons compared to MCSD models with similar smoothness, while clinicians do not perceive a significant quality difference. Optimal PSR models $d = 7$, exhibit a significant improvement of visual quality compared to optimal MCSD in terms of accuracy/complexity trade-off. Automatic PSR can be seamlessly integrated in clinical workflows already using MCSD, since the processing time is similar to that of MCSD.

The contribution of this work, is therefore, a step towards the automation and quality needed for a wide adoption of 3D patient-specific models in the medical community. Currently, at Oslo University Hospital (The Intervention Centre), PSR is employed in a fully automatic way (after segmentation, which takes place in a semi-automated way) to obtain patient-specific models of liver parenchyma in selected patients undergoing laparoscopic liver resection. Fig. 6.9a shows a complete patient-specific liver model in which the parenchyma was obtained through PSR ($d = 7$) while MCSD was applied for vessels and tumors. During operation, (Fig 6.9b) the patient-specific model, which includes the resection path, helps the surgeons to perform the resection according to the pre-operative plan.

Bibliography

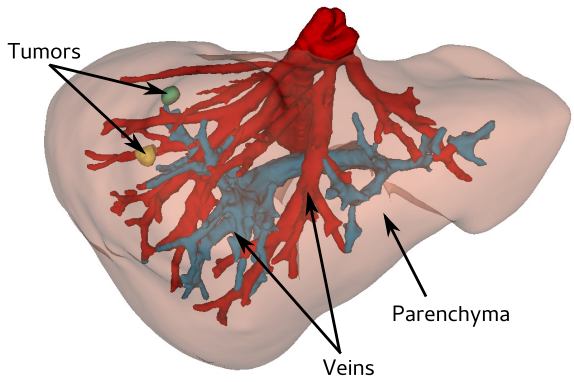
[1] Ahmedin Jemal, Freddie Bray, Melissa M Center, Jacques Ferlay, Elizabeth Ward, and David Forman. Global cancer statistics. *CA: a cancer journal for clinicians*, 61(2):69–90, 2011. ISSN 1542-4863. doi: 10.3322/caac.20107.

[2] Ester Vanni and Elisabetta Bugianesi. Obesity and liver cancer. *Clinics in liver disease*, 18(1):191–203, feb 2014. ISSN 1557-8224. doi: 10.1016/j.cld.2013.09.001.

[3] Hashem B El-serag. Hepatocellular Carcinoma. *The New England Journal of Medicine*, 365(12):1118–1127, 2011.

[4] Evangelos P Misiakos, Nikolaos P Karidis, and Gregory Kouraklis. Current treatment for colorectal liver metastases. *World journal of gastroenterology : WJG*, 17(36):4067–75, sep 2011. ISSN 2219-2840. doi: 10.3748/wjg.v17.i36.4067.

[5] Richard Bryant, Alexis Laurent, Claude Tayar, Jeanne Tran van Nhieu, Alain Luciani, and Daniel Cherqui. Liver resection for hepatocellular carcinoma. *Surgical oncology clinics of North America*, 17(3):607–33, ix, jul 2008. ISSN 1055-3207. doi: 10.1016/j.soc.2008.02.002.



(a)

(b)

Figure 6.9: (a) Complete patient-specific model including parenchyma (PSR), vessels (MCSD) and tumors (MCSD). (b) Use of a patient-specific model for guiding a liver resection surgical procedure.

[6] Timothy M Pawlik and Jean-Nicolas Vauthey. Surgical margins during hepatic surgery for colorectal liver metastases: complete resection not millimeters defines outcome. *Annals of surgical oncology*, 15(3):677–9, mar 2008. ISSN 1534-4681. doi: 10.1245/s10434-007-9703-2.

[7] Nobuhiko Taniai, Koho Akimaru, Hiroshi Yoshida, and Takashi Tajiri. Surgical treatment for better prognosis of patients with liver metastases from colorectal cancer. *Hepato-gastroenterology*, 54(78):1805–1809, 2007. ISSN 0172-6390.

[8] W Lamadé, G Glombitza, L Fischer, P Chiu, C E Cárdenas, M Thorn, H P Meinzer, L Grenacher, H Bauer, T Lehnert, and C Herfarth. The impact of 3-dimensional reconstructions on operation planning in liver surgery. *Archives of surgery (Chicago, Ill. : 1960)*, 135(11):1256–61, nov 2000. ISSN 0004-0010.

[9] Hauke Lang, Arnold Radtke, Milo Hindennach, Tobias Schroeder, Nils R Frühauf, Massimo Malagó, Holger Bourquain, Heinz-Otto Peitgen, Karl J Oldhafer, and Christoph E Broelsch. Impact of virtual tumor resection and computer-assisted risk analysis on operation planning and intraoperative strategy in major hepatic resection. *Archives of surgery (Chicago, Ill. : 1960)*, 140(7):629–38; discussion 638, jul 2005. ISSN 0004-0010. doi: 10.1001/archsurg.140.7.629.

[10] C Hansen, S Zidowitz, and B Preim. Impact of model-based risk analysis for liver surgery planning. *International Journal of Computer Assisted Radiology and Surgery*, 9(3):473–80, may 2014. ISSN 1861-6429. doi: 10.1007/s11548-013-0937-0.

[11] Pablo Lamata, Félix Lamata, Valentin Sojar, Piotr Makowski, Laurent Massoptier, Sergio Casciaro, Wajid Ali, Thomas Stüdeli, Jérôme Declerck, Ole Jakob Elle, and Bjørn Edwin. Use of the Resection Map system as guidance during hepatectomy. *Surgical endoscopy*, 24(9):2327–37, sep 2010. ISSN 1432-2218. doi: 10.1007/s00464-010-0915-3.

[12] Jacques Marescaux, Christophe Koehl, Yves Russier, Didier Mutter, Endocrine Surgery, and Groupe Epidaure. Virtual Reality Applied to Hepatic Simulation : The Next Revolution. *Annals of surgery*, 228(5):627–634, 1998.

[13] Hervé Delingette. Simplex Meshes : a General Representation for 3D Shape Reconstruction. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 856—-859, 1994.

[14] Dirk Selle, Bernhard Preim, Andrea Schenk, and Heinz-Otto Peitgen. Analysis of vasculature for liver surgical planning. *IEEE transactions on medical*

imaging, 21(11):1344–57, nov 2002. ISSN 0278-0062. doi: 10.1109/TMI.2002.801166.

[15] Bernhard Reitinger and Alexander Bornik. Liver surgery planning using virtual reality. *Computer Graphics and Applications*, 26(6):36–47, 2006.

[16] P. Lamata, a. Jalote-Parmar, F. Lamata, and J. Declerck. The Resection Map, a proposal for intraoperative hepatectomy guidance. *International Journal of Computer Assisted Radiology and Surgery*, 3(3-4):299–306, jun 2008. ISSN 1861-6410. doi: 10.1007/s11548-008-0226-5.

[17] Christian Hansen, Stephan Zidowitz, Felix Ritter, Christoph Lange, Karl Oldhafer, and Horst K Hahn. Risk maps for liver surgery. *International journal of computer assisted radiology and surgery*, 8(3):419–28, may 2013. ISSN 1861-6429. doi: 10.1007/s11548-012-0790-6.

[18] Steffen Oeltze and Bernhard Preim. Visualization of vasculature with convolution surfaces: method, validation and evaluation. *IEEE transactions on medical imaging*, 24(4):540–8, apr 2005. ISSN 0278-0062.

[19] C Schumann, S Oeltze, R Bade, B Preim, and HO Peitgen. Model-free Surface Visualization of Vascular Trees. In *EuroVis*, pages 283–290, 2007.

[20] Bernhard Preim and Steffen Oeltze. *3D visualization of vasculature: an overview*. Springer, 2008.

[21] Jianhuang Wu, Renhui Ma, Xin Ma, Fucang Jia, and Qingmao Hu. Curvature-dependent surface visualization of vascular structures. *Computerized medical imaging and graphics : the official journal of the Computerized Medical Imaging Society*, 34(8):651–8, dec 2010. ISSN 1879-0771. doi: 10.1016/j.compmedimag.2010.07.006.

[22] Qingqi Hong, Qingde Li, and Jie Tian. Implicit reconstruction of vasculatures using bivariate piecewise algebraic splines. *IEEE transactions on medical imaging*, 31(3):543–53, mar 2012. ISSN 1558-254X. doi: 10.1109/TMI.2011.2172455.

[23] Jianhuang Wu, Qingmao Hu, and Xin Ma. Comparative study of surface modeling methods for vascular structures. *Computerized medical imaging and graphics : the official journal of the Computerized Medical Imaging Society*, 37 (1):4–14, jan 2013. ISSN 1879-0771. doi: 10.1016/j.compmedimag.2013.01.002.

[24] WE Lorensen and HE Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, 21(4):163–169, 1987.

[25] Timothy S. Newman and Hong Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5):854–879, oct 2006. ISSN 00978493. doi: 10.1016/j.cag.2006.07.021.

[26] Taubin Gabriel, Tong Zhang, and Gene Golub. Optimal Surface Smoothing as Filter Design. In *Computer Vision — ECCV '96*, pages 283–292. Springer Berlin Heidelberg, 1996.

[27] Ragnar Bade, Jens Haase, and Bernhard Preim. Comparison of Fundamental Mesh Smoothing Algorithms for Medical Surface Models. *SimVis*, 1(c):1–16, 2006. doi: 10.1.1.60.895.

[28] T Moench, S Adler, and B Preim. Staircase-aware smoothing of medical surface meshes. *EG VCBM'10 Proceedings of the 2nd Eurographics conference on Visual Computing for Biology and Medicine*, 2010.

[29] WJ Schroeder, JA Zarge, and WE Lorensen. Decimation of triangle meshes. *ACM SIGGRAPH 19th annual conference on Computer graphics and interactive techniques*, 26(2):65–70, jul 1992. ISSN 00978930. doi: 10.1145/142920.134010.

[30] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. In *Eurographics Symposium on Geometry Processing 2006*, pages 61–70, 2006.

[31] Shahram Izadi, Andrew Davison, Andrew Fitzgibbon, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Dustin Freeman. Kinect Fusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, page 559, 2011. ISSN 9781450307161. doi: 10.1145/2047196.2047270.

[32] Remote Sensing, Spatial Information Sciences, Csaba Benedek, and Distributed Events. Towards 4D Virtual City Reconstruction From Lidar Point Cloud. In *ISPRS*, volume II, pages 15–20, 2013.

[33] Martin Habbecke and Leif Kobbelt. A surface-growing approach to multi-view stereo reconstruction. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007. ISSN 10636919. doi: 10.1109/CVPR.2007.383195.

[34] Valentin Leonardi, V Vidal, JL Mari, and Marc Daniel. 3D reconstruction from CT-scan volume dataset application to kidney modeling. In *Proceedings of the 27th Spring Conference on Computer Graphics*, volume D, pages 111–120, 2011. ISBN 9781450319782.

[35] Jianhuang Wu, Mingqiang Wei, Yonghong Li, Xin Ma, Fucang Jia, and Qingmao Hu. Scale-adaptive surface modeling of vascular structures. *Biomedical engineering online*, 9(1):75, 2010. ISSN 1475-925X. doi: 10.1186/1475-925X-9-75.

[36] Matthew Bolitho, Michael Kazhdan, Randal Burns, and Hugues Hoppe. Parallel poisson surface reconstruction. In *Advances in Visual Computing*, pages 678–689. Springer, 2009.

[37] S. Pieper, M. Halle, and R. Kikinis. 3D Slicer. In *2004 2nd IEEE International Symposium on Biomedical Imaging: Nano to Macro (IEEE Cat No. 04EX821)*, pages 632–635, 2004. ISBN 0-7803-8388-5. doi: 10.1109/ISBI.2004.1398617.

[38] David Doria and Arnaud Gelas. Poisson Surface Reconstruction for VTK. *The VTK Journal*, Januar:5, 2010.

[39] AAsmund Avdem Fretland, Airazat M Kazaryan, Bjorn Atle Bjornbeth, Kjersti Flatmark, Marit Helen Andersen, Tor Inge Tonnessen, Gudrun Maria Waaler Bjornelv, Morten Wang Fagerland, Ronny Kristiansen, Karl Oyri, and Bjorn Edwin. Open versus laparoscopic liver resection for colorectal liver metastases (the Oslo-CoMet study): study protocol for a randomized controlled trial. *Trials*, 16(1):1–10, 2015. ISSN 1745-6215. doi: 10.1186/s13063-015-0577-5.

[40] Paul a. Yushkevich, Joseph Piven, Heather Cody Hazlett, Rachel Gimpel Smith, Sean Ho, James C. Gee, and Guido Gerig. User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability. *NeuroImage*, 31(3):1116–1128, 2006. ISSN 10538119. doi: 10.1016/j.neuroimage.2006.01.015.

[41] Szymon Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *Proceedings - 2nd International Symposium on 3D Data Processing, Visualization, and Transmission. 3DPVT 2004*, pages 486–493, 2004. ISBN 0769522238. doi: 10.1109/TDPVT.2004.1335277.

[42] Christian Schumann, Mathias Neugebauer, Ragnar Bade, Bernhard Preim, and Heinz Otto Peitgen. Implicit vessel surface reconstruction for visualization and CFD simulation. *International Journal of Computer Assisted Radiology and Surgery*, 2(5):275–286, 2008. ISSN 18616410. doi: 10.1007/s11548-007-0137-x.

[43] Xiaoyan Feng, Anna Morandi, Martin Boehne, Tawan Imvised, Benno M. Ure, Joachim F. Kuebler, and Martin Lacher. 3-Dimensional (3D) laparoscopy improves operating time in small spaces without impact on hemodynamics and psychomental stress parameters of the surgeon. *Surgical Endoscopy*, 29 (5):1231–1239, 2015. ISSN 0930-2794. doi: 10.1007/s00464-015-4083-3.

[44] Bernard CK Choi, Anita WP Pak, and For Cdc. A Catalog of Biases in Questionnaires. *Prev Chronic Dis*, 2(1):1–13, 2005.

[45] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Transactions on Graphics*, 22(3):463, jul 2003. ISSN 07300301. doi: 10.1145/882262.882293.

[46] Tobias Moench. Generation of smooth and accurate surface models for surgical planning and simulation. In *SPIE Medical Imaging 2010: Visualization, Image-Guided Procedures, and Modeling*, pages 76252G–76252G, 2010.

[47] Miriah Meyer, Robert M. Kirby, and Ross Whitaker. Topology, accuracy, and quality of isosurface meshes using dynamic particles. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1704–1711, 2007. ISSN 10772626. doi: 10.1109/TVCG.2007.70604.

[48] Miriah Meyer, Ross Whitaker, Robert M. Kirby, Christian Ledergerber, and Hanspeter Pfister. Particle-based sampling and meshing of surfaces in multimaterial volumes. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1539–1546, 2008. ISSN 10772626. doi: 10.1109/TVCG.2008.154.

[49] Christopher Dyken, Kjetil Olsen Lye, Johan Seland, Erik W. Bjonnes, Jon Hjelmervik, Jens Olav Nygaard, and Trond Runar Hagen. A framework for OpenGL client-server rendering. In *CloudCom 2012 - Proceedings: 2012 4th IEEE International Conference on Cloud Computing Technology and Science*, pages 729–734, 2012. ISBN 9781467345095. doi: 10.1109/CloudCom.2012.6427506.

[50] Kun Zhou, Minmin Gong, Xin Huang, and Baining Guo. Data-parallel octrees for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 17(5):669–681, 2011. ISSN 10772626. doi: 10.1109/TVCG.2010.75.

[51] Apollon Zygomalas, Dionissios Karavias, Dimitrios Koutsouris, Ioannis Maroulis, Dimitrios D. Karavias, Konstantinos Giokas, and Vasileios Megalooikonomou. Computer-assisted liver tumor surgery using a novel semiautomatic and a hybrid semiautomatic segmentation algorithm. *Medical and Biological Engineering and Computing*, 2015. ISSN 17410444. doi: 10.1007/s11517-015-1369-5.

[52] Luc Soler and Jacques Marescaux. Patient-specific surgical simulation. *World Journal of Surgery*, 32(2):208–212, 2008. ISSN 03642313. doi: 10.1007/s00268-007-9329-3.

[53] Stéphane Nicolau, Luc Soler, Didier Mutter, and Jacques Marescaux. Augmented reality in laparoscopic surgical oncology. *Surgical Oncology*, 20(3): 189–201, 2011. ISSN 09607404. doi: 10.1016/j.suronc.2011.07.002.

[54] Shunsuke Ohshima. Volume analyzer SYNAPSE VINCENT for liver analysis. *Journal of Hepato-Biliary-Pancreatic Sciences*, 21(4):235–238, 2014. ISSN 18686982. doi: 10.1002/jhbp.81.

[55] Dimitrios Ntourakis, Ricardo Memeo, Luc Soler, Jacques Marescaux, Didier Mutter, and Patrick Pessaux. Augmented Reality Guidance for the Resection of Missing Colorectal Liver Metastases: An Initial Experience. *World journal of surgery*, 40(2):419–426, 2015. ISSN 1432-2323 (Electronic). doi: 10.1007/s00268-015-3229-8.

[56] Francesco Volonté, François Pugin, Pascal Bucher, Maki Sugimoto, Osman Ratib, and Philippe Morel. Augmented reality and image overlay navigation with OsiriX in laparoscopic and robotic surgery: Not only a matter of fashion. *Journal of Hepato-Biliary-Pancreatic Sciences*, 18(4):506–509, 2011. ISSN 18686974. doi: 10.1007/s00534-011-0385-6.

CHAPTER 7

PAPER II: A NOVEL METHOD FOR PLANNING LIVER RESECTIONS USING DEFORMABLE BÉZIER SURFACES AND DISTANCE MAPS

Rafael Palomar · Faouzi A. Cheikh · Bjørn Edwin · Åsmund A. Fretland · Azeddine Beghdadhi · Ole J. Elle

Abstract *Background and Objective:* For more than a decade, computer-assisted surgical systems have been helping surgeons to plan liver resections. The most widespread strategies to plan liver resections are: drawing traces in individual 2D slices, and using a 3D deformable plane. In this work, we propose a novel method which requires low level of user interaction while keeping high flexibility to specify resections. *Methods:* Our method is based on the use of Bézier surfaces, which can be deformed using a grid of control points, and distance maps as a base to compute and visualize resection margins (indicators of safety) in real-time. Projection of resections in 2D slices, as well as computation of resection volume statistics are also detailed. *Results:* The method was evaluated and compared with *stated-of-the-art* methods by a group of surgeons ($n = 5$, 5-31 years of experience). Our results show that the the proposed method presents planning times as low as *state-of-the-art* methods ($174\ s$ median time) with high reproducibility of results in terms of resected volume. In addition, our method not only leads to smooth virtual resections easier to perform surgically compared to other *state-of-the-art* methods, but also shows superior preservation of resection margins. *Conclusions:* Our

method provides clinicians with a robust and easy-to-use method for planning liver resections with high reproducibility, smoothness of resection and preservation of resection margin. Our results indicate the ability of the method to represent any type of resection and being integrated in in real clinical work-flows.

7.1 Introduction

Liver cancer is one of the most common causes of cancer death worldwide and its frequency is increasing in some geographical areas of historically low incidence rates [1]. Liver resection, which refers to the surgical removal of a liver tumor, is the only curative treatment for liver cancer. Planning of liver resections is usually based on the anatomic division of the liver in segments, as described in Couinaud [2]. The Couinaud division, which presents a wide consensus in the medical community, separates the liver into 8 areas (segments) according to the blood supply, and establishes a framework for the classification of resections in different types [3].

Liver cancer is either primary (arising from normal liver tissue) or secondary (spreading to the liver from cancer located in other organs). For hepatocellular carcinoma (primary), which accounts for 70%-80% of the liver cancer cases worldwide [4], surgical resection is the treatment of choice and is considered to be potentially curative [5]. Selected patients with metastatic (secondary) liver tumors—which develop in 50% of the cases of colorectal cancer—present up to 58% increased 5-year survival rates after liver resection [6].

In contemporary liver surgery, pre-operative planning becomes increasingly important. New techniques like parenchymal-sparing [7] can use pre-operative planning to help surgeons optimizing the resection path, potentially increasing the remnant liver. Volume expanding techniques (like associating liver partition and portal split (ALPPS) [8], and portal vein embolization) can also make use of pre-operative planning to derive the volumetry of the resection. This can help ensuring that remnant liver is large enough and with sufficient function to prevent post-operative liver failure.

For nearly two decades, surgeons and other clinicians have employed computer-assisted surgical systems to support the decision-making process for planning and guiding surgical interventions. In the case of liver resections, these systems have recently been evaluated in the clinical practice and have shown improvements not only in tumor localization and precision of surgery planning [9, 10, 11], but also an improved orientation and confidence of the surgeon during the operation [12].

Liver resection planning systems are based on the definition of *virtual resections* [13]. Virtual resections help clinicians to visualize the resection (surgical cutting

path), affected vessels and resection margins (safety distance kept between the tumors and the resection path). In addition, virtual resections allow the computation of the estimated resected volume.

Simplicity of use and flexibility to specify the virtual resections are key features of surgery planning systems. Simplicity and flexibility are often considered as diverging objectives—simple interactions usually impose constraints on the freedom to describe virtual resections. The two most common strategies proposed for specification of virtual resections are: drawing traces in 2D individual slices [14] (**DS**) and definition of and virtual resections defined using a deformable cutting plane [15] (**CP**).

7.1.1 Contribution

In this work, we present a new method for planning liver resection procedures. The novelty of our method is twofold. On one hand, our method is based on the use of Bézier surfaces, which can be deformed in real-time solely by a set of control points. On the other hand, we propose the use of distance maps to project the safety margins in real-time onto the resection surface, thus allowing the user to modify the resection proposal until the safety requirements are met.

In addition, an implementation of the method based on the open-source software *3D Slicer* [16] is presented. This implementation includes interaction mechanisms which not only avoid the use of manual drawing of lines (both in the 3D model as in CP, and in the 2D slices as in DS), but also presents a flexible yet simple way to define virtual resections regardless of their type (e.g., hemihepatectomy, parenchymal-sparing). Details on visualization aspects, projection of resection surfaces onto individual 2D slices, as well as resected volume computation based on our method, are also detailed.

7.2 Theoretical Background

In this section, we briefly describe the foundations of Bézier tensor product surfaces and some of their most important properties. A deeper description can be found in [17, 18, 19]. For simplicity and clarity reasons, in this work we focus on parametric non-rational Bézier surfaces, however, the methods described in this work can be easily adapted to other Bézier formulations.

Formally, a parametric non-rational Bézier surface $\mathbf{S} \in \mathbb{R}^3$ of degree (m, n) can be defined as:

$$\mathbf{S}(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{C}_{i,j} B_{i,m}(u) B_{j,n}(v), \tag{7.1}$$

with $u, v \in [0, 1]$. $\mathbf{C}_{i,j} \in \mathbb{R}^3$ are the control points characterizing the shape of the surface, and the i-th and j-th bases, $B_{i,m}$ and $B_{j,n}$ with degrees m and n respectively, are Bernstein polynomials given by:

$$B_{i,m}(t) = \binom{m}{i}(1-t)^{(m-i)}t^i. \tag{7.2}$$

Lemma 2 *Let* $\mathbf{S}$ *be a parametric bi-linear Bézier surface of degree* (m, n) *as described in Eq. (7.1). Such surface, has the following properties:*

(a) Surface contained in the convex hull $\mathbb{CH}$*:*

$$\mathbf{S}(u, v) \in \mathbb{CH}(\mathbf{C}_{0,0}, ..., \mathbf{C}_{m,n}) \; \forall (u, v). \tag{7.3}$$

(b) Affine transformation invariance:

$$\mathbf{T}(\mathbf{S}) = \sum_{i=0}^{m}\sum_{j=0}^{m} \mathbf{T}(\mathbf{C}_{i,j})B_{i,p}(u)B_{j,q}(v), \tag{7.4}$$

where T *is an affine transformation (i.e., rotation, reflection, translation or scaling).*

(c) Polyhedral approximation: under triangulation, the net of control points forms a planar polyhedral approximation of the surface.

A proof for these properties follows easily from the proof of the analogous properties for Bézier curves [17]. In the remaining of the document, we will refer to these properties to justify design aspects and properties of the proposed method.

7.3 Materials and Methods

7.3.1 Overview of the Method

Regardless of the type of model supporting the definition of virtual resections (i.e., voxel-based or 3D models), models employed in planning of liver resection ultimately rely on patient-specific segmented models typically obtained from computed tomography (**CT**).

The approach presented in this work is entirely supported by patient-specific 3D models. In order to construct these models, first, a medical image is obtained from CT. Medical images are represented as scalar fields $F : \mathbb{R}^3 \to \mathbb{R}$ where the points

$\{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^N$ present intensity values $F(\mathbf{p}_i) = v$. Through segmentation, different tissues (i.e., vessels, parenchyma[1] and tumors) are separated in a new scalar field $S : \mathbb{R}^3 \rightarrow \{l_1, ..., l_k\}$ with k classes (tissues). Finally, through isosurface extraction methods, like marching cubes [20, 21], 3D models of the labeled tissues are obtained.

In computer graphics, surface descriptions such as 3D models and Bézier surfaces are commonly represented as triangle meshes. In the remaining of this work, triangular meshes are denoted as sets $\mathcal{M} = \{V, T\}$ with vertices $V = \{0, 1, 2, ...\}$ and its associated positions $\mathbf{p}_i \in \mathbb{R}^3$, edges $E = \{(i, j)|i, j \in V\}$, and triangles $T = \{(i, j, k)|(i, j), (j, k), (k, i) \in E\}$.

As in CP approaches, the work-flow of our approach (Figure 7.1), consists of two steps: initialization (planar approximation) and modification of the resection. The differences with other CP approaches like [15] lie in the underlying representation, and deformation methods. This not only leads to new properties of the resection surfaces, but also to different user interaction schemes. In our method, first, the 3D mesh models $\mathcal{M}_p$ (parenchyma) and $\mathcal{M}_t$ (tumor) generated previously, are used to define a planar contour around the parenchyma. Unlike in CP, user interaction required to specify the contour is not based on manual drawing, but on a slicing movable plane (Section 7.3.2). This contour leads to the generation of a planar resection surface. In a second step, the user can deform the planar surface by means of a grid of control points (Section 7.3.3).

7.3.2 Initialization of the Resection

The goal of this process is to obtain a first approximation (planar Bézier) of the resection surface which will be used as a starting point for subsequent modifications. In order to obtain this approximation, the user is first provided with the 3D representation of the liver and tumor, as illustrated in Figure 7.2a and 7.2b. Together with these anatomic representations, a line connecting the centroid of the tumor $\mathbf{c}$ with an arbitrarily placed 3D end-point $\mathbf{p}_e$ (in Figure 7.2a,7.2b this corresponds to $\mathbf{p}_{end0}$ and $\mathbf{p}_{end1}$ in different interaction times) is displayed. This line is associated to an invisible plane P (in Figure 7.2a this corresponds to P_0 and P_1 at different interaction times) passing through the middle point of the line connecting $\mathbf{c}$ and a end-point $\mathbf{p}_e$ which satisfies the point-normal form:

$$\underbrace{(\mathbf{p}_e - \mathbf{c})}_{\vec{\mathbf{n}}} \cdot \left(\mathbf{x} - \left(\frac{\mathbf{c} + \mathbf{p}_e}{2} \right) \right) = 0. \tag{7.5}$$

[1] In this work, the term parenchyma is used to refer to the part of the liver which is neither tumor tissue nor blood vessels.

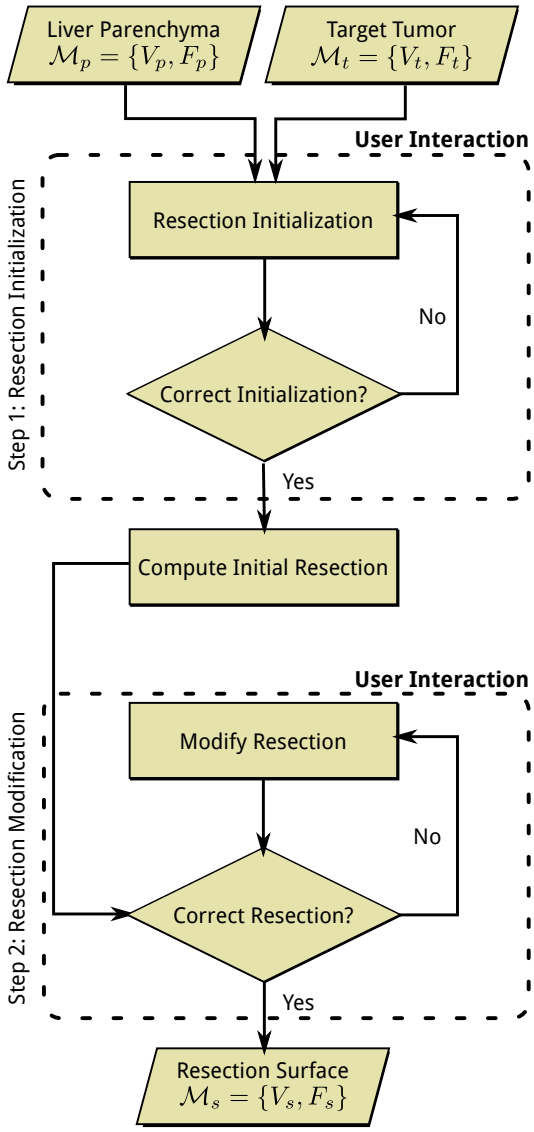


Figure 7.1: Flow chart of the proposed method

The plane P is then used to slice the parenchyma model $\mathcal{M}_p$, thus providing a contour representation V_s (ring around the parenchyma in Figure 7.2b). User interaction takes place by moving the 3D end-point $\mathbf{p}_e$. The effect of moving this end-point is the modification of the slicing plane, which effectively creates a contour (around the parenchyma) moving in real-time. This initialization process is formally described in Algorithm 1.

Algorithm 1 Computation of resection contour

Precondition: User-defined end point $\mathbf{p}_e$ inside parenchyma mesh $\mathcal{M}_p$.

1: **function** CONTOUR($\mathcal{M}_p$, $\mathcal{M}_t$, $\mathbf{p}_e$)
2: $\quad\mathbf{c} \leftarrow Centroid(\mathcal{M}_t)$ ▷ *Centroid of tumor*
3: $\quad\vec{\mathbf{n}} \leftarrow \mathbf{p}_e - \mathbf{c}$ ▷ *Normal vector*
4: $\quad P \leftarrow Plane(\frac{\mathbf{p}_e+\mathbf{c}}{2}, \vec{\mathbf{n}})$ ▷ *Slicing plane* $P \perp \vec{\mathbf{n}}$
5: $\quad V_s \leftarrow Slice(\mathcal{M}_p, P)$ ▷ *Point-based contour*
6: $\quad$**return** V_s
7: **end function**

The resulting contour V_s is then used to compute resection approximation in terms of a planar surface (Figure 7.2c). Similarly to [15], the origin, extent and orientation of this plane is obtained by means of principal component analysis (**PCA**). The orientation of the initial resection is given by the two eigenvectors $\vec{\mathbf{E}}_1$ and $\vec{\mathbf{E}}_2$ presenting the larger eigenvalues e_1 and e_2. These eigenvalues, are then used to compute the size of the initial resection, in our case:

$$\begin{cases} l_1 = 4\sqrt{e_1} \\ l_2 = 4\sqrt{e_2} \end{cases} . \tag{7.6}$$

The election of the lengths l_i is based on the consideration of $\sqrt{e_i}$ as estimators of the standard deviations of the contour V_s along the eigenvectors $\vec{\mathbf{E}}_i$. Assuming uniform distribution of the contour along these eigenvectors, l_i exceed the length of the contour, and therefore, the initial plane also exceeds the boundaries of the parenchyma.

The origin of the plane (center) is computed with respect to the centroid of the contour V_s. First the centroid is computed using all the points that make up the contour:

$$\mathbf{c} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i \ , \ \mathbf{p}_i \in V_s. \tag{7.7}$$

Then the origin is computed as the translation of the centroid in the direction $(-\vec{\mathbf{E}}_1, -\vec{\mathbf{E}}_2)$ by half the extent of the plane on each direction, this is:

$$\mathbf{o} = \mathbf{c} - \frac{l_1}{2}\vec{\mathbf{E}}_1 - \frac{l_2}{2}\vec{\mathbf{E}}_2. \tag{7.8}$$

Once the geometry of the initial resection is computed, we map a 2D grid of $m \times n$ equally spaced points. This grid of points will serve as a base to build a deformable Bézier surface—from Lemma 1.a it follows that, if all control points lie in a plane, the associated Bézier surface also lies on the same plane. Formally, this process is described in Algorithm 2.

Algorithm 2 Compute initial resection

Precondition: Cross-section contour represented as the set of N 3D points $V_s = \{\mathbf{p}_i\}_{i=1}^{N}$. m and n determine the dimensions of the output control polygon.

function InitialResection(V_s, m, n)
  $\mathbf{c}_c \leftarrow Centroid(V_s)$ ▷ *Centroid of contour*
  $[e_1, e_2, \vec{\mathbf{E}}_1, \vec{\mathbf{E}}_2] \leftarrow PCA(V_s)$
  $l_1 \leftarrow 4\sqrt{e_1}$ ▷ *Width of resection plane*
  $l_2 \leftarrow 4\sqrt{e_2}$ ▷ *Height of resection plane*
  $\mathbf{o} \leftarrow \mathbf{c}_c - \frac{l_1}{2}\vec{\mathbf{E}}_1 - \frac{l_2}{2}\vec{\mathbf{E}}_2$ ▷ *Plane origin*
  for $i \leftarrow 1$ to m **do**
    for $j \leftarrow 1$ to n **do**
      $\mathbf{C}_{i,j} \leftarrow \mathbf{o} + \frac{il_1}{m}\vec{\mathbf{E}}_1 + \frac{jl_2}{n}\vec{\mathbf{E}}_2$
    end for
  end for
  $C \leftarrow \{\mathbf{C}_{i,j}\}_{i,j=1}^{m,n}$ ▷ *Control polygon*
  return C
end function

7.3.3 Deformation of Bézier Surfaces

Deformation of a Bézier tensor-product surface is performed through the interactive manipulation of the coordinates of the control points (distributed in a connected grid). The control points do not normally lie on the surface (except for the corners, which always lie in the surface). The fact that the net of control points is an approximation of the surface (Lemma 1.c) makes that the deformations of

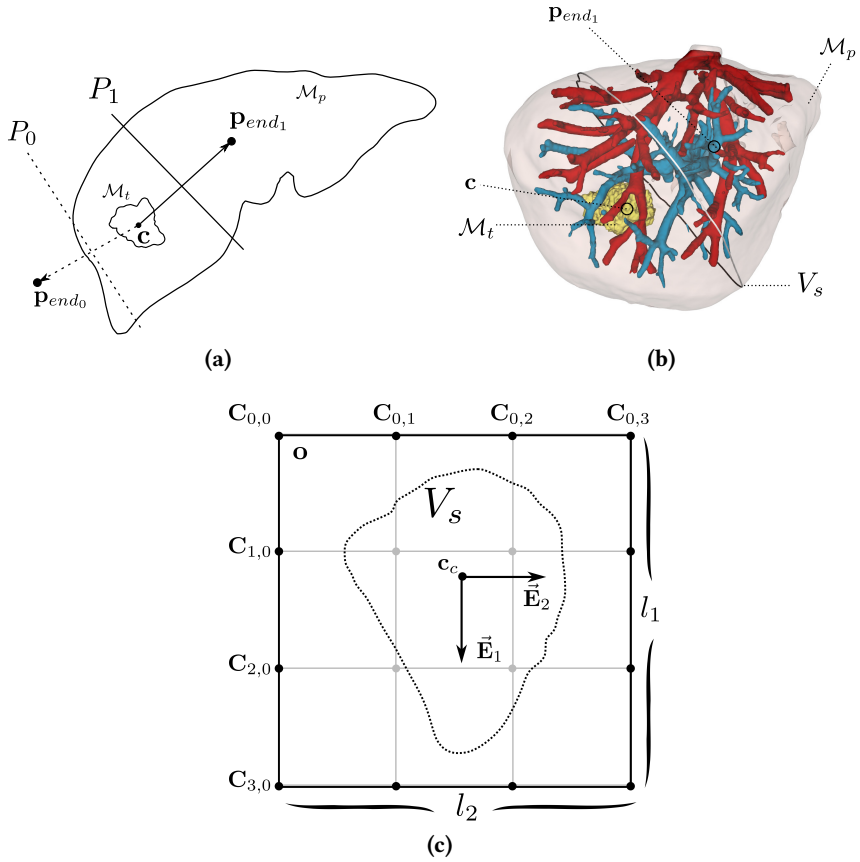


Figure 7.2: Initialization of the resection: (a) 2D illustration of the initialization process where the initial point $\mathbf{p}_{end_0}$ (which produces the initial plane P_0), is moved to $\mathbf{p}_{end_1}$, thus producing the initial plane P_1; (b) 3D representation initial resection resulting at $\mathbf{p}_{end_1}$; (c) Geometry of the initial resection G based on PCA of the contour V_s.

the surface occur in coherence with the manipulation of the control points.

The number of control points is an important design consideration. On one hand, increasing the number of control points increases the number of interactions as the user may have to modify more control points. On the other hand, and as derived from Eq. (7.1) and Eq. (7.2), the number of control points determines the degree of the surface, and hence, its representation flexibility. In this work, surfaces defined by 4×4 control points are employed.

The surface resolution has an impact on the performance of computing Bézier surfaces. For our method, this is a very important consideration since the computation of Bézier surfaces is followed by other processing stages (Section 7.3.4) and all the computations involved must be performed in real-time. In this work, surfaces of resolution 40×40 points are used.

Updating the resection surface when a control point changes its position requires re-computing the whole extent of the surface—the reader should notice that this is an inherent property of the formulation (Eq. 7.1). Algorithm 3 describes this process for surfaces of variable number of control points and resolution.

Algorithm 3 Update Bézier Surface

Precondition: $\mathbf{C}$ being the grid of control points of size $m \times n$, and $r_u \times r_v$ representing the resolution of the surface.

function UpdateBezier($m, n, r_u, r_v, \mathbf{C}$)
  for $i \leftarrow 1$ to r_u **do**
    $u \leftarrow i/(r_u - 1)$
    for $j \leftarrow 1$ to r_v **do**
      $v \leftarrow j/(r_v - 1)$
      for $k \leftarrow 1$ to m **do**
        $B_u \leftarrow \binom{m}{k} u^k (1-u)^{m-k}$
        for $l \leftarrow 1$ to n **do**
          $B_v \leftarrow \binom{n}{l} v^l (1-v)^{n-l}$
          $\mathbf{S_{i,j}} \leftarrow \mathbf{C}_{i,j} B_u B_v$
        end for
      end for
    end for
  end for
  return S
end function

7.3.4 3D Visualization and Projection in 2D Slices

Together with the visualization of the 3D surface defining the virtual resection, our approach includes the visualization of the resection margin—which refers to the safety distance that should be kept between the resection surface and the tumors. The resection margin is updated when the resection is modified. In order to compute the resection margin, we employ the *point-to-surface* distance δ:

$$\delta(\mathbf{p}) = \min_{\forall \mathbf{q}_i \in V_t} \|\mathbf{p} - \mathbf{q}_i\| \,, \tag{7.9}$$

where V_t is the set of points of the tumor model $\mathcal{M}_t$ and $\mathbf{p}$ is a point belonging to the resection surface model $\mathcal{M}_r$; $||.||$ refers to the euclidean norm. The *point-to-surface* distance is computed for all points of the resection surface which effectively generates a distance map projected onto the resection surface (Figure 7.3b). Using these distance maps, it is possible to determine the validity of the resection surface in terms of resection margin; for instance, if the margin set by clinicians is under $10\ mm$, then the resection would be valid only if all the points in the surface are further than $10\ mm$ from the tumor.

For visualization purposes, we avoid the use of a color-map projected onto the surface. Visually, the color-map contains more information than clinicians need and all this information can be distracting. Instead, we threshold the distance map according to the resection margin. The areas violating the resection margin are then highlighted in yellow (with blue contour) while the rest of the surface remains in gray (Figure 7.3b, 7.3c). The part of the Bézier surface exceeding the liver surface can be hidden as well as the net of control points (Figure 7.3c). This facilitates the visualization of the resection by avoiding occlusions and simplifying the scene.

The projection of the surface onto individual 2D slices (Figure 7.3d, 7.3e, 7.3f) is obtained by the intersection of axial, coronal and sagittal planes with the 3D Bézier surface.

7.3.5 Computation of Resected Volume

Computation of resection volumetry is a key functionality provided by existing software solutions for planning liver resections. Our approach to compute the resected volume consists of three steps. First, a high-resolution Bézier surface ($r_u = 300, r_v = 300$) is generated. Secondly, all the points of this high-resolution surface are mapped into a segmented image $M : \mathbb{R}^3 \rightarrow \{l_b = 0, l_p = 1, l_t = 2, l_r = 3\}$ (same dimensions and spacing as the original image taken from the patient for diagnosis), where the background (l_b), liver parenchyma (l_p), target

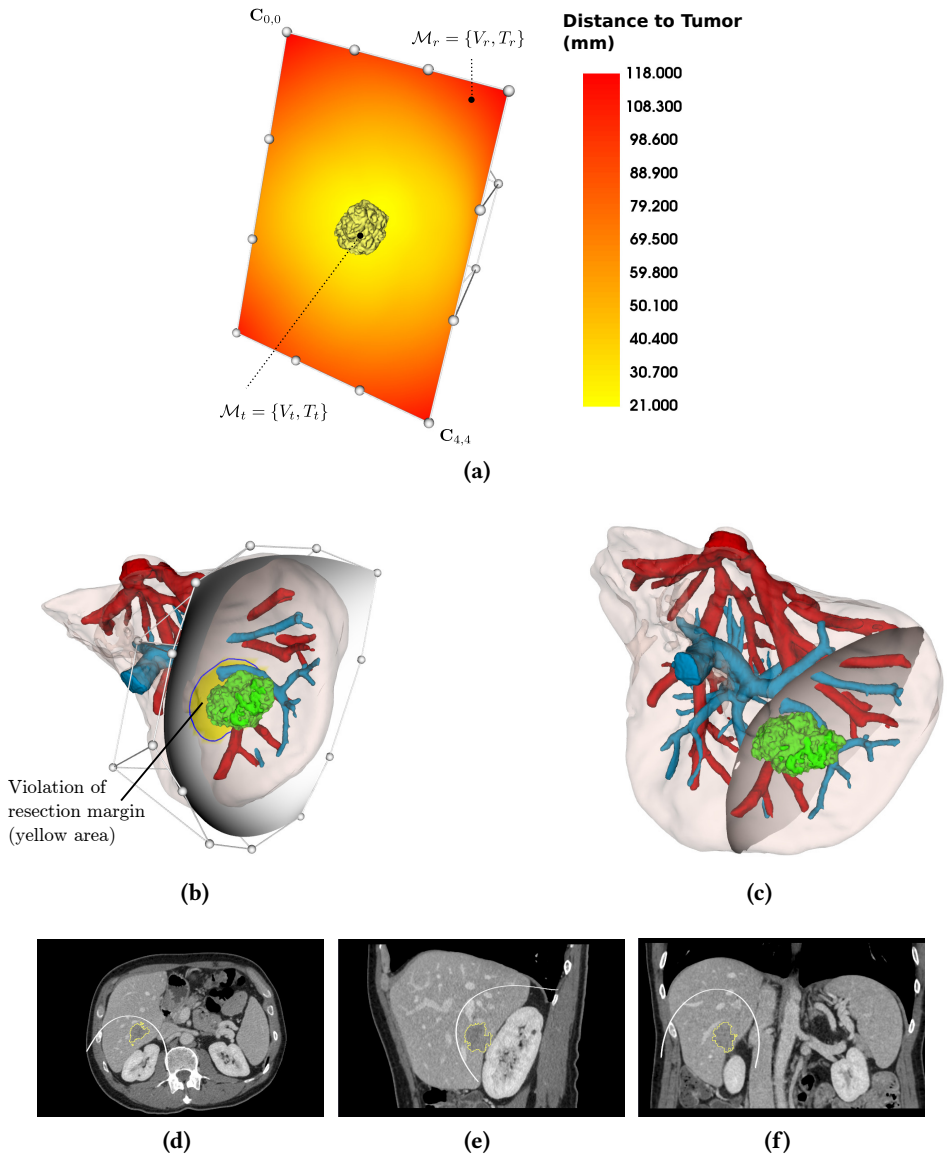


Figure 7.3: Visualization of the resection path: (a) distance map derived from the tumor model $\mathcal{M}_t$ and the resection surface $\mathcal{M}_r$. (b) visualization of the resection surface given by a 3D Bézier surface and thresholding of the distance map using the resection margin; the violation of resection margin is highlighted in yellow (blue contour around); (c) visualization of the final resection surface where the control points and the resection exceeding the parenchyma are hidden; (d,e,f) projection of the resection surface into individual 2D slices with axial, coronal, sagittal orientation respectively.

tumors (l_t) and resection surface (l_r) are separated by different label values. The mapping of the high-resolution surface is performed on a basis of a $3 \times 3 \times 3$ voxels per surface point, which effect is the extrusion (thickening) of the mapped resection surface. This, together with the high-resolution construction of the surface, guarantees both continuity of the mapped surface and a clear boundary between the resected and the remnant volumes of the liver. Finally, a connected threshold region growing is applied (low threshold $l = l_p$ and upper threshold $u = l_r$) with a seed point arbitrarily chosen from a target tumor.

In order to compute volumes using this process, the resection path must enter and leave the parenchyma completely. This not only makes sense under the point of view of the application, but also guarantees a separation between the resected and the remnant volume.

7.3.6 User Interaction

In order to keep the simplicity of use and flexibility of resection representation, we introduce two new interaction mechanisms: global translation of the resection surface and modification of control points in groups. An example of the possible sequence of interactions using these mechanisms, and leading to a valid resection plan is illustrated in Figure 7.4.

Global translation of Bézier surfaces defined by a grid of 4×4 control points requires the modification of all the 16 control points—which implies a considerable number of user interactions. To avoid this, we set the control polygon connecting the control points as a interactive frame that can be moved through *drag-and-drop* interactions. Moving the frame produces a translation transformation on all control points which effectively produces the translation of the surface (Lemma 1.b).

Resections, regardless of their type, can be defined by a virtual resection resulting from a resection surface with *pseudo-parabolic* shape. For a resection surface defined by a grid of 4×4 control points, this implies the movement of either the 4 inner points of the grid or the 12 remaining (outer) points. In our implementation, simple mouse *right-click* on any of the 4 inner points will produce translation of all these points together. The same applies for the 12 outer points. This type of interaction allows the simple construction of *pseudo-parabolic* shapes, which can then be refined by individual modification of the control points. For illustration purposes we refer to Figure 7.2c, where the 4 inner points are shown in light-gray, while the 12 outer are shown in black.

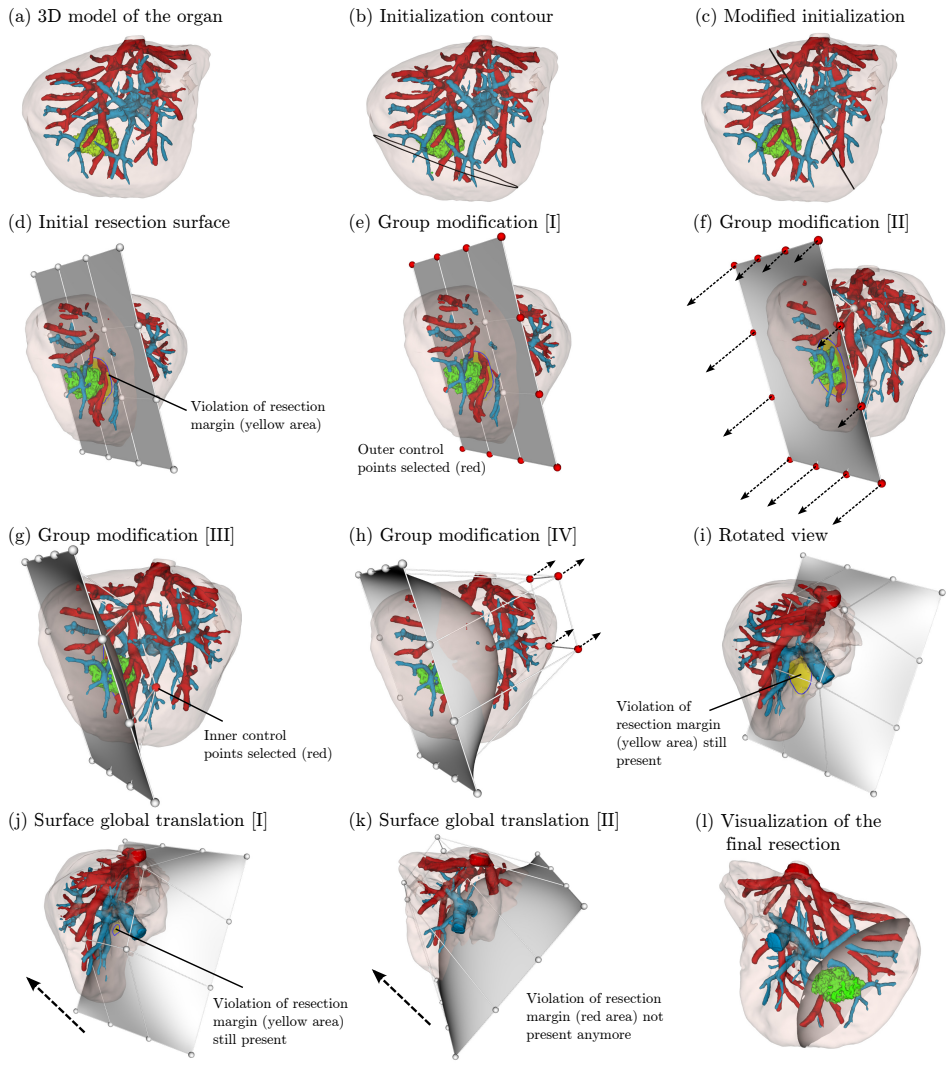


Figure 7.4: Instance of liver resection planning using the proposed method including the proposed user interaction techniques. The sequence of interactions (a) to (d) illustrates the process of obtaining the initial approximation of the resection surface. (e) and (f) show the modification of the group of outer control points. In a similar way, (g) and (h) show the modification of the inner group of control points. In (i) a rotation of the view is performed. Later, in (j) and (k) the surface is translated (globally). Finally in (l) the visualization of the final resection is presented. The reader should notice, that from (d) to (j) the resection presents a yellow (blue contour around) area indicating the violation of the resection margin (arbitrarily set at 20mm). In (k) the resection margin is preserved (no yellow area in the surface) indicating validity of resection in terms of safety margin.

Table 7.1: Implementation aspects for DS, CP and Bézier.

Aspect	DS	CP	Bézier
Underlying representation	Bi-quadratic polynomial	Discrete grid	Bi-cubic polynomial
Surface resolution	20×20	$30 \times n$	40×40
Visualization	2D Slices / 3D Models	3D Models	3D Models
Resection margin visualization	✗	✗	✓
Interaction	1. Drawing in slices (3-5) traces	1. Traces on parenchyma 2. Local deformation	1. Slicing plane on parenchyma 2. Bézier deformation

7.4 Evaluation Methodology

In order to validate and evaluate the proposed method we perform a user study which includes a comparison with our own implementation of CP and DS in 3D Slicer [16]. The implementation details of CP and DS are described in the following and summarized in Table 7.1.

The implementation of the CP approach is based on [13] and [15]. This approach uses a surface with a variable mesh resolution $30 \times n$ square quads, where 30 is the number of quads in the short axis and n is the number of square quads needed to fill the extent of the plane (Eq. (7.6)) in the long axis.

Implementation of DS is based on the general principles established in [13] combined with design aspects in [14]. In this implementation, the user can draw and overwrite complete traces individually over the set of 2D slices. Navigation between traces was implemented so the user could easily find individual traces and their corresponding slices. Parametric linear interpolation was applied to individual traces to obtain a regularly spaced sampled traces (20 points per trace). The final surface was computed by means of parametric quadratic interpolation between the traces, which requires at least 3 traces. Modification of the surface was allowed on the basis of traces, this is, redrawing of one or more traces and fast re-computation of the interpolated surface.

Study design and quantitative analysis are performed according to [22], which provides a comprehensive guide for the design and data analysis of experiments similar to the one presented in this work. In order to compare the different methods we establish the criteria and their corresponding objective evaluation metrics described in the following.

Preservation of resection margin This criteria is concerned with how accurately the resection margin is preserved. This is measured by means of the minimum *point-to-surface* distance between the tumor and the resection surface derived from Eq. (7.9).

Inter-subject reproducibility of results Surgery planning tools are essentially geometric modeling methods. This criteria considers how accurately different users can reach the same resection plan. In order to measure similarity of resections between users, we measure the resection volume difference (in %) with respect to the reference resection volume. Volumetry of resection is computed using the procedure in Section 7.3.5.

Planning time Integration in the clinical work-flow is of paramount importance for new computational methods. Therefore the planning time should improve, or at least be similar, with respect to *state-of-the-art* methods.

Smoothness of results Resection smoothness is a desirable feature. Smoothness not only helps the interpretation and visualization of 3D models, but also increases the feasibility of performing the planned resection during surgery (e.g., *"curvy"* surfaces are more difficult to perform surgically and sometimes even impossible). As indicator of surface smoothness we use the mean curvature [23]:

$$H = \frac{\mathcal{K}_1 + \mathcal{K}_2}{2} \tag{7.10}$$

where $\mathcal{K}_1$ and $\mathcal{K}_2$ are the principal curvatures.

7.4.1 Study Design

Our approach to evaluate and compare the three different planning methodologies (in the following: Bézier, CP and DS) is the design of a study where the three planning techniques are used by the same expert users in different clinical cases .The group of participants consists of 5 gastro-intestinal surgeons ({5,8,10,11,31} years of clinical experience).

The evaluation was conducted using a data-set consisting of 5 patient-specific models (obtained from the Oslo-CoMet study [24]). This data-set includes CT volumes, segmentation and 3D modeling of vessels, parenchyma (liver surface) and tumors. From this data-set, each surgeon generated 8 virtual resections (all atypical resections). Some of these resections target either single or multiple tumors. For comparison purposes and in order to avoid differences in clinical criteria—which could potentially lead to different resection plans for the same tumors—a set of resection plans was employed as reference. The reference set (median resected volume $208.98\ ml$) was generated by the most experienced surgeon in an earlier pilot study (3 months earlier). All the participants were asked to perform the same resection plan as in the reference. To do this, the participants were allowed to explore (< 5 minutes) the reference resection plan beforehand.

The experts' comments were recorded after each resection plan (see Section 6.4).

7.4.2 Procedure

Before starting the experiments (during the same session), the surgeons were shown the graphical user interface and the process to obtain resections with the different methods (CP, DS, Bézier). Surgeons were allowed to use the system to perform a sample resection as training (< 1 hour).

The experiment consisted of planning the different cases using CP, DS and Bézier for all the cases. The cases were ordered for all the participants, however, the order of the method is *a-priori* randomized to reduce the impact of confounding factors (i.e., training or sequence effects). The participants were allowed get help by a technician on any technical aspect related to the use of the interface whenever needed (due to the short training session). A resection plan was considered finalized when the participant indicated (either by obtaining the desired resection or believing the plan cannot be further improved) and the verification by a technician that the resection was complete (surface exceeds the parenchyma in all directions). Time to complete the resection plan (excluding technician assistance in questions related to user interaction and verification of resection) was recorded, together with the geometry of the 3D surface models derived from the resection plan.

7.5 Results

In this section, we present results derived from the use CP, DS and Bézier by clinicians at Oslo University Hospital, as described in Section 7.4. A descriptive analysis of quantitative results is shown in Table 7.2. Subjective feedback of the participants—which will be use as a base for discussion in Section 7.6—is recorded in Table 7.3.

In the same line as [22], we conduct statistical tests for normality of data (*Shapiro-Wilk*), difference between methods (*ANOVA, Friedman*) and pairwise differences between methods (*Wilcoxon, paired Student's t-test*) with *Bonferroni* correction [25]. Due to the Bonferroni correction, all effects derived from pairwise comparisons are reported at a 0.0167 (i.e., one third of the p-value 0.05) level of significance. Statistical analysis was carried out with the R statistics software package.

Surgery Planning Time The surgery planning time was recorded for every resection performed by the participants (Fig. 7.5a). The Friedman test reveals no significant difference between methods in terms of time, with $\mathcal{X}^2(2) = 1.849$, $p = 0.39 > 0.05$, where the median completion times were $174\ s$ for Bézier, $179\ s$ for DS and $180\ s$ for CP.

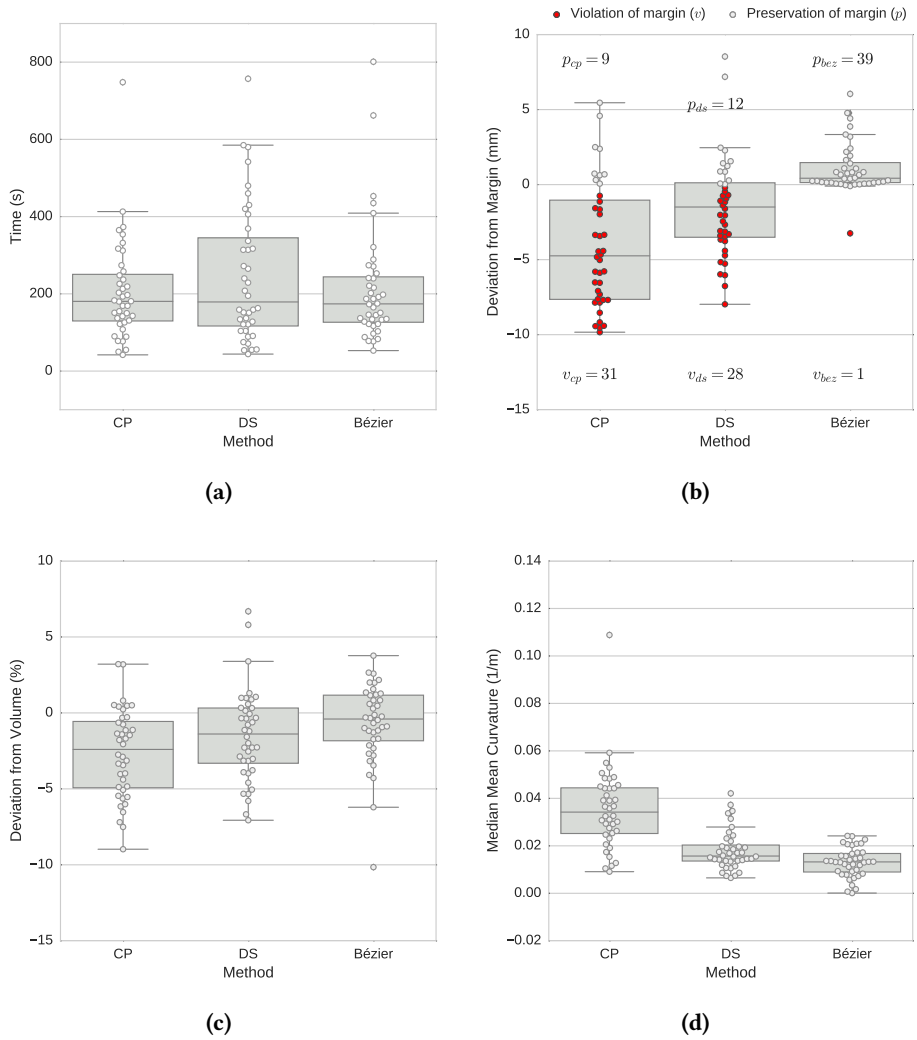


Figure 7.5: Box plots of results derived from the quantitative evaluation: (a) time, (b) deviation from margin which includes the number of resections violating the margin ($dev < -0.01\ mm$ marked in red) and the number of resections preserving the margin ($dev \geq -0.01\ mm$), (c) deviation from volume and (d) median mean curvature.

Table 7.2: Descriptive analysis derived from the quantitative evaluation

Method		Time (s)	Deviation from Margin (mm)	Deviation from Volume (%)	Median Mean Curv. (1/m)
Bézier	min	53.00	-3.25	-10.15	0.00
	25%	126.50	0.13	-1.83	0.01
	50%	174.00	0.42	-0.40	0.01
	75%	244.00	1.47	1.17	0.02
	max	801.00	6.05	3.77	0.02
CP	min	42.00	-9.84	-8.97	0.01
	25%	129.75	-7.65	-4.93	0.03
	50%	180.50	-4.75	-2.40	0.03
	75%	250.50	-1.04	-0.56	0.04
	max	748.00	5.46	3.21	0.11
DS	min	44.00	-7.98	-7.07	0.01
	25%	116.75	-3.51	-3.31	0.01
	50%	179.00	-1.49	-1.39	0.02
	75%	345.00	0.12	0.32	0.02
	max	757.00	8.53	6.69	0.04

Deviation from Resection Margin Pairwise comparison between methods in terms of deviation from resection margin (Fig. 7.5b) through Wilcoxon signed rank test yields:

- $V = 164, p = 0.0006 < 0.167$ between CP and DS,
- $V = 32, p = 5.03e-09 < 0.167$ between CP and Bézier,
- $V = 75, p = 8.69e-07 < 0.167$ between DS and Bézier.

These results show significant differences regarding the deviation of resection margin between methods. The median deviations from resection margin were $0.42\ mm$ for Bézier, $-1.49\ mm$ for CP and $-4.74\ mm$ for DS. Bézier presents the least deviation from resection margin. The number of resections violating the resection margin ($dev < -0.01$) is $v_{cp} = 31$ for CP, $v_{ds} = 28$ for DS and $v_{bez} = 1$ for Bézier.

Deviation from Reference Volume Deviation from the reference volume (Fig. 7.5c was computed as the difference (in %) between the resected volumes obtained by the participants and their corresponding resected volumes in the reference data-set. Wilcoxon signed rank test t-test yields:

- $V = 308, p = 0.174 > 0.0167$ between CP and DS,
- $V = 67, p = 3.875e - 07 < 0.0167$ between CP and Bézier,
- $V = 190, p = 0.0025 < 0.0167$ between DS and Bézier.

Pairwise tests show significant differences between Bézier and the other two methods. The median volume deviation is -0.4% for Bézier, -1.39% for DS and -2.40% for CP. Bézier shows the least deviation with respect to the reference resected volume.

Resection Curvature Pairwise comparison of resection curvature between methods (Fig. 7.5d) through Wilcoxon signed rank test yields:

- $V = 638, p = 0.001 < 0.167$ between CP and DS,
- $V = 654, p = 0.007 < 0.167$ between CP and Bézier,
- $V = 475, p = 0.39 > 0.167$ between DS and Bézier.

These results show that the difference in curvature for CP is different from both DS and Bézier. No significant difference was found between Bézier and DS. Median mean curvature is $0.01\ m^{-1}$ for both Bézier and DS, and $0.03\ m^{-1}$ for CP. Both Bézier and DS produce resections with lower curvature than CP.

7.6 Discussion

Computer-assisted systems for planning and guiding liver resections have existed for nearly two decades now. Although some of these systems have made their way into clinical reality, none of them seems to be established as a *gold-standard* solution replacing previous clinical practices. To a great extent, this is due to the difficulties of generating the patient-specific models employed by these systems—segmentation, for instance, is still considered a research problem and a bottleneck for the generation of patient-specific models. No consensus exists about planning liver resections—DS and CP approaches currently coexist in the surgery planning market. New methods for planning liver resections should, at least, highlight their differences, as well as their advantages/disadvantages with respect to the existing techniques. Therefore, in this section, a comparison of our approach with DS and CP strategies is discussed on the basis of the results presented in Section 7.5.

According to our results, the required time (median) for completion of a resection plan using the proposed method ($t = 174\ s$) is, as low as for the *state-of-the-art* methods CP ($t = 180\ s$) and DS ($t = 179\ s$). This indicates that the adoption

Table 7.3: Comments from the experts (S1-S5 indicates the participant who provided/expressed the comment).

General comments
[**GC1**] Undo functionality would be useful (S1, S4).
[**GC2**] Ability to set transparency of surfaces would be useful (S1, S5)
[**GC3**] Pre-defined views aligned to surgical way of looking at the liver would be useful (S1).
[**GC4**] Rotation of resection can be useful in some cases, specially in CP and Bézier (S1).
Comments on DS
[**CDS1**] Poses the steepest learning curve / is the least intuitive method (S1, S2, S3, S5).
[**CDS2**] Can be difficult to specify resections with high curvature (S1, S5).
[**CDS3**] Can be adequate for *quasi-planar* resections (S1).
[**CDS4**] Could not reach exactly the desired resection in some cases (S2, S3).
[**CDS5**] Some resections could be better defined by combination of traces in different views (axial, coronal, sagittal) (S3).
Comments on CP
[**CCP1**] Resections derived from drawing traces in parenchyma sometimes produce unexpected results in terms of desired curvature (S1, S3).
[**CCP2**] Modification of resections in CP present more degree of freedom (complexity) than needed. More simplicity would be a benefit (S1,S3).
[**CCP3**] *"Curvy/Bumpy/Wavy"* resection plans derived from CP can be difficult to perform surgically (S1, S3, S4, S5).
[**CCP4**] Local deformations can be useful in particular cases like peripheral metastases (S3 ,S5).
[**CCP5**] Deformation can be difficult when the initial plane is nearly perpendicular to the screen plane (S4).
Comments on Bézier
[**CB1**] Visualization of resection margin is an advantage of this method (S1).
[**CB3**] In addition to visualization of the margin on the surface, a global warning of resection violation could be useful. Sometimes violation or resection is occluded (S2).
[**CB4**] Deformation of resection in Bézier does not look obvious (S4).
[**CB5**] Bézier is the most intuitive method (S5).

of the proposed method in the clinical routine would not imply any significant change in the clinical work-flow.

Bézier shows the least deviation from the reference plan in terms of volume (-0.40%) compared to DS (-1.39%) and CP (-2.40%). In our study, small deviations from resected plans are expected from all the methods since the median resected volume for the reference data-sets is relatively small ($208.98\ ml$); larger deviations in volume are expected for larger resections (e.g., hemihepatectomies).

The comments from the participants show wide consensus on considering DS the most difficult method to use (**CDS1**), particularly for resections exhibiting higher curvature (**CDS2, CDS3**). Furthermore, in some cases, DS did not provide satisfactory results (**CDS4**); to mitigate these problems, the ability to combine traces in different views is suggested (**CDS5**). No consensus was found on whether CP or Bézier is the most intuitive (**CCP2, CB4, CB5**). Considering task completion time as indicator of usability, and despite the fact that no statistical significance between the methods was found, the higher variability of DS with respect to CP and Bézier seems to support that DS is less intuitive than CP and Bézier, which are comparable in this regard.

As discussed in [15], continuous visualization of distance from the resection surface to the tumor is a desirable feature since it is associated with the preservation of resection margin. Our results show the visualization technique proposed in Section 7.3.4 is an effective mechanism (**CB1**) to avoid violations of resection margin ($v_{bez} = 1$ for Bézier compared to $v_{cp} = 31$ and $v_{ds} = 28$); median deviation from resection margin is also lower using Bézier ($0.42\ mm$) as compared to using CP ($-4.75\ mm$) or DS ($-1.49\ mm$). Despite the good results in terms of preservation of resection margin of our proposed method, this was not sufficient to avoid all the violation of resection margin; occlusions of resection margin visualization (e.g., by vessels) might lead to unnoticed resection violations. To avoid this, and in line with the participants' comments (**CB3**), an indicator of margin violation external to the visualization of the surface should be provided (e.g., bi-color state widget in the GUI or a warning icon).

The shape of the virtual resection is an important aspect since it relates to the feasibility of performing the resection surgically; resections presenting *wavy* resection trajectories might be not realizable during surgery as they are specified in the virtual plan. In this sense, resections presenting low curvature are associated with higher surgical feasibility than resections with high curvature. According to our results, Bézier and DS provide resections which are easier to perform surgically (lower curvature) compared to CP. In this line, and according to the participants' comments (**CCP3**), using CP might lead to resections that are difficult to perform

surgically.

Some of the techniques described in this work can be employed to improve CP and DS; visualization of resection margin (**CB1**), for instance, was already discussed in [15] as a possible improvement. Some other improvements suggested in our experiments by the experts users, like the possibility of a semi-transparent visualization of resection surface (**CG2**) and the possibility to undo actions (**CG1**) were also found in [15] and should be considered for further improvement of all the methods. Rotation of the resection, particularly for CP and Bézier (**GC4**), and predefined alignments of the 3D view to surgical positions (e.g., anterior-posterior axis) (**GC3**) are also the could be implemented for methods other than Bézier.

Despite that our method showed good performance in terms of planning time, reproducibility of results, preservation of margin and curvature, expert users highlight scenarios where the use of DS and CP could be still advantageous—such as for *quasi-planar* resections (**CDS3**) like hemihepatectomies or small local resections like peripheral metastases (**CCP4**). In this regard, and since all the methods are similar in terms of time, software platforms for planning liver resections could consider including all the methods to provide clinicians with greater flexibility to represent resections. Furthermore, CP and Bézier could even be combined so that local deformations like in CP are preceeded by global deformations like in Bézier.

7.7 Conclusion

In this work we propose a novel method for planning liver resection procedures. This method is based on the use of deformable Bézier surfaces for the specification of resection geometry and the projection of risk areas (representing violations of safety margins) onto the resection surface through distance maps. Our implementation of the method includes mechanisms to reduce the number of interactions making the system easy-to-use by clinicians.

Our experimental results show that the planning time of our method is as low as *state-of-the-art* methods, and therefore, can be integrated in the clinical reality without modifications in the clinical work-flow. Our method, not only shows superior preservation of resection margin methods, but also higher reproducibility of surgery planning results than *state-of-the-art*. In addition, the proposed method provides smooth virtual resections presenting high feasibility to be performed surgically (e.g., absence of sharp corners and wavy trajectories).

Acknowledgments

This work was supported by the Research Council of Norway through the Hypercept project (number 221073), and The Intervention Centre, Oslo University

Hospital (Norway). Authors thank Leonid Barkhatov, David Aghayan, Sheraz Yaqub, Mushegh Sahakyan, Kristoffer Lassen and Bård I. Røsok for fruitful discussions and their contribution to the evaluation of the method proposed in this work. Authors would like to thank also Xiaoran Lai for reviewing the data analysis.

Bibliography

[1] Lindsey A. Torre, Freddie Bray, Rebecca L. Siegel, Jacques Ferlay, Joannie Lortet-tieulent, and Ahmedin Jemal. Global Cancer Statistics, 2012. *CA: a cancer journal of clinicians.*, 65(2):87–108, 2015. ISSN 1542-4863 (Electronic). doi: 10.3322/caac.21262.

[2] C Couinaud. *Le foie: études anatomiques et chirurgicales.* Masson & Cie, 1957.

[3] Robert J Aragon and Naveenraj L Solomon. Techniques of hepatic resection. *Journal of gastrointestinal oncology*, 3(1):28–40, 2012. ISSN 2219-679X. doi: 10.3978/j.issn.2078-6891.2012.006.

[4] Ester Vanni and Elisabetta Bugianesi. Obesity and liver cancer. *Clinics in liver disease*, 18(1):191–203, feb 2014. ISSN 1557-8224. doi: 10.1016/j.cld.2013.09.001.

[5] Richard Bryant, Alexis Laurent, Claude Tayar, Jeanne Tran van Nhieu, Alain Luciani, and Daniel Cherqui. Liver resection for hepatocellular carcinoma. *Surgical oncology clinics of North America*, 17(3):607–33, ix, jul 2008. ISSN 1055-3207. doi: 10.1016/j.soc.2008.02.002.

[6] Evangelos P Misiakos, Nikolaos P Karidis, and Gregory Kouraklis. Current treatment for colorectal liver metastases. *World journal of gastroenterology : WJG*, 17(36):4067–75, sep 2011. ISSN 2219-2840. doi: 10.3748/wjg.v17.i36.4067.

[7] Matthew J. Schuchert, Brian L. Pettiford, James D. Luketich, and Rodney J. Landreneau. Parenchymal-Sparing Resections: Why, When, and How. *Thoracic Surgery Clinics*, 18(1):93–105, 2008. ISSN 15474127. doi: 10.1016/j.thorsurg.2007.11.007.

[8] Giovanni Vennarecci, Andrea Laurenzi, Roberto Santoro, Marco Colasanti, Pasquale Lepiane, and Giuseppe Maria Ettorre. The ALPPS procedure: A surgical option for hepatocellular carcinoma with major vascular invasion. *World Journal of Surgery*, 38(6):1498–1503, 2014. ISSN 14322323. doi: 10.1007/s00268-013-2296-y.

[9] W Lamadé, G Glombitza, L Fischer, P Chiu, C E Cárdenas, M Thorn, H P Meinzer, L Grenacher, H Bauer, T Lehnert, and C Herfarth. The impact of 3-dimensional reconstructions on operation planning in liver surgery. *Archives of surgery (Chicago, Ill. : 1960)*, 135(11):1256–61, nov 2000. ISSN 0004-0010.

[10] Hauke Lang, Arnold Radtke, Milo Hindennach, Tobias Schroeder, Nils R Frühauf, Massimo Malagó, Holger Bourquain, Heinz-Otto Peitgen, Karl J Oldhafer, and Christoph E Broelsch. Impact of virtual tumor resection and computer-assisted risk analysis on operation planning and intraoperative strategy in major hepatic resection. *Archives of surgery (Chicago, Ill. : 1960)*, 140(7):629–38; discussion 638, jul 2005. ISSN 0004-0010. doi: 10.1001/archsurg.140.7.629.

[11] C Hansen, S Zidowitz, and B Preim. Impact of model-based risk analysis for liver surgery planning. *International Journal of Computer Assisted Radiology and Surgery*, 9(3):473–80, may 2014. ISSN 1861-6429. doi: 10.1007/s11548-013-0937-0.

[12] Pablo Lamata, Félix Lamata, Valentin Sojar, Piotr Makowski, Laurent Massoptier, Sergio Casciaro, Wajid Ali, Thomas Stüdeli, Jérôme Declerck, Ole Jakob Elle, and Bjørn Edwin. Use of the Resection Map system as guidance during hepatectomy. *Surgical endoscopy*, 24(9):2327–37, sep 2010. ISSN 1432-2218. doi: 10.1007/s00464-010-0915-3.

[13] Bernhard Preim and Charl P Botha. *Visual Computing for Medicine: Theory, Algorithms, and Applications*. Newnes, 2nd edition, 2013.

[14] László Ruskó, Ilona Mátéka, and András Kriston. Virtual volume resection using multi-resolution triangular representation of B-spline surfaces. *Computer methods and programs in biomedicine*, 111(2):315–29, aug 2013. ISSN 1872-7565. doi: 10.1016/j.cmpb.2013.04.017.

[15] O Konrad-Verse, Arne Littmann, and Bernhard Preim. Virtual Resection with a Deformable Cutting Plane. In *SimVis*, pages 203–214, 2004.

[16] 3D Slicer.

[17] Les Piegl and Wayne Tiller. *The NURBS Book (2Nd Ed.)*. Springer-Verlag New York, Inc., New York, NY, USA, 1997. ISBN 3-540-61545-8.

[18] Jean Gallier. *Curves and Surfaces in Geometric Modeling: Theory and Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000. ISBN 1-55860-599-1.

[19] Géraldine Morin and Ron Goldman. On the smooth convergence of subdivision and degree elevation for Bézier curves. *Computer Aided Geometric Design*, 18(7):657–666, 2001. ISSN 01678396. doi: 10.1016/S0167-8396(01)00059-0.

[20] WE Lorensen and HE Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, 21(4):163–169, 1987.

[21] Timothy S. Newman and Hong Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5):854–879, oct 2006. ISSN 00978493. doi: 10.1016/j.cag.2006.07.021.

[22] S Glaßer, P Saalfeld, P Berg, N Merten, and B Preim. How to Evaluate Medical Visualizations on the Example of 3D Aneurysm Surfaces. In *Eurographics Visual Computing for Biology and Medicine*, number August, 2016. ISBN 978-3-03868-010-9. doi: 10.2312/vcbm.20161283.

[23] Ron Goldman. Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design*, 22(7 SPEC. ISS.):632–658, 2005. ISSN 01678396. doi: 10.1016/j.cagd.2005.06.005.

[24] AAsmund Avdem Fretland, Airazat M Kazaryan, Bjorn Atle Bjornbeth, Kjersti Flatmark, Marit Helen Andersen, Tor Inge Tonnessen, Gudrun Maria Waaler Bjornelv, Morten Wang Fagerland, Ronny Kristiansen, Karl Oyri, and Bjorn Edwin. Open versus laparoscopic liver resection for colorectal liver metastases (the Oslo-CoMet study): study protocol for a randomized controlled trial. *Trials*, 16(1):1–10, 2015. ISSN 1745-6215. doi: 10.1186/s13063-015-0577-5.

[25] Juliet P. Shaffer. Multiple hypothesis testing. *Annual Review of Psychology*, 46:561–584, 1995. ISSN 15734412. doi: 10.1016/S1573-4412(84)02006-7.

CHAPTER 8

Paper III: High-Performance Computation of Bézier Surfaces on Parallel and Heterogeneous Platforms

Rafael Palomar · Juan Gómez-Luna · Faouzi A. Cheikh · Joaquín Olivares-Bueno · Ole J. Elle

Abstract Bézier surfaces are mathematical tools employed in a wide variety of applications. Some works in the literature propose parallelization strategies to improve performance for the computation of Bézier surfaces. These approaches, however, are mainly focused on graphics applications and often are not directly applicable to other domains. In this work, we propose a new method for the computation of Bézier surfaces, together with approaches to efficiently map the method onto different platforms (CPUs, discrete and integrated GPUs). Additionally, we explore CPU-GPU cooperation mechanisms for computing Bézier surfaces using two integrated heterogeneous systems with different characteristics. An exhaustive performance evaluation—including different data-types, rendering and several hardware platforms—is performed. The results show that our method achieves speedups as high as 3.12x (double-precision) and 2.47x (single-precision) on CPU, and 3.69x (double-precision) and 13.14x (single-precision) on GPU compared to other methods in the literature. In heterogeneous platforms, the CPU-GPU cooperation increases the performance up to 2.09x with respect to the GPU-only version. Our method and the associated parallelization approaches can be easily employed

in domains other than computer-graphics (e.g., image registration, bio-mechanical modeling and flow simulation), and extended to other Bézier formulations and Bézier constructions of higher order than surfaces.

8.1 Introduction

Bézier tensor-product surfaces (in the following referred to as Bézier surfaces) are geometric constructions widely used in engineering and computer-graphics. Despite Bézier curves and surfaces have been studied for decades, they are still an active field of research [1, 2].

Due to their simplicity and mathematical properties, Bézier surfaces have been employed in applications such as surface reconstruction from clouds of points [3], modeling of free-form deformations [4, 5], interactive manipulation of three-dimensional meshes and rendering [6, 7, 8], bio-mechanical modeling [9], hybrid volumetric object representation [10], registration in medical imaging [11, 12], and computer games [13] among others.

Computation of tensor-product Bézier constructions—regardless of whether these are surfaces or higher order constructions like volumes—is considered a computationally expensive task. Applications such as shape optimization in aerodynamics [14], flow modeling [15], simulation [16] and non-rigid medical image registration [11] require, indeed, high-degree Bézier formulations to cope with the complexity of the underlying data.

In the last decade, strategies to parallelize the evaluation[1] of Bézier surfaces have been developed (Section 8.3). These strategies, however, circumscribe mostly to the field of computer-graphics as part of tessellation applications (conversion of continuous surfaces to discrete triangle meshes). Furthermore, these strategies are often limited to the computing of bi-cubic Bézier patches widely used in rendering and animation.

New trends in computing like heterogeneous computing systems (**HCS**), where multi-core processors are integrated (on-chip) with GPUs in the same device, allow new possibilities for improving the performance. These systems, as opposed to traditional computing systems (e.g., CPU and GPU in separate devices) establish cooperation mechanisms across computing units (e.g., CPU+GPU).

In the literature, works evaluating the performance of traditional systems often present their results as a comparison of devices competing to reach the higher performance. In this context, the performance of multi-core CPUs, GPUs and FPGAs have been evaluated in multiple application domain like image processing

[1] In the line of other related works, we use the term *evaluation* to refer to computation.

[17] and computer graphics [18]. As opposed to this approach, HCS evaluate performance results in terms of cooperative work across computing units. These mechanisms range from distribution of the same processing stage among the computing units [19] to distribution of processing stages (from pipelines) based on optimal mapping to the most adequate computing unit [20, 21]. To date, organization of new applications using HCS is an active area of research; in order to understand the challenges and opportunities for HCS for leveraging improved performance over traditional computing systems, benchmark suites like Hetero-Mark[22] and CHAI [23] have been recently developed.

Generalized parallel strategies going beyond bi-cubic Bézier schemes, together with techniques to map the parallelization efficiently onto different hardware platforms, including HCS, have consequently the potential to make an impact in the performance of not only computer-graphics, but a broader range of applications.

8.1.1 Contribution

The aim of this work is computing real-time Bézier tensor-product surfaces that can be employed not only in rendering applications—where bi-cubic Bézier surfaces are predominant—but also in applications requiring high-degree surfaces.The main contribution of this work is threefold:

- A multi-level method (**MLE**) for the computation of parametric non-rational Bézier tensor-product surfaces of arbitrary degree. The use of this method can be further applied to other formulations (e.g., rational Bézier), as well as tensor-products of higher order than surfaces.
- We propose different techniques to map MLE onto different hardware platforms, including central processing units (**CPU**), discrete and integrated graphics processing units (**GPU**) as well as mobile integrated GPUs—these latter ones being poorly explored in the literature.
- As the latest trends in computing move towards hybrid systems (more than one kind of processor present), we also propose CPU-GPU cooperation mechanisms, including the exploitation of (HCS) models with different properties.

In addition, we review and classify the most important works in the literature concerned with the optimization and acceleration of computation of Bézier surfaces.

The rest of the paper is organized as follows. Section 8.2 provides fundamental mathematical background on Bézier surfaces. Section 8.3 lists and shortly reviews relevant works in the literature which accelerate and optimize the computation

of Bézier surfaces. In Section 8.4 the proposed method (MLE) is described. Section 8.5, on other hand, addresses the parallelization and mapping of MLE onto different computing platforms, including CPUs, GPUs and HCSs. In Section 8.6, our experiments and results are described. These results and the most relevant findings are discussed in Section 8.7. Finally, in Section 8.8, some concluding remarks are presented.

8.2 Background

In this section, a brief description of Bézier surfaces is provided. A deeper description of this type of surfaces and its properties can be found in [24]. For simplicity and clarity reasons, in this work, the focus is on the use of the parametric non-rational formulation of Bézier surfaces. However, the methods presented in this paper are generalizable to other Bézier tensor-product formulations (e.g., rational formulations or higher order tensors).

Mathematically, non-rational Bézier tensor-product surfaces $\mathbf{S} : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ are defined as:

$$\mathbf{S}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{P}_{i,j} B_{i,m}(u) B_{j,n}(v), \tag{8.1}$$

where $u, v \in [0,1]$ form the parametric space of the surface and $\mathbf{P}_{i,j}$ are control points. The m and n values determine the degree of the Bernstein polynomials $B_{i,m}(u)$ and $B_{j,n}(v)$ used as basis functions. These polynomials are generically defined as:

$$B_{i,m}(u) = \binom{m}{i} (1-u)^{(m-i)} u^i, \tag{8.2}$$

with $0 \leq i \leq m$. $B_{j,n}(v)$ is defined similarly.

The most common case of Bézier surface in the scientific literature is the bi-cubic surface ($m = n = 3$). An example of this type of surface together with its control points is shown in Figure 8.1. Bézier surfaces can also be expressed in terms of the matrix product:

$$\mathbf{S}(u,v) = \mathbf{U}(u)\mathbf{R}(m)\mathbf{P}\mathbf{R}(n)^T\mathbf{V}(v)^T, \tag{8.3}$$

where the $\mathbf{P}$ is the matrix representing the net of control points. This matrix is

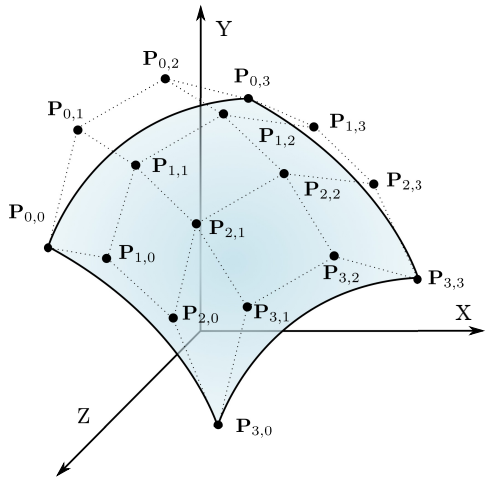


Figure 8.1: Bi-cubic Bézier tensor-product surface and its 4×4 net of control points.

given by:

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \dots & \mathbf{P}_{0,n} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \dots & \mathbf{P}_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}_{m,0} & \mathbf{P}_{m,1} & \dots & \mathbf{P}_{m,n} \end{bmatrix}.$$

.

The vectors $\mathbf{U}$ and $\mathbf{V}^T$ are polynomial spaces of degree m and n, associated to the parameterization directions u and v respectively. Generically, these basis vectors take the form $\mathbf{T}(t) = [t^\alpha, t^{\alpha-1}, ..., t^0]$, where α is the degree of the polynomial space. The matrix of coefficients $\mathbf{R}$ is then defined as:

$$\mathbf{R}(t) = \begin{bmatrix} \binom{t}{0}\binom{t}{t}(-1)^t & \binom{t}{1}\binom{t-1}{t-1}(-1)^{t-1} & \dots & \binom{t}{t}\binom{t-1}{t-1}(-1)^0 \\ \vdots & \vdots & \ddots & \vdots \\ \binom{t}{0}\binom{t}{1}(-1)^1 & \binom{t}{1}\binom{t-1}{0}(-1)^0 & \dots & 0 \\ \binom{t}{0}\binom{t}{0}(-1)^0 & 0 & \dots & 0 \end{bmatrix}.$$

In the literature, some authors like [25] express Equation (8.3) in a more compact form:

$$\mathbf{S}(u, v) = \mathbf{U}(u)\mathbf{G}\mathbf{V}(v)^T \tag{8.4}$$

where $\mathbf{G} = \mathbf{R}(m)\mathbf{P}\mathbf{R}(n)^T$ is constant. This has important implications under the point of view of the implementation and optimization.

8.3 Related work

To a great extent, the driving force behind the use and development of Bézier surfaces has been the computer-graphics community. In this context, tessellation algorithms emerged as a mechanism for converting surfaces to triangle meshes [26].

As Bézier surfaces were gaining popularity—eventually becoming the standard for the representation and communication of geometric data—performance became more important. Initially, tessellation was performed in the CPU, then the results were sent to the GPU for further processing and visualization. This transferring process was considered as a bottleneck for high-quality surfaces (which imply high number of triangle transfers). To address this issue, [27] proposed an algorithm and a specific hardware architecture integrated in the GPU.

Later, the GPUs evolved into programmable parallel processors where the graphics pipeline could be redefined by software. These GPUs, included two new programmable units, the *vertex processing unit* dealing with geometry and attributes (i.e., texture coordinates, colors, etc.) and *fragment processing unit* dealing with data stored in textures. User-defined programs, also referred to as *shaders*, were also structured into either *vertex programs* or *fragment programs*. Some works made use of this approach to evaluate and render Bézier surfaces [7, 12, 28, 29, 30].

A more contemporary trend to exploit the massive parallelism of graphics hardware is the *general purpose GPU* (**GPGPU**), made available through the CUDA [31] and OpenCL [32] programming frameworks. Some works make use of this approach for the evaluation of Bézier surfaces not only with applications to computer-graphics [4, 8, 33, 34, 35, 36].

Regardless of the implementation mechanisms, many of the parallelization strategies can be utilized in both shaders and GPGPU approaches. These strategies can be roughly classified into algorithmic strategies or hardware-specific strategies.

Algorithmic strategies are generally concerned with reducing the number of operations to perform. In this line, [7] uses the matrix formulation in Equation (8.3) instead of Equation (8.1). Later in [25], the authors make use of Equation (8.4), which allows, not only the reduction of the number of operations needed (constant values are pre-calculated), but also the exploitation of spatial coherence of data. Other algorithmic strategies employ numerical approximations, like [8] which is based on forward differencing [37]. However, these methods are subject to error accumulation, and therefore, the generalization to high-degree surfaces is limited.

Hardware-specific strategies are based on providing an efficient mapping of the

Table 8.1: Summary of GPU evaluation of Bézier tensor products in the scientific literature.

Publication	Bézier formulation	Max. degree evaluated	Optimization strategies		Implementation		Rendering
			Algorithmic	Hardware	Shaders	GPGPU	
[29]	Non-rational	4×4			•		•
[27]	Non-rational	3×3	•	•			•
[33]	Non-rational	3×3			•		•
[12]	Non-rational	3×3			•		
[8]	Rational	3×3	•	•		CUDA	•
[34]	Rational	3×3		•		CUDA	•
[38]	Rational	3×3		•	•		•
[30]	Non-rational	N/A		•	•		•
[35]	Rational	N/A				CUDA	
[7]	Non-rational	3×3	•		•		•
[25]	Non-rational	3×3	•		•		•
[4]	Non-rational	N/A	•	•		CUDA	•
Our work	Non-rational	12×12	•	•		CUDA/OpenCL	*

* Our work does not target rendering applications specifically, however, we provide results with rendering through graphics interoperability.

method on the underlying hardware. The flexibility provided by CUDA and OpenCL allows for a more fine-grained mapping of the method than that obtained by using shaders. [8] make use of a selective transfer of control points to the fast on-chip memory of the GPU (in the following referred to as GPU shared memory), thus providing fast access to those elements frequently accessed during computations. Additionally, [34] utilizes a selective distribution of threads (one GPU thread per control point for evaluation of bi-cubic Bézier patches and one GPU thread per patch for subsequent processing). In [38], the authors present a more generic evaluation (based on non-uniform rational B-splines) approach in which the operations are distributed between CPU and GPU, so that inherently serial operations are carried out by the CPU.

Despite the recent advances of computing in mobile devices, the evaluation of Bézier surfaces in these devices has been given very little attention. To the best of our knowledge, the only work bringing evaluation of Bézier surfaces in mobile platforms is [7]. In this work, the authors highlight the difficulties for real-time tessellation of complex objects.

A summary of all the works considered in this section can be found in Table 8.1. For each of these works, the table includes the type of Bézier formulation, maximum degree evaluated, employed optimization strategies, programming model used, and whether rendering was the purpose of the application.

8.4 Multi-level evaluation of Bézier surfaces

On the basis of designing a flexible algorithm able to adapt to different applications, we define the following requirements:

(A) Update of control points coordinates: this is the most common criterion for real-time evaluation of Bézier surfaces. In this case, the coordinates of the control points change in every evaluation cycle, while the number of control points $[(m+1)\times(n+1)]$ and surface resolution $(\rho\times\delta)$ remain invariant. Applications related to evaluation of Bézier surfaces in regular grids, like 3D representation using Bézier patches or deforming surfaces, meet this requirement.

(B) Variable resolution of surface: subsequent evaluations of the surface present different resolutions $(\rho\times\delta)$ (e.g. tessellation applications).

(C) Variable degree of the surface: this implies a change on the number of control points $[(m+1)\times(n+1)]$ in subsequent evaluations (e.g., applications related to degree elevation and surface subdivision).

In order to fulfill these requirements and to reduce the number of operations, we propose an approach based on the a decomposition of the Bézier formulation in a hierarchy of levels. First, we expand Equation (8.1) as:

$$\mathbf{S}(u,v)=\sum_{i=0}^{m}\sum_{j=0}^{n}\mathbf{P}_{i,j}\overbrace{\underbrace{\binom{n}{i}}_{\mathbf{C^u}}(1-u)^{(n-i)}u^{i}}^{\mathbf{B^u}}\overbrace{\underbrace{\binom{m}{j}}_{\mathbf{C^v}}(1-v)^{(m-i)}v^{j}}^{\mathbf{B^v}}. \tag{8.5}$$

From this formulation, where the different terms, for all points in the surface, are computed and stored in arrays: $\mathbf{B^u}$ and $\mathbf{B^v}$ for arrays of Bernstein basis with lengths $|\mathbf{B^u}|=(m+1)\rho$ and $|\mathbf{B^v}|=(n+1)\delta$ for the directions u and v respectively; $\mathbf{C^u}$ and $\mathbf{C^v}$ for arrays of binomial coefficients with lengths $|\mathbf{C^u}|=m+1$ and $|\mathbf{C^v}|=n+1$ for the directions u and v respectively.

From a data-dependency standpoint, those arrays can be structured into a hierarchy of levels (Figure 8.2) in which each level directly corresponds to the computation of a set of terms:

Level 1 is formed by the coordinates of the set of points $\mathbf{S}$ belonging to the Bézier surface. This level requires the array of control points $\mathbf{P}$ and the Bernstein polynomials corresponding to the u and v directions ($\mathbf{B^u}$ and $\mathbf{B^v}$ respectively), which could have been pre-calculated. A description of the computation of Bézier surfaces with pre-calculated Bernstein basis is given in Algorithm 4. Similarly to Equation (8.3), the computation of *level 1* is equivalent to a matrix-vector form of a Bézier tensor-product:

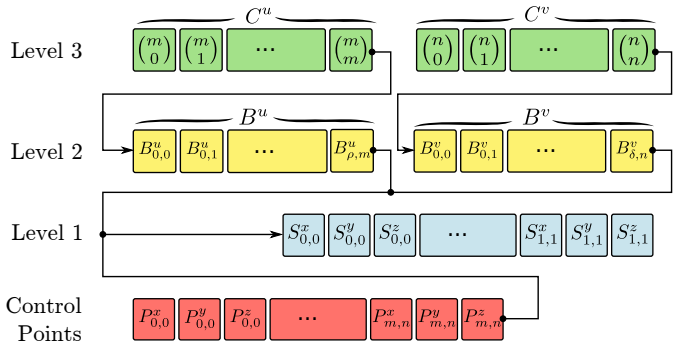


Figure 8.2: Decomposition of Bézier formulation in a hierarchy of levels and the associated data items and dependencies.

$$\mathbf{S}(u,v) = \boldsymbol{\phi}(u)^T \mathbf{P} \boldsymbol{\psi}(v), \tag{8.6}$$

where $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ are subsets of $\mathbf{B^u}$ and $\mathbf{B^v}$, and $\mathbf{P}$ is the matrix of control points previously described. The complexity of *level 1* is characterized by $\mathcal{O}(\rho \times \delta \times m \times n)$.

Level 2 represents the basis $\mathbf{B^u}$ and $\mathbf{B^v}$ of the tensor-product (Bernstein polynomials) in the directions u and v respectively, which complexity is characterized by $\mathcal{O}(\rho \times m + \delta \times n)$. Under isotropy conditions (i.e., equal number of control points and resolution for the directions u and v), $\mathbf{B^u}$ equals $\mathbf{B^v}$, and therefore one of these arrays can be obtained from the other by either memory copy or direct memory addressing. A description of the process, including the *copy/direct-addressing* mechanism, is described in Algorithm 5.

Level 3 is composed by the array of binomial coefficients $\mathbf{C}^u$ and $\mathbf{C}^v$ which are employed in the computation of the Bernstein basis arrays $\mathbf{B}^u$ and $\mathbf{B}^v$. Algorithm 6 shows the computation of the binomial coefficients. As in the Bernstein basis (*level 2*), there is no need of duplicated calculation of both $\mathbf{C}^u$ and $\mathbf{C}^v$ under isotropy conditions. The complexity of *level 3* is characterized by $\mathcal{O}(m+n)$.

As shown in Figure 8.2, the multi-level evaluation of Bézier surfaces opens up the computation (per evaluation cycle) of only those coefficients needed, while reusing all those coefficients remaining invariant. Hence, for evaluations where the number of control points and resolution are constant (i.e., requirement [A]), we can re-utilize pre-computed *level 2* and *level 3*, therefore using computing resources for only *level 1*. Following the same logic, evaluations with variable resolution (requirement [B]) can re-utilize *level 3*, only computing *level 1* and *level 2* for every cycle. Finally, in case the number of control points changes (requirement [C]), all levels need to be computed every cycle.

Algorithm 4 Bézier Surface (*level 1*)

function BézierSurface($m, n, \rho, \delta, \mathbf{B^u}, \mathbf{B^v}, \mathbf{P}$)
  ▷ *Loop over resolution in u direction*
  for $i \leftarrow 0$ to $\rho - 1$ **do**
  ▷ *Loop over resolution in v direction*
    for $j \leftarrow 0$ to $\delta - 1$ **do**
      $\mathbf{S}_{i,j} \leftarrow 0$
  ▷ *Loop over control points in u direction*
      for $k \leftarrow 0$ to m **do**
  ▷ *Loop over control points in v direction*
        for $l \leftarrow 0$ to n **do**
          $\mathbf{S}_{i,j} \leftarrow \mathbf{S}_{i,j} + \mathbf{P}_{k,l} \times B^u_{i,k} \times B^v_{j,l}$
        end for
      end for
    end for
  end for
  return S
end function

Algorithm 5 Bernstein basis (*level 2)*

function BernsteinBasis($m, n, \mathbf{C^u}, \mathbf{C^v}, \rho, \delta$)
  for $i \leftarrow 0$ to $\rho - 1$ **do**
    $\mu_i \leftarrow \frac{i}{(\rho-1)}$
    for $j \leftarrow 0$ to m **do**
      $B^u_{i,j} \leftarrow C^u_j \times \mu_i^j \times (1-\mu_i)^{\rho-1-j}$
    end for
  end for
  if $m \neq n$ or $\rho \neq \delta$ **then** ▷ *Isotropy check*
    for $i \leftarrow 0$ to $\delta - 1$ **do**
      $\mu_i \leftarrow \frac{i}{(\delta-1)}$
      for $j \leftarrow 0$ to n **do**
        $B^v_{i,j} \leftarrow C^v_j \times \mu_i^j \times (1-\mu_i)^{\delta-1-j}$
      end for
    end for
  else
    $\mathbf{B^v} \leftarrow \mathbf{B^u}$ ▷ *Copy/Direct-addressing* on isotropy
  end if
  return $[\mathbf{B^u}, \mathbf{B^v}]$
end function

Algorithm 6 Binomial coefficients (*level 3*)

```
1: function BinomialCoefficients(m, n)
   ▷ This loop computes the binomial coefficients for u-direction basis
2:     for i ← 1 to m do
3:         C_i^u ← (m−1)! / (i!(m−i−1)!)
4:     end for
5:     if m ≠ n then                                   ▷ Isotropy check
   ▷ This loop computes the binomial coefficients for v-direction basis
6:         for j ← 1 to n do
7:             C_j^v ← (n−1)! / (j!(n−j−1)!)
8:         end for
9:     else
10:        C^v ← C^u                                   ▷ Copy/Direct-addressing on isotropy
11:    end if
12:    return [C^u, C^v]
13: end function
```

8.5 Parallel implementations

The evaluation of Bézier surfaces is a problem suitable for parallelization due to the lack of data dependencies between output data items (i.e., 3D surface points). The simplest way to parallelize is the use of OpenMP [39] pre-processor directives on multi-core CPUs. For instance, Algorithm 4 can be easily parallelized employing the directive: `#pragma omp parallel for`.

More interestingly, the huge amount of points in a typical Bézier surface matches the availability of computing resources in massively parallel processors such as GPUs. The following section describes our GPU implementation of the evaluation of Bézier surfaces. Afterwards, we explore the cooperation of CPU and GPU on integrated heterogeneous systems in order to attain further acceleration.

8.5.1 GPU parallel computing

Algorithmically, the multi-level evaluation approach proposed in the previous section facilitates the parallelization possibilities of the evaluation of Bézier surfaces. Hence, the parallelization strategy described in this work follows a scheme based on the parallelization of levels, particularly *level 1* and *level 2*. The reader should note that these levels require significantly more operations than *level 3* (e.g, bi-cubic surfaces require the computation of only $2 \times 4 \times 4$ binomial coefficients), even for high-degree surfaces. Furthermore, for most of the applications, *level 3* remains unchanged. Therefore, *level 3* can be computed by the CPU with a

negligible impact.

Parallelization of *level 1* (Algorithm 7) and *level 2* (Algorithm 8) is carried out in different kernel functions using a *gather approach* [40], this is, assigning a GPU thread to each output data item (i.e., Bernstein basis coefficient for *level 2* and 3D surface point for *level 1*). This pattern ensures that threads have write access to disjoint memory locations, thus avoiding mutual exclusion and thread synchronization mechanisms, which may introduce serialization. The geometry of the kernels is 2D thread-blocks clustered in a 2D grid [2], where the block/thread indexes are used to address memory locations related to a particular thread. The parallelization optimizations we apply to these kernel functions can be better understood in terms of existing data dependencies in the computation of an output data item and its neighboring data items within the same block (Figure 8.3a), and mapping of data items (basis, surface points and binomial coefficients) onto different processors and memory locations (Figure 8.3b).

As shown in Figure 8.3a, a GPU thread assigned to the computation of a 3D output surface point (light yellow tile) requires: all the control points, a subspace of the basis $\mathbf{B^u}$ and $\mathbf{B^v}$, as well as all the binomial coefficients (all these dependencies highlighted in dark red in the figure). Similarly, neighboring GPU threads within a block (dotted tiles) require: all the control points, all the binomial coefficients and neighboring sub-spaces of $\mathbf{B^u}$ and $\mathbf{B^v}$ (in the figure, tiles are highlighted with a dot).

Considering these data dependencies, Figure 8.3b shows the distribution of memory in the GPU. Following the line of [8], in our *level 1*, control points are transferred to GPU shared memory (Algorithm 7, lines 2-4), which is a fast on-chip memory that is accessible by all threads within a block. Additionally, in our work, sub-spaces of $\mathbf{B^u}$ and $\mathbf{B^v}$ are transferred to GPU shared memory (Algorithm 7, lines 5-12), since these elements are going to be accessed frequently. Spatial locality of control points and basis ensures coalesced memory accesses, as consecutive threads load consecutive data items. After loading the data items into GPU shared memory, and given that not all threads within a block perform the same amount of memory transfers, intra-block synchronization is necessary (Algorithm 7, line 13).

Computation of Bernstein basis (*level 2*) in GPU can not benefit from using GPU shared memory since the binomial coefficients are accessed only once. However, as in the case of CPUs, it is possible to use the *copy/direct-addressing* mechanism in order to reduce the number of basis elements computed (Algorithm 8, lines

[2] CUDA threads and thread blocks correspond to OpenCL work-items and work-groups respectively. In this work, we use the CUDA terminology.

9-19).

Distribution of data items across memory units can be complemented with distribution of computing across processors. To a great extent, the design and distribution of computing is guided by the application. As shown in Figure 8.3b, and due to the inherent parallelism, it is indicated that *level 1* and *level 2* are processed by the GPU. Alternatively, and for cases where *level 2* is constant (i.e., constant resolution and number of control points), *level 2* can be pre-computed in CPU. Changes in coordinates of the control points often happen upon user interaction or in a pre-defined way, CPU is therefore adequate. Although these distributions imply cooperation between processors, CPU and GPU do not operate simultaneously, but sequentially one after another. In the next section, we present more elaborated cooperation techniques which allow processors to operate simultaneously in *level 1* (as shown in Figure 8.3b), including inter-processor coordination mechanisms.

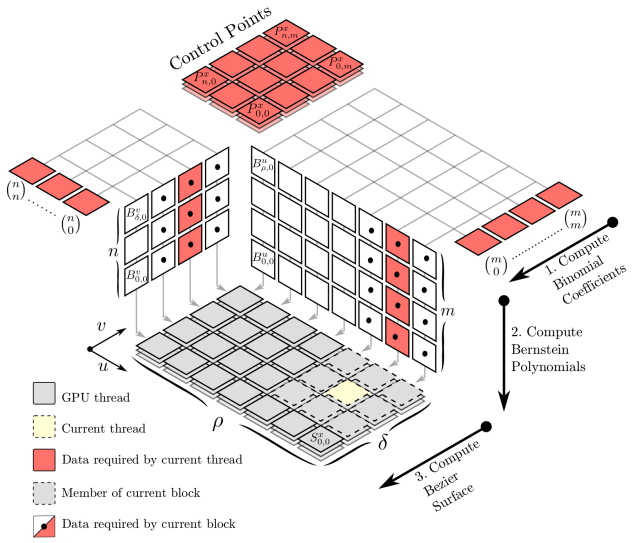
8.5.2 Heterogeneous parallel computing

Heterogeneous computing systems (**HCS**) are composed by hybrid collections of processors (frequently GPUs and CPUs) in the same system [41], and often in the same chip. This trend is intended to satisfy the computational needs of every workload. Inherently sequential or modestly parallel computations are typically executed on the CPU side, while massively parallel phases are executed on the GPU side.

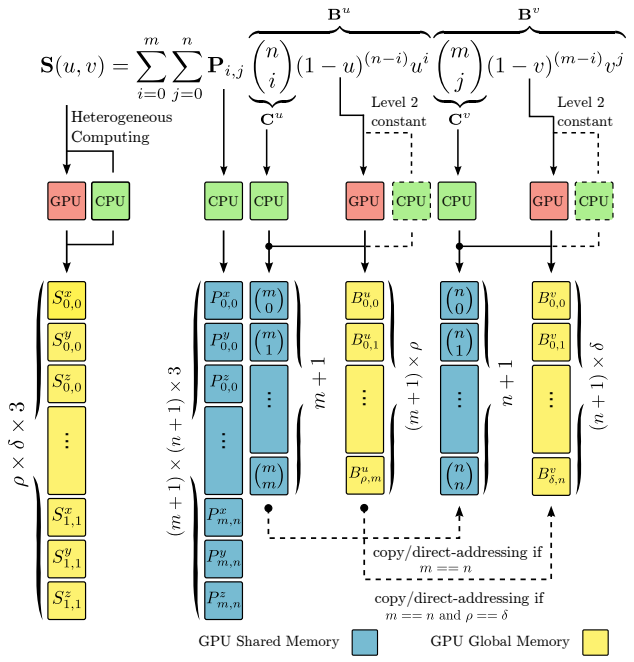
Besides discrete heterogeneous systems where CPU and GPU are connected through *peripheral component interconnect express* (**PCIe**), a more recent trend integrates CPU and GPU cores on the same die. The integrated heterogeneous systems solve the bottleneck of the data transfers through the PCIe bus by means of a unified *dynamic random-access memory* (**DRAM**).

Current developments try to facilitate communication and concurrency across CPU and GPU cores. The *heterogeneous system architecture* (**HSA**) [42] provides cache coherence mechanisms [43] and cross-device atomic operations [44]. These systems allow CPU and GPU cores to access the same memory space simultaneously.

A common way to exploit CPU-GPU cooperation is assigning serial tasks to the CPU and parallel tasks to the GPU. However, the regularity of the operations involved in the evaluation of Bézier surfaces makes possible cooperation strategies in which both GPU and CPU can perform the same operations. In this line, we use two CPU-GPU schemes based on the distribution of computation between processors through tiling [45]. The use of tiling is supported by the fact that output surface elements are independent and written in disjoint memory locations.



(a)



(b)

Figure 8.3: Parallel computation of Bézier surfaces using MLE under GPU and heterogeneous computing approaches. (a) Geometry of the data and data dependencies at different levels for a GPU thread and its neighboring threads in a thread-block. (b) Distribution of MLE elements across computing units and memory units.

Algorithm 7 Bézier Surface GPU kernel (*level 1*).

Note: s_x, s_y block sizes in the x and y dimensions. t_x, t_y thread indexes in x and y dimensions. Elements with double-dot accent (e.g. $\ddot{B}_i^u$) represent allocations in GPU shared memory.

1: **function** BézierSurfaceGPU($m, n, \rho, \delta, \mathbf{B^u}, \mathbf{B^v}, \mathbf{P}$)

▷ *This loop transfers the control points to shared memory*

2: **for** $i \leftarrow t_y \times s_x + t_x$ **to** $m \times n$ **step** $s_x \times s_y$ **do**

3: $\ddot{\mathbf{P}}_i \leftarrow \mathbf{P}_i$ ▷ *Load into shared memory*

4: **end for**

▷ *This loop transfers needed basis (u-direction) elements to shared memory*

5: **for** $i \leftarrow t_y \times s_x + t_x$ **to** $s_y \times m$ **step** $s_x \times s_y$ **do**

6: $j \leftarrow b_y \times s_y + (i \bmod s_y) + \frac{i}{s_y} \times \rho$

7: $\ddot{B}_i^u \leftarrow B_j^u$ ▷ *Load into shared memory*

8: **end for**

▷ *This loop transfers needed basis (v-direction) elements to shared memory*

9: **for** $i \leftarrow t_y \times s_x + t_x$ **to** $s_x \times n$ **step** $s_x \times s_y$ **do**

10: $j \leftarrow b_y \times s_x + (i \bmod s_x) + \frac{i}{s_x} \times \delta$

11: $\ddot{B}_i^v \leftarrow B_j^v$ ▷ *Load into shared memory*

12: **end for**

▷ *Synchronization of threads within block*

13: intra-block_synchronization()

14: $a \leftarrow b_x \times s_y + t_x$ ▷ *Thread-index of output item (x-coordinate)*

15: $b \leftarrow b_y \times s_x + t_y$ ▷ *Thread-index of output item (y-coordinate)*

▷ *Evaluation of the corresponding surface point*

16: **if** $a < \rho$ and $b < \delta$ **then** ▷ *If thread index is within surface*

17: $\mathbf{q} \leftarrow 0$

18: **for** $k_i \leftarrow 0$ **to** m **do**

19: $b_i \leftarrow \ddot{B}^u_{t_x + k_i \times s_y}$

20: **for** $k_j \leftarrow 0$ **to** n **do**

21: $b_j \leftarrow \ddot{B}^v_{t_x + k_j \times s_x}$

22: $\mathbf{q} \leftarrow \mathbf{q} + \ddot{\mathbf{P}}_{k_i + n + k_j} \times b_i \times b_j$

23: **end for**

24: **end for**

25: $\mathbf{S}_{a \times \delta + b} \leftarrow \mathbf{q}$

26: **end if**

27: **return S**

28: **end function**

Algorithm 8 Bernstein basis GPU kernel (*level 2*).

Note: s_x, s_y block sizes in the x and y dimensions. t_x, t_y thread indexes in x and y dimensions.

1: **function** BernsteinBasisGPU($m, n, \mathbf{C^u}, \mathbf{C^v}, \rho, \delta$)
2: $x \leftarrow s_x b_y + t_x$
3: **if** $x < (m+1)\rho$ **then**
4: $i \leftarrow x \bmod \rho$
5: $j \leftarrow \frac{x}{\rho}$
6: $\mu \leftarrow \frac{i}{(\rho-1)}$
7: $U_{i+j\times\rho} \leftarrow C_j^u \times \mu^j \times (1-\mu)^{(m-j)}$
8: **end if**
9: **if** $m \neq n$ or $\rho \neq \delta$ **then** ▷ *Isotropy check*
10: $x \leftarrow s_x b_x + t_x$
11: **if** $x < (nx+1)\delta$ **then**
12: $i \leftarrow x \bmod \delta$
13: $j \leftarrow \frac{x}{\delta}$
14: $\mu \leftarrow \frac{i}{(\delta-1)}$
15: $V_{i+j\times\delta} \leftarrow C_j^u \times \mu^j \times (1-\mu)^{(m-j)}$
16: **end if**
17: **else**
18: $V_{i+j\times\delta} \leftarrow U_{i+j\times\delta}$ ▷ *Copy/Direct-addressing* on isotropy
19: **end if**
20: **return** $[\mathbf{U}, \mathbf{V}]$
21: **end function**

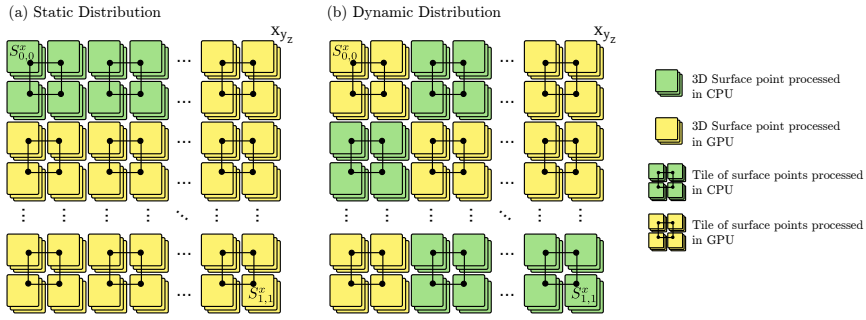


Figure 8.4: Heterogeneous parallel computing for a grid of 2×2 tiles of 3D surface elements. (a) SDC where, the first n blocks are statically assigned to the CPU and the remaining blocks are assigned to the GPU. (b) DDC where, blocks are assigned *on-the-fly* (in run-time) to the CPU and the GPU as they finish processing other blocks and become available.

Two schemes for distributing the computational burden between CPU and GPU are possible depending on the characteristics of the underlying hardware.

Scheme 1: static distribution of computation (**SDC**). In absence of memory coherence, dynamic communication between CPU and GPU is not possible. Translated to our problem, this means that in run-time none of the processors can verify which part of the 3D surface has already been computed by other processors with concurrency condition guarantees. Despite of that, it is still possible to statically assign the amount of elements that should be processed by the CPU and the GPU. As shown in Figure 8.4a, the output surface space is divided into tiles containing neighboring 3D surface points. Each tile is then statically assigned to either a CPU thread or a GPU block. In highly regular problems, the number of tiles processed by the CPU is significantly lower than the number of tiles processed by the GPU. If only *level 1* is computed, offline profiling can help to find a fair workload distribution thanks to the regularity of computations.

Scheme 2: dynamic distribution of computation (**DDC**). Under memory coherence conditions such as HSA platforms (Figure 8.4b), processors assume the computation of non-processed tiles available in a list. This list can be concurrently accessed by CPU and GPU cores through the use of cross-device atomic operations. As opposed to SDC, the assignment of tiles to the CPU threads and the GPU blocks is not known beforehand.

8.5.3 Rendering and graphics interoperability

As previously mentioned, computer-graphics applications and in particular tessellation are the most prominent fields of application for Bézier surfaces. The mechanism by which GPUs can utilize and coordinate computing and graphics capabilities is known as *graphics interoperability*. This mechanism consists of a common buffer of vertices known as *vertex buffer object* (**VBO**) which is shared by the evaluation of the surface (CUDA/OpenCL) and the rendering. The use of graphics interoperability avoids large data transfers between the CPU and the GPU.

In order to test the performance of MLE in rendering applications, we have implemented an event-driven renderer which employs OpenGL and GLUT [46], together with CUDA graphics interoperability. With the aim to keep simplicity and generality of results, our renderer only considers geometry/topology rendering without coloring and illumination which are not present in every application. The resolution of the output is set to high definition (1920×1080) regardless of the underlying hardware. The reader should note that event-driven always present some degree of CPU computing handle the events. This type of renderers is closer to applications where interactions that change the properties of the surface occur (e.g., computer-aided design applications or computer games). Higher performance can be achieved using dedicated engines avoiding events (e.g, animation rendering).

8.6 Performance evaluation and results

In this section, we present the evaluation setup and the results obtained by the proposed method and its parallel mapping onto different hardware platforms, including: one CPU; two discrete GPUs running our CUDA (for NVIDIA) implementation; one mobile integrated GPU (NVIDIA Jetson TK1) running our CUDA implementation; and one integrated GPU (AMD) running our OpenCL implementation. A summary of the employed architectures and the associated implementations are shown in Table 8.2.

The presented results are based on the comparison of the proposed approach (MLE) with: 1) a *"brute force"* (**BRF**) iterative approach which computes Equation (8.1) and all its elements, including those in Equation (8.2) for every evaluation cycle (every frame); and 2) the matrix form in Equation 8.4 (**MAT**) employed by [25]. The reader should note that the latter approach can be seen as an optimized formulation of the matrix form in Equation (8.3) previously employed by [7].

Performance evaluation in multi-threading (CPUs) often leads to significant variation of results [47, 48] due to memory access mechanisms and operating system

Table 8.2: GPU/CPU architectures employed for the evaluation of Bézier surfaces in this work.

Device	Codename	Type	Year	Implementation
Intel® Core™ i7 930 2.80GHz	Nehalem	CPU	2008	OpenMP
NVIDIA® GTX™ 460	Fermi	Discrete GPU	2010	CUDA
NVIDIA® GTX™ 980	Maxwell	Discrete GPU	2014	CUDA
NVIDIA® Jetson™ TK1	Logan/Kepler	Integrated GPU*	2014	CUDA/OpenMP
AMD® A10-7850K	Kaveri	Integrated GPU**	2014	OpenCL

** Heterogeneous mobile architecture. 4-Core ARM Cortex-A15 CPU + Kepler GPU (192 CUDA cores).*

*** Heterogeneous architecture 4-core AMD Steamroller CPU + R7 GPU with 8 compute units.*

scheduling policies. This poses a challenge for benchmarking. In order to reduce the variability of results, our implementations consist of 10 warm-up evaluations followed by 10 measured evaluations which are then averaged. This process is repeated 10 times, providing 10 averaged samples from which the observations exceeding the mean value plus 1.96 standard deviations (95% confidence interval) are removed. After removal of outliers, the resulting observations are used to calculate the mean execution time.

Some of our results are expressed in terms of performance increments, measured by a function f in frames per second (**FPS**). Hence, $\Delta\mathbf{MAT} = f(MAT) - f(BRF)$ represents the difference between MAT and BRF, and $\Delta\mathbf{MLE} = f(MLE) - f(MAT)$ is the difference between MLE (in *level 1*) and MAT. The results include evaluation (in our results *evaluation*), as well as evaluation followed by a rendering stage (in our results *evaluation+rendering*) through the graphics interoperability mechanisms explained.

In order to widen the applicability of the results, the evaluation takes into consideration the use of single-precision (`float`), as well as double-precision (`double`) data types. For many applications this is an important consideration which may have an impact on the performance, precision and numerical stability. Computer-graphics applications, for instance, are mostly concerned about performance and often use single-precision as the data-type of choice; in simulation applications on the other hand, precision and numerical stability are of paramount importance, and hence, a double-precision data-type is preferred.

8.6.1 Evaluation on CPU

The evaluation on CPU is performed using a quad-core *Intel® Core™ i7 930 2.80GHz* processor (64-bits architecture) with *Hyper-Threading* technology enabled, thus providing up to 8 virtual cores. The parallel implementation of all the methods evaluated is obtained by means of OpenMP pre-processor directives. The degree

of parallelization ranges from 1 to 8 CPU threads.

In a first experiment, the number of control points and the resolution are arranged so that the size of the Bernstein basis arrays meet favorable memory alignment conditions (4×4 control points and 256×256 resolution). Then, evaluations using different methods and different numbers of threads are performed both with and without rendering. For double-precision, the results (Figure 8.5a) show a notable performance improvement of MLE over both MAT (2.41x to 3.12x speedup) and BRF (18.98x to 25.47x speedup). The maximum performance is reached by using 4 CPU threads in *evaluation* (1149 FPS), and 8 CPU threads in *evaluation+rendering* (254 FPS). The use of single-precision favors the performance of *evaluation+rendering* (1.62x to 2.82x speedup) over *evaluation*, where there is no significant difference in performance. This is mainly due to the reduction of the CPU-GPU data transfer. Our results show adequate scalability of performance in *evaluation*, this is, a linear increase of performance as the number of threads increases (both for single and double-precision); this behavior holds separately for 1-4 cores (physical cores) and 5-8 (*Hyper-threading*). Similarly, for *evaluation+rendering* linear scaling of results is observed; for double precision, however, linear scaling does not hold for 5-8 threads (*Hyper-threading*) due to memory transfers.

In a second experiment, the number of CPU threads is fixed to 4 and evaluations are performed as combinations of a variable number of control points (4×4, 8×8 and 12×12) with variable surface resolutions (256×256, 384×384 and 512×512). As in the first experiment, for double-precision, MLE exhibits higher performance than both MAT (2.18x to 3.15x speedup) and BRF (24.97x to 72.89x speedup). The speedup of MLE over MAT and BRF diminishes as the degree of the surfaces increases—which is further discussed in Section 8.7. For CPUs, including rendering stages not only implies more operations to perform, but also CPU-GPU memory transfers. Thus, the performance of *evaluation+rendering* is within the range 24.4 to 261.8 FPS for double-precision. As in the first experiment, the use of single-precision can improve the performance significantly (1.02x to 2.50x speedup). Scalability of results adheres to the size ($\rho \times \delta$) and degree ($m \times n$) in a linear manner for both parameters according to $\mathcal{O}(\rho \times \delta \times m \times n)$ described in Section 4; this phenomenon can be easily observed in Figure 8.5c,d where a quadratic increase in either the size or the degree of the surface produces a quadratic performance decrease.

8.6.2 Evaluation on GPUs

In this section, we test the CUDA implementation of the different methods on two of the three most recent NVIDIA architectures (*Maxwell* and *Fermi*). More

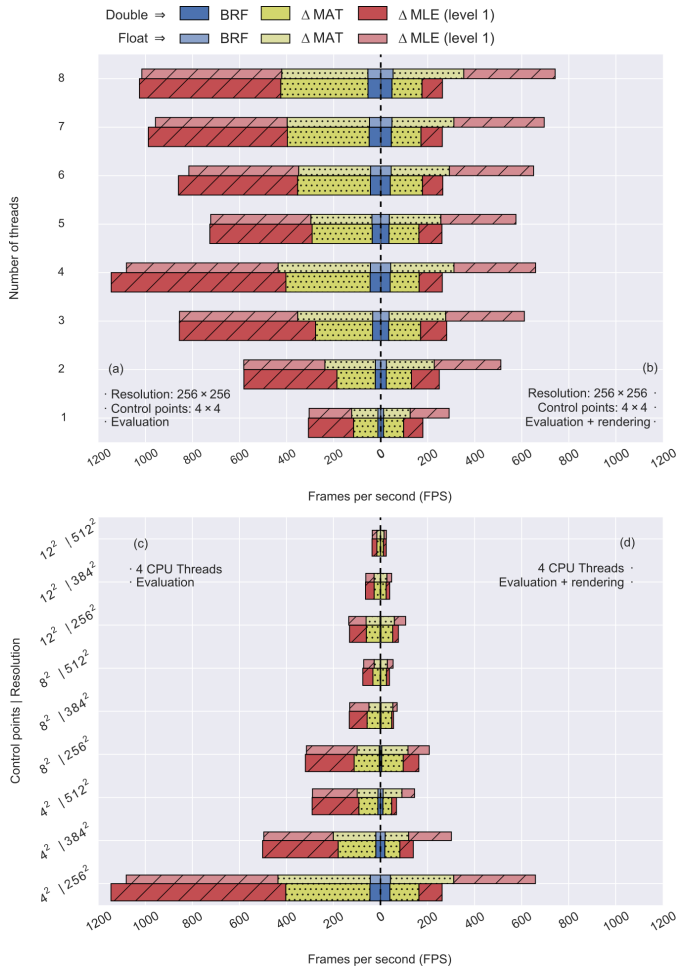


Figure 8.5: Parallel evaluation of bi-cubic Bézier surfaces in *Intel® Core™ i7 CPU 930 2.80GHz*. (a) and (b) compare computation of the same surface using different number of CPU threads. (c) and (d) compare computation of surfaces of variable resolution and number of control points using 4 CPU threads.

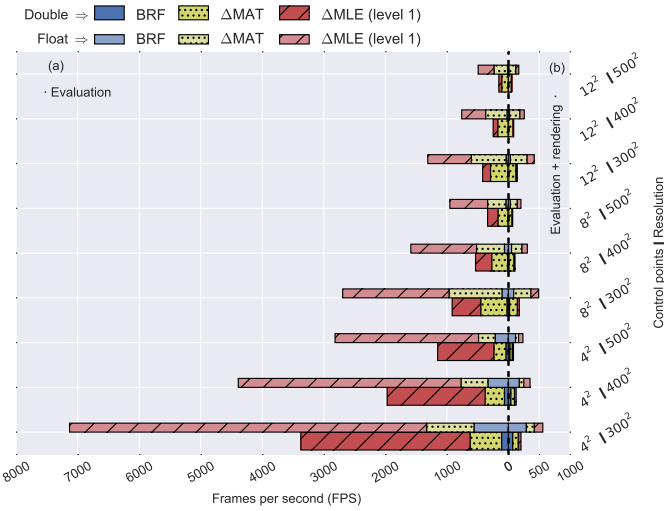


Figure 8.6: Parallel evaluation and rendering of Bézier surfaces (variable control points and resolution) in *NVIDIA® GTX™ 460* (*Fermi* architecture).

precisely, the results are obtained from a *NVIDIA® GTX™ 980 4GB* and a *NVIDIA® GTX™ 460 1GB*, both over a PCIe 2.0 bus. Additionally, we present results on a mobile GPU (*NVIDIA® Jetson™ TK1*). The geometry (size of blocks) of the kernel functions is 16×16 GPU threads, which showed slightly better performance than 8×8 and 32×32 GPU threads, thanks to a higher occupancy value (number of active threads per GPU core).

In a first experiment, the performance of the methods on the older architecture (*GTX 460*) is evaluated. In the case of CPUs, surfaces with variable number of control points (4×4, 8×8 and 12×12) and resolution (300×300, 400×400 and 500×500) for evaluation and rendering. As shown in Figure 8.6, for double-precision, MLE obtains a significant performance improvement over both MAT (1.43x to 5.42x speedup) and BRF (28.63x to 49.20x speedup). For *evaluation+rendering*, the performance varies between 57 FPS and 206 FPS under double-precision. The use of single-precision favors both *evaluation+rendering* (2.11x to 3.17x speedup) and *evaluation* (2.84x to 2.98x speedup) in a similar manner.

For the most recent architecture (*GTX 980*), the experiment consists of the evaluation and rendering of high-resolution surfaces. The complexity of the surfaces combines variable number of control points (4×4, 8×8, and 12×12) with a variable resolution (500×500, 1000×1000, and 2000×2000). The results, in Figure 8.7, show a performance improvement of MLE over both MAT (1.33x to 3.69x

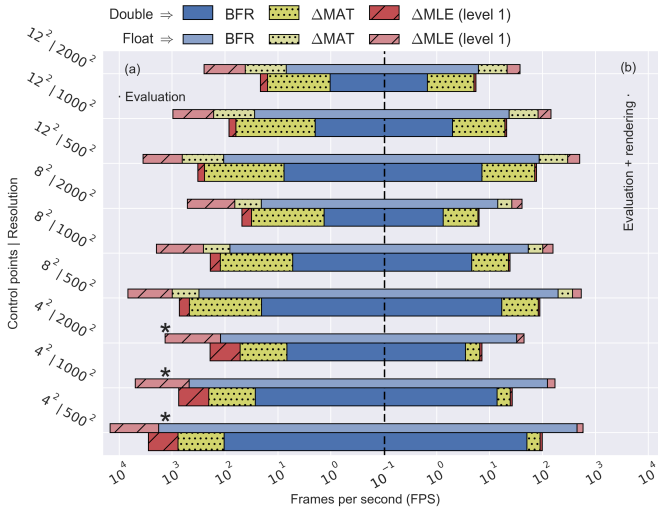


Figure 8.7: Parallel evaluation and rendering of Bézier surfaces (variable control points and resolution) in *NVIDIA® GTX™ 980* (*Maxwell* architecture). Results in logarithmic scale. * BRF and MAT show similar performance in both *evaluation* and *evaluation+rendering*.

speedup) and BRF (20.68x to 42.62x speedup). For *evaluation+rendering* in double precision, the performance varies between 5 FPS and 99 FPS depending on the complexity of the surface. The use of single-precision increases the performance dramatically (5.15x to 11.82x speedup for *evaluation*, and 5.62x to 7.32x speedup for *evaluation+rendering*). Following the same trend as in CPU (Section 8.6.1), the speedup of MLE over MAT diminishes as the degree of the surfaces increases.

The evaluation in mobile GPU was performed using a *NVIDIA® Jetson™ TK1*. For evaluation purposes, we set the hardware parameters to a high-performance profile (i.e., no CPU down-scaling and GPU frequency at 852MHz). As in previous evaluations, we conduct an experiment consisting o fthe execution of the methods with the complexity of the surfaces presenting variable number of control points (4×4, 8×8, and 12×12) combined with variable resolution (300×300, 400×400, and 500×500). Figure 8.8 shows the results in double-precision where *evaluation* on MLE outperforms MAT (1.67x to 4.68x speedup) and BRF (29.3x to 49.07x speedup). The performance of *evaluation+rendering* varies from 3.7 FPS to 11.5 FPS for double-precision. The use of single-precision over double-precision increases the performance notably (3.72x to 4.80x speedup for *evaluation* and 9.71x to 18.09x for *evaluation+rendering*).

Similarly to our performance results in CPU, performance results in GPU adhere to the linear decrease of performance established by $\mathcal{O}(\rho \times \delta \times m \times n)$ in Section 4.

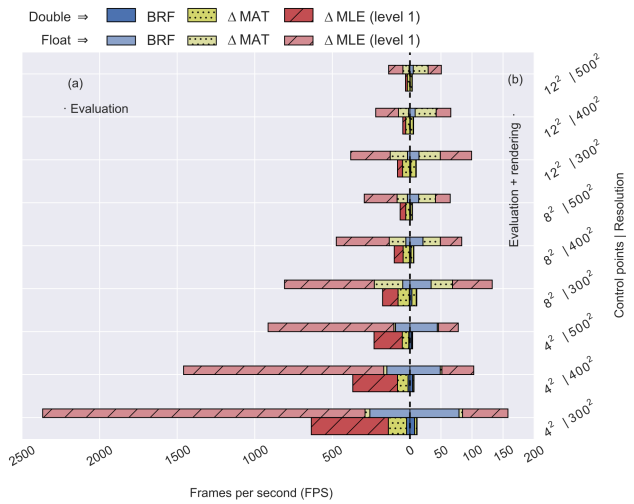


Figure 8.8: Parallel evaluation and rendering of single-precision Bézier surfaces (variable control points and resolution) in *NVIDIA® Jetson™ TK1* (*Kepler* architecture).

8.6.3 Evaluation on HCSs

Experiments in this section were designed to demonstrate how the CPU-GPU cooperation can improve the performance of the evaluation of Bézier surfaces in HCSs. Rendering is not subject to cooperation since this is a task carried out by solely the GPU.

For the SDC scheme, we consider the multi-level evaluation approach with predefined distribution of computation on a *NVIDIA® Jetson™ TK1* (no memory coherence). In order to obtain the optimal computation distribution, surfaces presenting a different number of control points (4×4, 8×8, and 12×12) and different resolutions (300×300, 400×400, and 500×500) were evaluated using MLE. For each of these evaluations we assign a CPU load from 0% to 100% in steps of 5%, which determines how many tiles the CPU (and therefore the GPU) will process. We employ 4 CPU threads in order to use the four available cores. The results, in Figure 8.9, show how the performance increases when the CPU assumes 10% to 15% of the workload (1.07x to 1.22x speedup compared to a GPU-only approach). Such stable percentages of workload prove that offline profiling has the ability to ensure a reasonably good workload distribution regardless of the number of control points and resolution.

For the DDC scheme, MLE is executed on an AMD® Kaveri™ (HSA) under DDC. The use of different number of CPU threads (1, 2 and 4) was tested for a variable number of control points (4×4, 8×8, and 12×12) and different resolutions

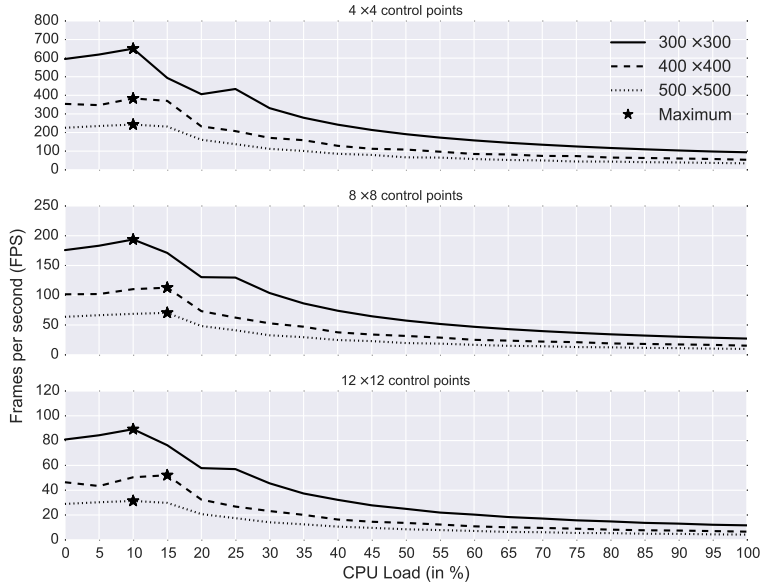


Figure 8.9: Parallel evaluation and rendering of Bézier surfaces in *NVIDIA® Jetson™ TK1* under a SDC strategy.

(300×300, 400×400, and 500×500). Figure 8.10 shows the benefit of introducing some degree of CPU-GPU cooperation. The positive impact of the CPU-GPU cooperation and the difference between the use of different number of threads diminish as the surface becomes more complex. This increase of performance of the best CPU-GPU cooperation was 1.03x to 2.09x speedup compared to the GPU-only approach. In order to compare performance results in different configurations (i.e. 1,2 or 4 CPU threads cooperating with GPU) careful consideration should be paid to hardware-specific aspects such that power-saving policies; for instance, low complexity surfaces in Figure 8.10 show higher performance for the use of 1 and 2 CPU threads compared to 4 CPU threads. This phenomenon, produced by the power-saving policies of the hardware, is contrary to the expected behavior, which is produced for complex surfaces (Figure 8.10).

For hardware platforms allowing the use of both SDC and DDC approaches, DDC has greater potential for achieving optimal workload balance between GPU and CPU since the discrete steps on SDC profiling might only find sub-optimal workload balances. This effect is shown in Table 8.3 where the DDC *level1* computing time is lower than the SDC *level1* time. The use of separate memory spaces for CPU and GPU processing imposes two additional memory transfers: an initial CPU-to-GPU transfer the basis $\mathbf{B}^{\mathbf{u}}$ and $\mathbf{B}^{\mathbf{v}}$; and a final CPU-to-GPU transfer of

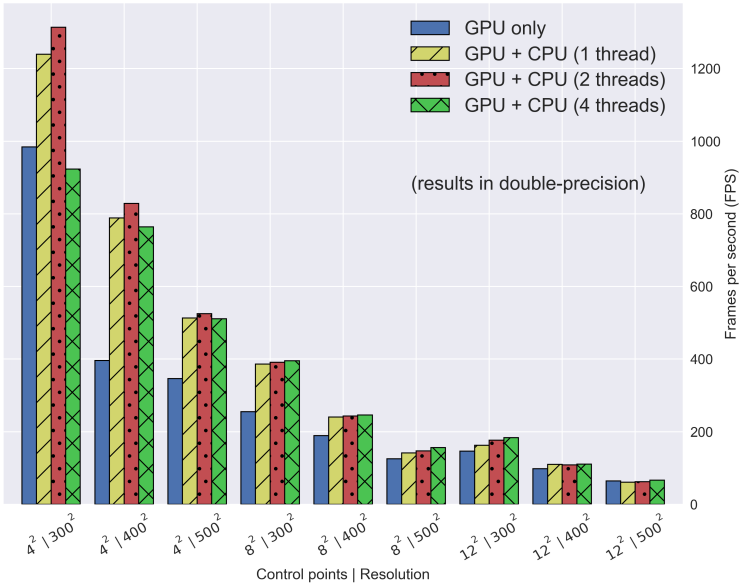


Figure 8.10: Parallel evaluation of Bézier surfaces in *AMD® Kaveri™* (HSA) using different MLE with different CPU-GPU cooperation degrees under a DDC strategy.

part of the surface to the memory holding the final result (in Table 8.3 final results are stored in GPU memory).

Theoretical order of complexity, regardless of the cooperation scheme, is preserved as described in Section 4, that is, linear decrease of performance is observed when either the degree or the size of the surface increases.

8.7 Discussion

The use of optimization techniques and parallel computing has been present in modern implementations utilizing Bézier constructions (not only surfaces). As shown in Table 8.1, the most prominent area of use of parallel computing and optimization techniques is computer-graphics. This explains the relatively low degree of the surfaces employed in the literature (since 3D mesh models can be composed of low-degree sets of Bézier patches). However, due to its interesting properties, Bézier constructions can be found in applications other than computer-graphics. New parallelization and optimization techniques, able to extend the application scope while keeping high-performance results in computer-graphics are, therefore, of great interest. In this line, we propose a method (MLE) and associated parallelization strategies to map the method onto different hardware platforms (CPUs, GPUs, mobile integrated GPUs and HCSs), thus covering a wide

Table 8.3: Time results (in ms) for heterogeneous approaches for surfaces of 400×400 and 8×8 control points in AMD® A10-7850K (Kaveri) including *level1* and *level2* computations, memory transfers and allocations.

OpenCL	Cooperation	Basis transfer	*level2*	*level1*	Surface transfer	Total
2.0*	CPU+GPU (DDC)	0.000	0.052	4.026	0.000	4.078
1.2	CPU+GPU (SDC)	0.293	0.052	4.631	0.278	5.253
2.0*	CPU+GPU (SDC)	0.000	0.054	4.603	0.000	4.657

* Allows the use of memory coherence for CPU-GPU cooperation in the same memory space.

spectrum of possible applications.

The idea behind the proposed method is to reduce the number of operations executed. In essence, our approach exploits re-utilization of data items, thus replacing computations by memory accesses. Compared to other methods like MAT in [25] (some degree of re-utilization) and BRT (no re-utilization), our approach shows a generalized increase of the performance. Using CPUs, the benefit can be as high as 3.12x speedup for double-precision MLE over MAT, and 25.47x speedup over BRF. For discrete GPUs the gain is even larger, making MLE up to 3.69x faster than MAT and up to 42.62x faster than BRF.

The trends observed in the results indicate that as the degree of the surface increases, the speedup of MLE over MAT reduces. This can be explained by looking at the number of operations performed by MAT (Equation 8.3) and MLE in *level 1* (Equation 8.6). Operationally, MAT needs to compute the basis $\mathbf{U}$ and $\mathbf{V}$, the products $\mathbf{UR}$ and $\mathbf{R}^T\mathbf{V}^T$, as well the final product with the matrix of control points $\mathbf{P}$. This imposes an overhead with respect to MLE which only needs to perform a matrix multiplication. In MAT, the size of $\mathbf{U}$ and $\mathbf{V}$ arrays of basis grows linearly with the number of control points while the size of the matrices grows quadratically. Therefore, the overhead takes a larger fraction of the total time for low-degree surfaces than for high-degree surfaces.

The use of single-precision vs. double-precision arises as a very important question since our aim is to cover a wide range of applications. Our results reveal a generalized performance gain, that can be as high 11.82x speedup, on using single-precision over double-precision in high-end discrete GPUs (*GTX 980*). Computer-graphics applications, which do not require double-precision are clear targets for choosing single-precision. For scientific applications requiring double-precision (e.g., simulations), the best performance can be achieved by the scientific-class GPU (e.g., *NVIDIA® Tesla™ K20*).

New trends in computing, like HCSs, can leverage higher performance by cooperation mechanisms between processors. In the two strategies for cooperation between processors (SDC and DDC), a speedup as high as 2.09x is observed using DDC, while with the use of SDC a maximum of 1.22x speedup is obtained. The use of SDC and DDC is determined by whether the system possesses memory coherence mechanisms and cross-device atomic operations. For hardware platforms allowing both SDC and DDC, DDC has greater potential to achieve optimal workload balance across processors.

In mobile computing, real-time evaluation and rendering of Bézier surfaces has been considered a difficult task [7]. Our results show that modern mobile processors including an integrated GPU (like *NVIDIA® Jetson™ TK1*), together with parallelization and CPU-GPU cooperation can achieve real-time performance (650.93 FPS in double-precision and 2366 FPS in single-precision) for relatively complex surfaces (bi-cubic at resolution 300×300). In the absence of rendering, the computation of Bézier surfaces can reach higher performance while presenting better hardware integration possibilities than modern CPUs. In respect to rendering (in which case single-precision is advised), performance can be as high as 157.91 FPS.

With the aim of providing other researchers with a broad coverage of applications, our results include evaluations far beyond the limits of other works found in the literature (Table 8.1). This, together with the broad set of hardware architectures evaluated, can be used as a guide to establish the limits and scalability of the use of Bézier surfaces in a wide variety of applications including high-degree and high-resolution Bézier surfaces. MLE and the associated strategies proposed, are also easily generalizable to higher-order Bézier construction and other Bézier formulations (i.e., rational Bézier).

8.8 Conclusion

In this work, we present a new method (MLE) and the use of different parallel computing techniques to accelerate the computation of Bézier tensor-product surfaces in different hardware platforms (CPUs, discrete GPUs, integrated GPUs, mobile GPUs and HCSs). In line with the latest trends in hybrid computing, we also propose two CPU-GPU cooperation strategies (SDC and DDC) to be exploited by HCS platforms. Our results—which include an exhaustive evaluation using different data-types, different degrees and resolution of surfaces and different computing platforms—show that our method achieves speedups as high as 3.12x (double-precision) and 2.47x (single-precision) on CPU compared to other proposals found in the literature. In GPU computing, the speedup is as high as 3.69x (double-precision) and 13.14x (single-precision). CPU-GPU cooperation strategies

employed in this work pose a clear benefit increasing the performance up to 2.09x with respect to GPU-only approaches. MLE, as well as the parallelization and CPU-GPU cooperation strategies are easily generalizable to high-order Bézier constructions (e.g., volumes in medical imaging) and other Bézier formulations (i.e., rational Bézier).

Bibliography

[1] SAA Karim and Azizan Saaban. Bézier Triangular Patches for Closed Surface. *Applied Mathematical Sciences*, 8(8):355–366, 2014.

[2] Huogen Yang and Guozhao Wang. Optimized design of Bézier surface through Bézier geodesic quadrilateral. *Journal of Computational and Applied Mathematics*, 273:264–273, 2015. ISSN 03770427. doi: 10.1016/j.cam.2014.06.017.

[3] G Akemi. Bezier Curve and Surface Fitting of 3D Point Clouds Through Genetic Algorithms , Functional Networks and Least-Squares Approximation. In *Computational Science and Its Applications–ICCSA 2007*, volume 4706, pages 680–693, 2007.

[4] C U I Yuan-min. Real-time accurate Free-Form Deformation in terms of triangular Bézier surfaces. *Applied Mathematics*, 29(4):455–467, 2014.

[5] L. Hilario, A. Falcó, N. Montés, and M.C. Mora. A tensor optimization algorithm for Bézier Shape Deformation. *Journal of Computational and Applied Mathematics*, 291:264–280, 2015. ISSN 03770427. doi: 10.1016/j.cam.2015.02.035.

[6] Raquel Concheiro, Margarita Amor, Emilio J. Padrón, and Michael Doggett. Interactive rendering of NURBS surfaces. *CAD Computer Aided Design*, 56:34–44, 2014. ISSN 00104485. doi: 10.1016/j.cad.2014.06.005.

[7] Raquel Concheiro, Margarita Amor, Emilio J Padrón, Marisa Gil, and Xavier Martorell. Rendering of Bézier Surfaces on Handheld Devices. In *21st International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG 2013)*, 2013.

[8] Michael Schwarz and Marc Stamminger. Fast GPU-based adaptive tessellation with CUDA. *Computer Graphics Forum*, 28(2):365–374, 2009. ISSN 01677055. doi: 10.1111/j.1467-8659.2009.01376.x.

[9] Kathleen Gilbert, Genevieve Farrar, Brett R Cowan, Avan Suinesiaputra, Christopher Occleshaw, Beau Pontre, James Perry, Sanjeet Hegde, Alison

Marsden, Jeff Omens, and Others. Creating shape templates for patient specific biventricular modeling in congenital heart disease. In *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*, pages 679–682. IEEE, 2015.

[10] Suqin Zeng and Elaine Cohen. Hybrid volume completion with higher-order Bézier elements. *Computer Aided Geometric Design*, 35-36:180–191, 2015. ISSN 01678396. doi: 10.1016/j.cagd.2015.03.008.

[11] Grzegorz Soza, Michael Bauer, Peter Hastreiter, Christopher Nimsky, and Günther Greiner. Non-rigid Registration with Use of Hardware-Based 3D Bezier Functions. *Proceedings of the 5th International Conference on Medical Image Computing and Computer-Assisted Intervention-Part II*, pages 549–556, 2002. ISSN 16113349. doi: 10.1007/3-540-45787-9_69.

[12] Bo Li, Alistair A Young, and Brett R Cowan. GPU accelerated non-rigid registration for the evaluation of cardiac function. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2008*, 11(Pt 2):880–887, 2008.

[13] Rui Wang, Xianjin Yang, Yazhen Yuan, Wei Chen, Kavita Bala, and Hujun Bao. Automatic Shader Simplification Using Surface Signal Approximation. *ACM Trans. Graph.*, 33(6):226:1–226:11, 2014. ISSN 0730-0301. doi: 10.1145/2661229.2661276.

[14] Michele Andreoli, Janka Ales, and Jean-antoine Desideri. Free-form-deformation parameterization for multilevel 3D shape optimization in aerodynamics. Technical report, INRIA, FR, 2006.

[15] Andrea Manzoni, Alfio Quarteroni, and Gianluigi Rozza. Shape optimization for viscous flows by reduced basis methods and free-form deformation. *International Journal for Numerical Methods in Fluids*, 70(5):646–670, 2012. ISSN 1097-0363. doi: 10.1002/fld.2712.

[16] Hugo Casquero, Lei Liu, Carles Bona-Casas, Yongjie Zhang, and Hector Gomez. A hybrid variational-collocation immersed method for fluid-structure interaction using unstructured T-splines. *International Journal for Numerical Methods in Engineering*, 2015.

[17] Georgios Georgis, George Lentaris, and Dionysios Reisis. Acceleration techniques and evaluation on multi-core CPU, GPU and FPGA for image processing and super-resolution. *Journal of Real-Time Image Processing*, (1):1–28, 2016. ISSN 18618200. doi: 10.1007/s11554-016-0619-6.

[18] Jesús Jiménez and J de Miras. Three-dimensional thinning algorithms on graphics processing units and multicore CPUs. *Concurrency and Computation: Practice and Experience*, 24(14):1551–1571, 2012.

[19] Weixin Luo, Xuan Yang, Xiaoxiao Nan, and Bingfeng Hu. GPU accelerated 3d image deformation using thin-plate splines. *Proceedings - 16th IEEE International Conference on High Performance Computing and Communications, HPCC 2014, 11th IEEE International Conference on Embedded Software and Systems, ICESS 2014 and 6th International Symposium on Cyberspace Safety and Security*, pages 1142–1149, 2014. doi: 10.1109/HPCC.2014.168.

[20] Joel Hestness, Stephen W. Keckler, and David A. Wood. GPU computing pipeline inefficiencies and optimization opportunities in heterogeneous CPU-GPU processors. *Proceedings - 2015 IEEE International Symposium on Workload Characterization, IISWC 2015*, pages 87–97, 2015. doi: 10.1109/IISWC.2015.15.

[21] Antonio Vilches, Angeles Navarro, Rafael Asenjo, Francisco Corbera, Rubén Gran, and María J. Garzarán. Mapping streaming applications on commodity multi-CPU and GPU on-chip processors. *IEEE Transactions on Parallel and Distributed Systems*, 27(4):1099–1115, 2016. ISSN 10459219. doi: 10.1109/TPDS.2015.2432809.

[22] Yifan Sun, Xiang Gong, Amir Kavyan Ziabari, Leiming Yu, Xiangyu Li, Saoni Mukherjee, Carter McCardwell, Alejandro Villegas, and David Kaeli. Hetero-mark, a benchmark suite for CPU-GPU collaborative computing. In *Proceedings of the 2016 IEEE International Symposium on Workload Characterization, IISWC 2016*, pages 13–22, 2016. ISBN 9781509038954. doi: 10.1109/IISWC.2016.7581262.

[23] Juan Gómez-Luna, Izzat El Hajj, Victor Chang Li-Wen Garcia-Flores, Simon de Gonzalo, Thomas Jablin, Antonio J Pena, and Wen-mei Hwu. Chai: Collaborative Heterogeneous Applications for Integrated-architectures. In *Performance Analysis of Systems and Software (ISPASS), 2017 IEEE International Symposium on*. IEEE, 2017.

[24] Les Piegl and Wayne Tiller. *The NURBS Book (2Nd Ed.)*. Springer-Verlag New York, Inc., New York, NY, USA, 1997. ISBN 3-540-61545-8.

[25] Raquel Concheiro, Margarita Amor, Montserrat Bóo, and Emilio J Padron. Free adaptive tessellation strategy of Bézier surfaces. In *Computer Graphics Theory and Applications (GRAPP), 2014 International Conference on*, pages 1–9. IEEE, 2014.

[26] FJ Espino, M Bóo, M Amor, and JD Bruguera. Adaptive tessellation of NURBS surfaces. *Journal of WSCG*, 11(1-3), 2003.

[27] F. J. Espino, M. Bóo, M. Amor, and J. D. Bruguera. Hardware support for adaptive tessellation of Bézier surfaces based on local tests. *Journal of Systems Architecture*, 53(4):233–250, 2007. ISSN 13837621. doi: 10.1016/j.sysarc.2006.10.011.

[28] Michael Guthe, Aákos Balázs, and Reinhard Klein. GPU-based trimming and tessellation of NURBS and T-Spline surfaces. *ACM Transactions on Graphics*, 24(3):1016, 2005. ISSN 07300301. doi: 10.1145/1073204.1073305.

[29] Charles Loop and Jim Blinn. Real-time GPU rendering of piecewise algebraic surfaces. *ACM Transactions on Graphics*, 25(3):664, 2006. ISSN 07300301. doi: 10.1145/1141911.1141939.

[30] R Concheiro and M Amor. Synthesis of Bézier Surfaces on the GPU. In *International Conference on Computer Graphics Theory and Applications*, pages 110–115, Angers, France, 2010.

[31] NVIDIA. NVIDIA CUDA C Programming Guide, 2008.

[32] AMD. Introduction to OpenCL Programming, 2010.

[33] C. Dyken, M. Reimers, and J. Seland. Real-time GPU silhouette refinement using adaptively blended Bézier patches. *Computer Graphics Forum*, 27(1):1–12, 2008. ISSN 01677055. doi: 10.1111/j.1467-8659.2007.01030.x.

[34] Christian Eisenacher, Quirin Meyer, and Charles Loop. Real-time view-dependent rendering of parametric surfaces. *Proceedings of the 2009 symposium on Interactive 3D graphics and games - I3D '09*, page 137, 2009. doi: 10.1145/1507149.1507172.

[35] Iddo Hanniel, Adarsh Krishnamurthy, and Sara McMains. Computing the Hausdorff distance between NURBS surfaces using numerical iteration on the GPU. *Graphical Models*, 74(4):255–264, 2012. ISSN 15240703. doi: 10.1016/j.gmod.2012.05.002.

[36] Yuanmin Cui and Jieqing Feng. Real-time B-spline Free-Form Deformation via GPU acceleration. *Computers and Graphics (Pergamon)*, 37(1-2):1–11, 2013. ISSN 00978493. doi: 10.1016/j.cag.2012.12.001.

[37] Bob Wallis. Tutorial on forward differencing. In *Graphics gems*, pages 594–603. Academic Press Professional, Inc., 1990.

[38] Adarsh Krishnamurthy, Rahul Khardekar, and Sara McMains. Optimized GPU evaluation of arbitrary degree NURBS curves and surfaces. *Computer-Aided Design*, 41(12):971–980, 2009. ISSN 00104485. doi: 10.1016/j.cad.2009.06.015.

[39] OpenMP. OpenMP Application Program Interface, 2008.

[40] John A Stratton, Christopher Rodrigues, I-Jui Sung, Li-Wen Chang, Nasser Anssari, Geng Liu, Wen-mei W Hwu, and Nady Obeid. Algorithm and Data Optimization Techniques for Scaling to Massively Threaded Systems. *Computer*, 45(8):26–32, 2012. ISSN 0018-9162. doi: http://doi.ieeecomputersociety.org/10.1109/MC.2012.194.

[41] Andre R. Brodtkorb, Christopher Dyken, Trond R. Hagen, Jon M. Hjelmervik, and Olaf O. Storaasli. State-of-the-art in heterogeneous computing. *Scientific Programming*, 18(1):1–33, 2010. ISSN 10589244. doi: 10.3233/SPR-2009-0296.

[42] George Kyriazis. Heterogeneous System Architecture : A Technical Review. Technical report, AMD, 2012.

[43] B A Hechtman, Shuai Che, D R Hower, Yingying Tian, B M Beckmann, M D Hill, S K Reinhardt, and D A Wood. QuickRelease: A throughput-oriented approach to release consistency on GPUs. In *High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on*, pages 189–200, feb 2014. doi: 10.1109/HPCA.2014.6835930.

[44] M Gupta, D Das, P Raghavendra, T Tye, L Lobachev, A Agarwal, and R Hegde. Implementing Cross-Device Atomics in Heterogeneous Processors. In *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International*, pages 659–668, may 2015. doi: 10.1109/IPDPSW.2015.40.

[45] Chang Xu, Steven R. Kirk, and Samantha Jenkins. Tiling for Performance Tuning on Different Models of GPUs. *2009 Second International Symposium on Information Science and Engineering*, pages 500–504, 2009. doi: 10.1109/ISISE.2009.60.

[46] Richard S Wright, Nicholas Haemel, Graham M Sellers, and Benjamin Lipchak. *OpenGL SuperBible: comprehensive tutorial and reference*. Pearson Education, 2010.

[47] Nicholas J. Wright, Shava Smallen, Catherine Mills Olschanowsky, Jim Hayes, and Allan Snavely. Measuring and Understanding Variation in Benchmark Performance. In *2009 DoD High Performance Computing Modernization Program Users Group Conference*, pages 438–443, 2009. ISBN 978-1-4244-5768-7. doi: 10.1109/HPCMP-UGC.2009.72.

[48] Kishore Kumar Pusukuri, Rajiv Gupta, and Laxmi N Bhuyan. Thread tranquilizer: Dynamically reducing performance variation. *ACM Transactions on Architecture and Code Optimization (TACO)*, 8(4):46, 2012.

CHAPTER 9

PAPER IV: INTRA-OPERATIVE MODELING OF THE LEFT ATRIUM: A SIMULATION APPROACH USING POISSON SURFACE RECONSTRUCTION

Rafael Palomar · Faouzi A. Cheikh · Azeddine Beghdadhi · Ole J. Elle

Abstract Electroanatomic Mapping (EAM) is an important process in Radio-frequency Catheter Ablations. In EAM, sample points are collected from the patient's atrium during intervention. This process is subject to inaccuracies contributed by different sources (e.g. tissue deformations and tracking errors). Poisson Surface Reconstruction (PSR) has recently been applied for intra-operative modeling of the left atrium through highly-dense clouds of points extracted from intra-operative and pre-operative imaging. In this work, we study the application of PSR under low-density sampling conditions which occur in some clinical work-flows. For this study we propose a simulation framework that is employed to characterize PSR in terms of accuracy of reconstruction. Our results show that a median error as low as $2.28\ mm$ can be obtained for a maximum of 600 sampled points. These results indicate the feasibility of applying PSR for low-dense clouds of points.

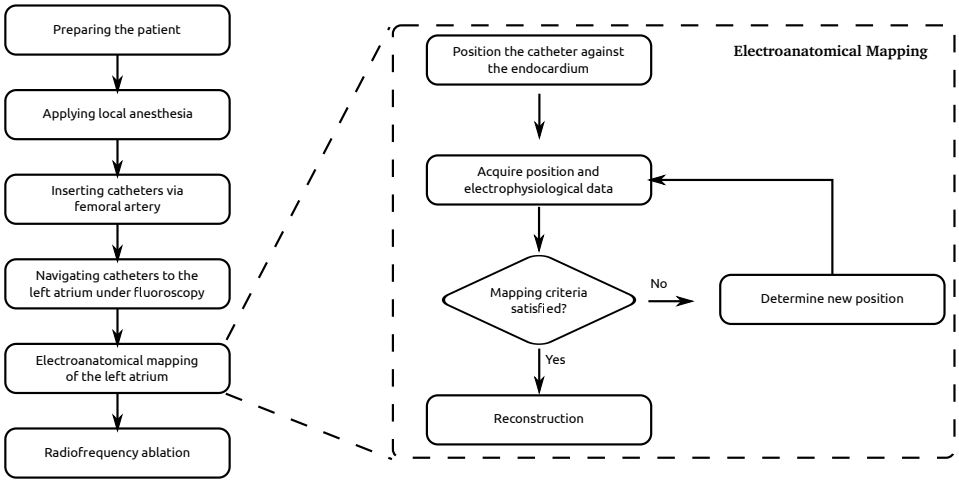


Figure 9.1: RFCA procedure: clinical work-flow (left) and electroanatomic mapping of the left atrium (right).

9.1 Introduction

Atrial Fibrillation (**AF**) is the most common heart rhythm disorder of clinical significance. Fibrillation of the aria is not only associated with a higher risk of stroke, but also contributes to heart failure and death. Around 8 million individuals in United States and Europe are affected by this condition—a higher life expectancy will increase this incidence up to more than 15 million individuals by 2050 [1]. Radio-frequency catheter ablation (**RFCA**), together with medication and cardioversion are the treatments of choice for AF.

RFCA (Fig. 9.1) is a minimally invasive surgical procedure generally occurring under local anesthesia. Ablation catheters are inserted into a femoral vessel and advanced to the left atrium under fluoroscopy guidance. Once in the atrium, sample points and electrophysiological data are acquired when the mapping catheter is in contact with the endocardium in a process known as electroanatomic mapping (**EAM**). The electrophysiological data associated with the geometry of the atrium is then reconstructed in a 3D model that will guide the ablation process. The aim of this process is the isolation of areas triggering atrial fibrillation, typically located around the pulmonary veins (**PV**s) ostia, by delivering energy through radio-frequency, thus causing ablation lesions.

EAM is an essential process during RFCA interventions. Several studies show the advantages of EAM compared to non-EAM guided approaches (e.g. [2]). 3D reconstructions from EAM are often merged with pre-operative models obtained

from magnetic resonance imaging (**MRI**) or computed tomography (**CT**). This provides electrophysiologists with visualization of great anatomical detail. However, factors like added cost, patient's discomfort and possible additional exposure to radiation, have to be considered in order to decide on the use of pre-operative images [3].

It is known that the EAM process is subject to errors contributed by different sources. Several works in the literature have assessed the accuracy of EAM [4, 5]. One source of errors is the underlying catheter tracking technology employed (usually based on electromagnetic tracking). Another source of errors is physiological factors like the heart motion caused by cardiac contractions and breathing [6].

Anatomical mapping of the left atrium is currently performed under two different strategies: *point-to-point* acquisition with triangulation-based reconstructions, and *progressive* reconstructions [7, 8]. Recently, *Poisson surface reconstruction* (**PSR**) [9] has been used as an alternative to obtain intra-operative patient-specific models of the left atrium from clouds of points [10, 11]. As opposed to the *point-to-point* acquisition approach, where a low density cloud of points is acquired, approaches based on PSR are supported on high-density clouds of points obtained either by intra-operative ultrasound or pre-operative models, which may not be available in some clinical work-flows.

9.1.1 Contribution

In this work, we characterize the use and accuracy of PSR for modeling of the left atrium using low-density clouds of points—as in the case of *point-to-point* electroanatomic mapping. To this end, we propose as simulation approach taking into consideration the impact of number of sample points, how these samples are distributed over the atrium chamber and inaccuracies introduced by *state-of-the-art* electromagnetic tracking.

9.2 Materials and methods

In this work, simulation of the EAM process is performed through sampling of left atria included in the CARMA pre-operative segmented left atria data-set[1] (57 atria) obtained from MRI. This data-set was acquired using a 1.5 Tesla and 3.0 Tesla scanners during contrast injection. The acquisition (transverse volume with voxel size $1.25\ mm \times 1.2\ 5\ mm \times 2.5\ mm$) was performed under free-breathing using navigator gating and then reconstructed to $0.625\ mm \times 0.625\ mm \times 1.25\ mm$ (3D inverse recovery GRE, $TR/TE = 5.4/2.3\ ms$).

[1] CARMA Left Atria MRI data-set available on (http://www.insight-journal.org/midas/collection/view/197)

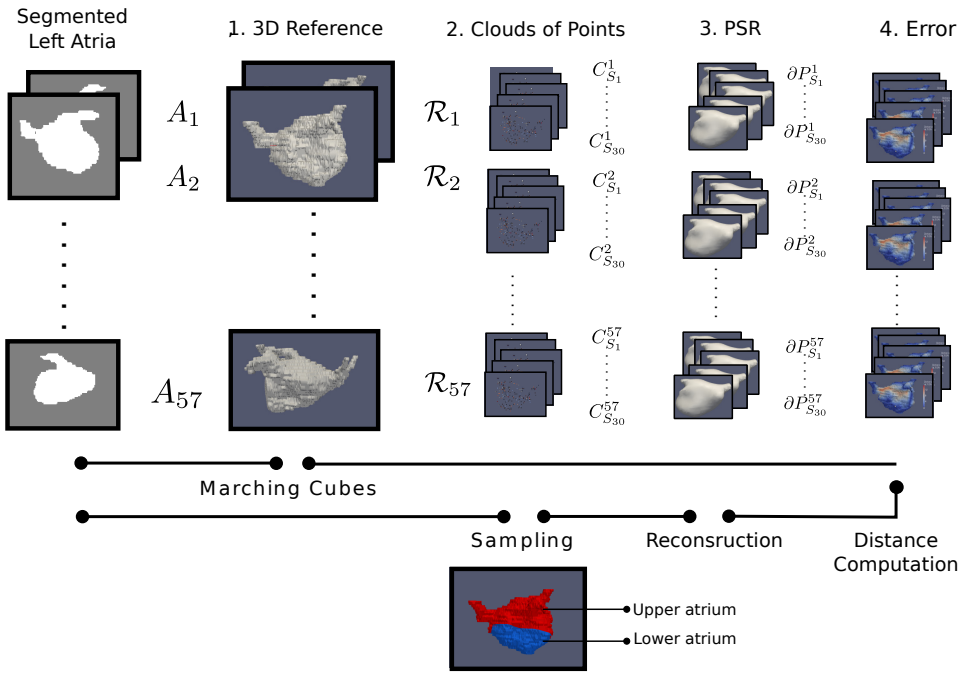


Figure 9.2: Proposed simulation process. First, a reference mesh models $\mathcal{R}_i$, and clouds of points $C^i_{S_j}$ are generated from the original segmented atria A_i. PSR models $\partial P^i_{S_j}$ are then generated from the clouds of points. Finally PSR errors are computed.

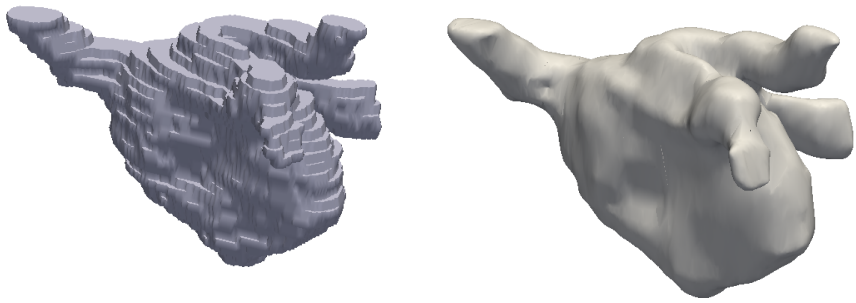
An evaluation framework (Fig. 9.2) consisting of several processing stages is applied. In this process:

1. A 3D reference set is generated from the original segmented data-set.
2. The reference set is sampled under different conditions.
3. The sampled set is reconstructed using PSR under different parameters.
4. The reconstructed set is compared to the reference. Reconstruction error measurements are derived from this comparison.

9.2.1 Reference set generation

Given the set $\mathbf{A} = \{A_1, A_2, ..., A_{57}\}$ of segmented atria, a set of reference tetrahedral meshes $\mathbf{R} = \{\mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_{57}\}$ is obtained from the application of *Marching Cubes*.

MRI acquisition presents anisotropic resolution, this is, different resolution over different axis. The effect of anisotropic resolution on the reconstruction leads



(a) (b)

Figure 9.3: Generation of the reference set: (a) Marching cubes, where staircase artifacts are visible and (b) Marching cubes with smoothing and decimation post-processing (staircase artifacts removed).

Table 9.1: Number of samples collected for EAM in the literature.

Reference	Number of samples
Sra *et al.* [12]	126 ± 13
Sy *et al.* [5]	90 ± 10
Smeets *et al.* [4]	110 ± 60
Porras *et al.* [13]	380 ± 219

to staircase artifacts (Fig. 9.3.a). In order to palliate this effect and to reduce the complexity of the mesh, surface smoothing and decimation techniques have been applied on the reference set $\mathbf{R}$. The result of this processing is shown in Fig. 9.3.b.

9.2.2 Sampling

During EAM, electrophysiologists acquire a variable number of samples in a process that can last a few hours. Though this number is usually between 100 and 200 samples, this can go as high as 600 as shown in Table 9.1. The number of samples and its distribution over the atrium chamber depends on the electrophysiologists criteria and it has an impact on the length of the procedure and the accuracy of the reconstructions.

In order to evaluate the impact of the sampling process in the atrium reconstruction, first, all the atria from the segmented set $\mathbf{A}$ are converted to oriented clouds of points $\mathbf{C} = \{C_1, C_2, ...C_{57}\}$ using gradient operators. Each of these clouds of points are then split into two separate clouds of points (upper and lower part of

the atrium):

$$\begin{aligned}\mathbf{U} &= \{\mathcal{U}_1, \mathcal{U}_2, ..., \mathcal{U}_{57}\} \\ \mathbf{L} &= \{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_{57}\}\end{aligned} \quad \Bigg| \quad \mathbf{C} = \mathbf{U} \uplus \mathbf{L} \tag{9.1}$$

with $\mathbf{U} \uplus \mathbf{L} = \{\mathcal{U}_1 \cup \mathcal{L}_1, U_2 \cup \mathcal{L}_2, ..., \mathcal{U}_{57} \cup \mathcal{L}_{57}\}$, where $\mathcal{U}_i \cup \mathcal{L}_i$ represents the union of the cloud of points $\mathcal{U}_i$ (upper) containing most geometrically complex part of the atrium, this is, the PVs; and $\mathcal{L}_i$ (lower) containing the remaining part of the atrium. The separation has been performed manually, with the aim of maximizing the volume contained in each $\mathcal{L}_i$, but making sure that all PVs are contained in $\mathcal{U}_i$. With this separation, different sampling densities can be given to different parts of the atrium, as it naturally happens during real EAM.

Once the atrium separation has been performed, a sampling space $\mathbf{S}$ representing all sampling conditions is generated as the Cartesian product ($\times$):

$$\mathbf{S} = \mathbf{P} \times \mathbf{D} = \{S_0, ..., S_{30}\} \tag{9.2}$$

with $\mathbf{P} = \{100, 200, ..., 600\}$ the set of number of samples approximating the space of samples in Table 9.1, and $\mathbf{D} = \{0.5, 0.6, ..., 0.9\}$ the distribution set in which $u \in \mathbf{D}$ is the proportion of samples taken from the upper part of the atrium ($\mathcal{U}_i$) and $l = 1 - u$ represents the proportion of samples taken from the lower part of the atrium ($\mathcal{L}_i$). This sampling space therefore, considers both, variability of samples and variability of distribution of samples.

For each atrium i in the data-set, a set of clouds of points $\mathbf{C}^i$ is generated by sampling according to the sampling conditions in $\mathbf{S}$:

$$\mathbf{C}^i = \{C^i_{S_0}, C^i_{S_1}, ..., C^i_{S_{30}}\} \tag{9.3}$$

9.2.3 Poisson Surface Reconstruction

Fig. 9.4 shows the general idea behind PSR. To obtain the surface, PSR uses the oriented cloud of points $C^i_{S_j}$ to reconstruct an indicator function $\mathcal{X}_M : \mathbb{R}^3 \to \mathbb{R}$ of an atrium model M. This indicator function resembles the segmented image A_i, this is, a function in which the values inside the atrium are $a_i = 0$ and the values outside the atrium are $a_o = 1$. The oriented cloud of points $C^i_{S_j}$ can be thought of as a set of samples taken from the gradient of the indicator function $\nabla\mathcal{X}_M$. Then the problem can be stated as finding the scalar function $\mathcal{X}_M$ whose gradient best matches the cloud of points $C^i_{S_j}$:

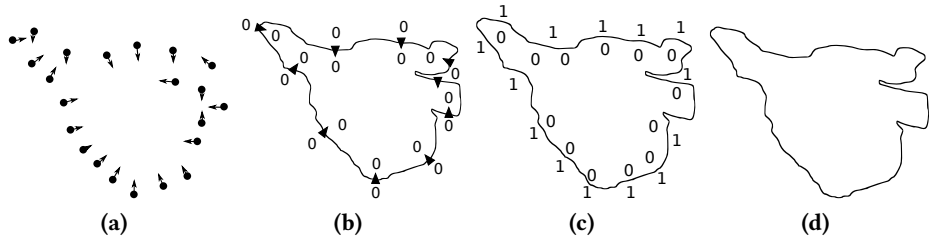


Figure 9.4: Overview of Poisson surface reconstruction in 2D contour of a left atrium: (a) oriented set of points $C^i_{S_j}$; (b) gradient of the indicator function $\nabla\mathcal{X}_M$; (c) indicator function $\mathcal{X}_M$; (d) Poisson surface reconstruction ∂P.

$$\tilde{\mathcal{X}}_M = \arg\min_{\mathcal{X}_M} \|\nabla\mathcal{X}_M - C^i_{S_j}\| \tag{9.4}$$

with $\|.\|$ the Euclidean norm.

The solution is represented through an adaptive and multi-resolution basis. More precisely, PSR constructs the minimal octree $\mathscr{O}$ with the property that every point sample falls into a leaf node at depth d. Intuitively, one can think of the depth as a parameter controlling the granularity of the mesh. Higher depth values thus lead to more complex models able to represent smaller features (and *vice versa*).

9.2.4 Error estimation

In this work, the error between two meshes is approximated as the *surface-to-surface* distance:

$$D_s(\mathcal{S}, \mathcal{S}') = \max_{p\in\mathcal{S}} D_p(p, \mathcal{S}') \tag{9.5}$$

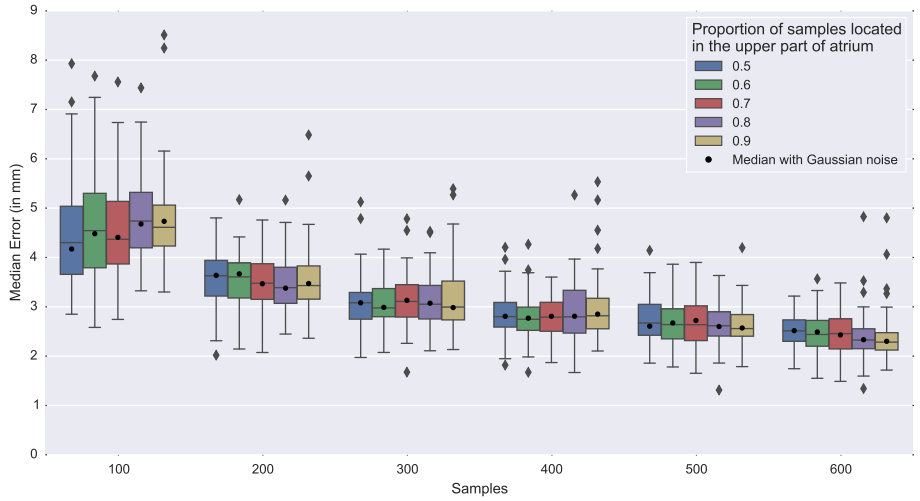
where p is a point contained in the surface described by $\mathcal{S}$. D_p is the *point-to-surface* distance defined as:

$$D_p(p, \mathcal{S}') = \min_{p'\in S'} \| p - p' \| \tag{9.6}$$

with $\| \cdot \|$ denoting the Euclidean norm. Since generally $D_s(\mathcal{S}, \mathcal{S}') \neq D_s(\mathcal{S}', \mathcal{S})$, we always calculate the distance from the reference mesh to the reconstructed meshes.

9.3 Results

The results presented in this section were obtained from the simulation process detailed in Section 9.2 to the *CARMA MRI Left Atria* data-set. The simulation sampling process, lead to the generation and reconstruction of 1710 atrium models.



Samples	100					200					300				
Upper	0.5	0.6	0.7	0.8	0.9	0.5	0.6	0.7	0.8	0.9	0.5	0.6	0.7	0.8	0.9
Median (no noise)	4.30	4.54	**4.37**	4.74	4.61	3.63	3.61	3.48	**3.39**	3.43	3.08	**2.98**	3.11	3.05	3.00
Median (noise)	**4.17**	4.48	4.40	4.67	4.73	3.63	3.66	**3.37**	3.47	3.08	2.98	3.13	3.07	**2.98**	
Samples	400					500					600				
Upper	0.5	0.6	0.7	0.8	0.9	0.5	0.6	0.7	0.8	0.9	0.5	0.6	0.7	0.8	0.9
Median (no noise)	2.80	**2.75**	2.79	2.80	2.82	2.67	2.64	2.64	2.62	**2.56**	2.51	2.44	2.46	2.33	**2.28**
Median (noise)	2.80	**2.76**	2.80	2.80	2.85	2.60	2.67	2.72	2.60	**2.57**	2.51	2.48	2.42	2.33	**2.30**

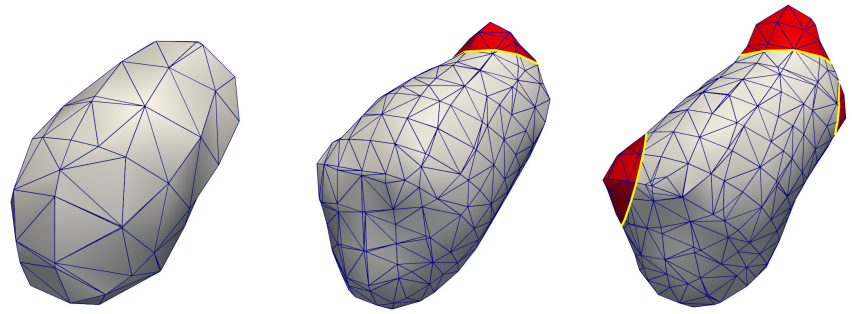
Minimum values per number of samples are highlighted in bold typeface.

Figure 9.5: Median reconstruction error using PSR ($d = 5$) under different sampling conditions (number of samples and distribution of samples over the atrium). The results in the table correspond to median values relative to the population of atria reconstructed.

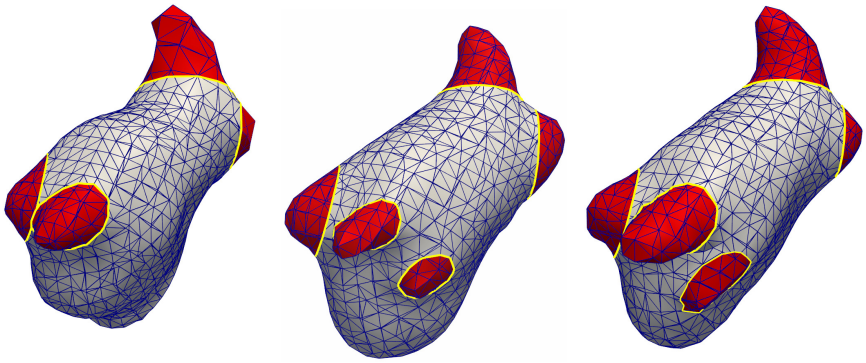
Additionally, the experiment was repeated for different PSR depth parameters ($d = \{4, 5, 6, 7, 8\}$) with the aim to find the best parameter in terms of accuracy. For $d > 4$ difference in results were negligible and therefore our results are based in PSR with $d = 5$. From the *point-to-surface* distance (Section 9.2.4), we derived the median error for every atrium reconstruction, which is employed to serve as a basis for our results (Figure 9.4).

In order to provide results for more realistic sampling conditions, we introduce Gaussian noise ($\mu = 0.76, \sigma = 0.67$) to every point in the cloud. This noise matches the characterization of the inaccuracies introduced by electromagnetic tracking in interventional radiology environments [14].

There is a clear trend of error reduction as the number of samples is increased. As shown in Figure 9.5, the mean error (considering different sampling proportions) ranges from: $4.52 \pm 0.22\ mm$ for 100 samples, $3.51 \pm 0.12\ mm$ for 200 samples, $3.04 \pm 0.06\ mm$ for 300 samples, $2.78 \pm 0.03\ mm$ for 400 samples, $2.61 \pm 0.05\ mm$



(a) $n = 100$ **(b)** $n = 200$ **(c)** $n = 300$



(d) $n = 400$ **(e)** $n = 500$ **(f)** $n = 600$

Figure 9.6: PSR ($d = 5, upper = 0.8$) of CARMA0046 atrium for different number of samples. Red areas indicate clear visibility of PVs and yellow contour indicate clear visibility of PVs ostia.

for 500 samples and $2.39 \pm 0.11\ mm$ for 600 samples. The proportions of upper atrium leading to the best reconstructions in terms of median error were: 0.7 for 100 samples, 0.8 for 200 samples, 0.6 for 300 samples, 0.6 for 400 samples, 0.9 for 500 samples and 0.9 for 600 samples. Including Gaussian noise does not change significantly the median errors (Figure 9.5).

The geometry of important features of the atrium, like the PVs ostia, where the ablation takes place, are not geometrically described for low number of samples. In this line, Figure 9.6 illustrates the reconstruction of CARMA0046 where the PVs are only visible for high number of samples (e.g. $n \geq 400$), while the PVs ostia can be visible for a relatively low number of samples (e.g. $n \geq 200$). For low number of samples (e.g. $n < 200$), PVs and PVs ostia are not visible.

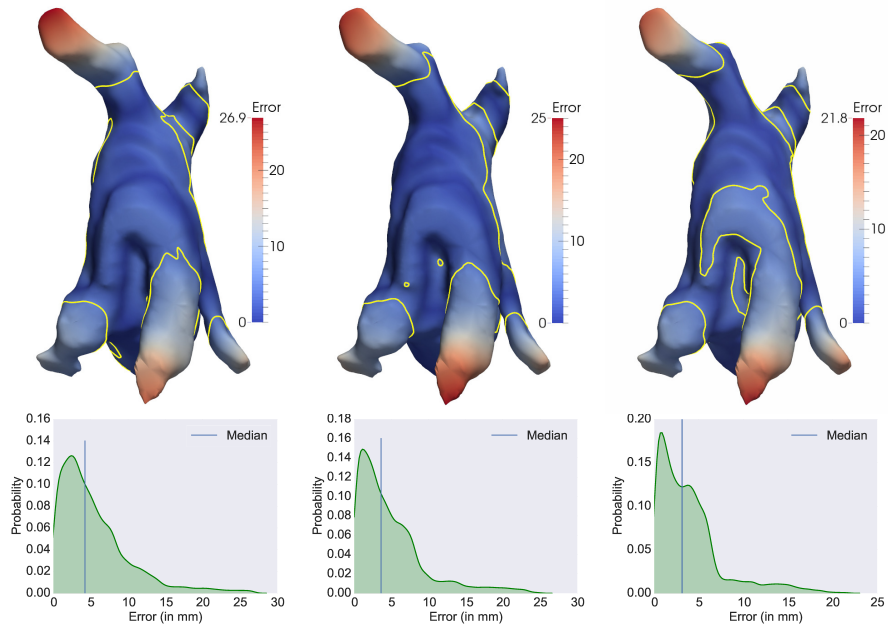
As shown in Figure 9.7, reconstruction errors over the atrium are not uniformly distributed. A low number of prominent errors are located deep in the PVs. Moderate errors were located in areas of relatively high curvature, like the PVs and small features of the atrium. As shown in the figure, these areas are close to the median error ($\tilde{e}$) contours. Finally, an elevated number of small errors were located in low-curvature areas of the atrium chamber.

Computing time for PSR, including input/output data transfers was within the range [0.31,2.5] seconds (median time $t_m = 0.60$ seconds).

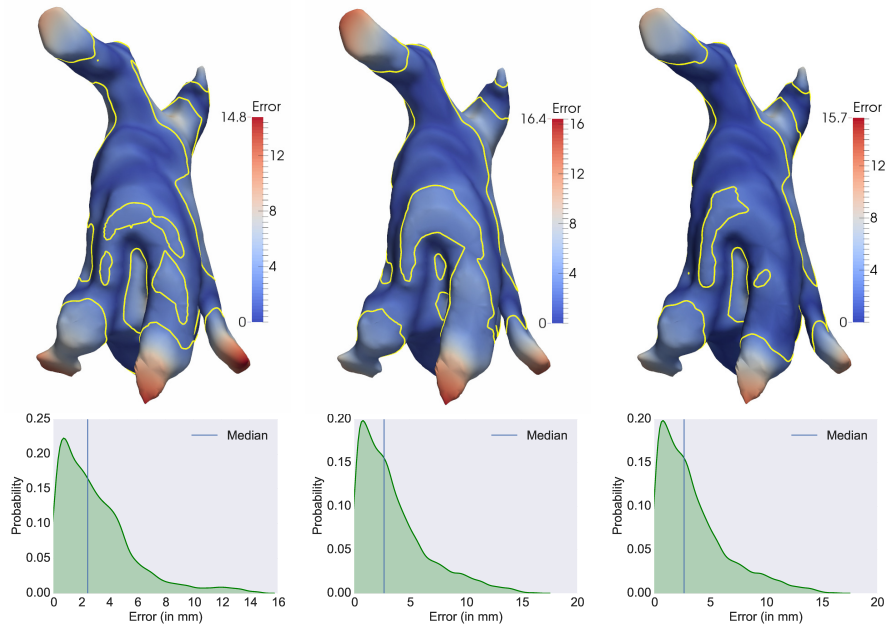
9.4 Discussion and Conclusion

In this work, the use of PSR for intra-operative modeling of left atria with application to RFCA is studied. The study is performed through a simulation approach, which is employed to isolate and analyze the errors introduced by solely the reconstruction method (PSR) and the sampling process. This, which is achieved by sampling static segmented MRI volumes (process that is not subject to motion and tracking errors), avoids errors contributed by other sources like breathing, cardiac motion or tracking technologies, thus leading to a better characterization of the error solely introduced by the method and the sampling process. This framework is applied for the evaluation of PSR from oriented clouds of points generated by sampling left atria under different sampling conditions. Clinically, as acquisition of points might not include normals, these can be estimated using techniques like in [15]. PSR has been previously used for the reconstruction of left atria under conditions of high density of samples [10, 11], which requires the use of pre-operative models or intra-operative imaging technologies. In our work, we analyze the PSR reconstruction under low-density sampling to obtain intra-operative models not requiring intra-operative imaging or pre-operative models. This approach, can be combined with different *state-of-the-art* techniques like image registration and image fusion, thus providing with more flexibility to reach a wider scope of clinical work-flows.

Our results show that, under low number of samples a median error in the range of $4.52 \pm 0.22\ mm$ is expected. This error can decrease down to $2.61 \pm 0.05\ mm$ provided that a high number of samples is acquired ($n = 600$). Though our results do not show any specific sampling proportion that should be employed in all cases, approaches where more samples are taken from the upper part of the atrium ($upper > 0.5$) generally present better behavior than just uniform sampling ($upper = 0.5$). This behavior is expected since the more complex geometry of the upper part of the atrium (local features and high curvature) would be better described by a higher number of sample points, in contrast to the relatively smooth and low-curvature shape of the lower part of the atrium. The



(a) $n = 100 \mid \tilde{e} = 4.23\ mm$ **(b)** $n = 200 \mid \tilde{e} = 3.60\ mm$ **(c)** $n = 300 \mid \tilde{e} = 3.11\ mm$



(d) $n = 400 \mid \tilde{e} = 2.44\ mm$ **(e)** $n = 500 \mid \tilde{e} = 2.67\ mm$ **(f)** $n = 600 \mid \tilde{e} = 2.38\ mm$

Figure 9.7: Error distribution PSR for CARMA0046 ($d = 5, upper = 0.8$), projected in the reference mesh. The contour (yellow) corresponds to the median error ($\tilde{e}$) without noise.

Table 9.2: Instances of errors (accuracy) reported in related works in the literature .

Reference	Errors reported	Type of error
[11]	$1.68\ mm$ RMS	Reconstruction from highly-dense cloud of points
[10]	$0.88 \pm 0.03\ mm$ RMS	Reconstruction from highly-dense clouds of points
[5]	$[2 - 6\ mm]$	Registration error
[6]*	$5.4 \pm 2.5\ mm$	Reconstruction in PVs (long axis) with motion compensation
	$3.3 \pm 2.7\ mm$	Reconstruction in PVs (short axis) with no motion compensation

*Authors made a much richer report of error values in this work. Here, for simplicity, only a sample of these was included.

number of samples considered in this study ($n = \{100, 200, ..., 600\}$) matches the range of those reported in the literature [4, 5, 12, 13]. When using PSR, our results shows the convenience of using a *moderate-to-high* number of samples to capture the geometry of the PVs ostia, which are the most important anatomic structures for EAM. Including Gaussian noise ($\mu = 0.76, \sigma = 0.67$) as expected in real acquisitions using electromagnetic tracking technologies does not increase the median errors significantly. This supports the adequacy of PSR for EAM.

The median accuracy reported in this work, is similar to the accuracy levels of other methods reported in the literature (Table 9.2). While this comparison can be used as indicator of adequacy of PSR for EAM, a careful comparison where all methods are evaluated in the same conditions would be needed to perform a more accurate assessment of the method. Furthermore, for anatomies where median errors are high (e.g. outliers in Figure 9.5) elevating the number of samples is indicated. Detection of these complex anatomies, as well as the separation in upper/lower atrium, could potentially be detected automatically by prior shape analysis (e.g. curvature and local features). Such analysis could also help performing non-uniform sampling based on curvature/complexity criteria.

New technologies like catheters with force-sensing capabilities [16], together with motion compensation techniques have the potential to enable the acquisition of more sample points and therefore, the possibility of reducing the reconstruction error beyond the results presented in this work. The framework presented in this work can be employed to characterize these reconstructions which can fill the gap between our approach and that of reconstructions using high-density clouds of points.

Acknowledgments. The authors would like to thank the Comprehensive Arrhythmia Research and Management (CARMA) Center, the Center for Integrative Biomedical Computing (CIBC) and the National Institute of Health (NIH), for providing the *CARMA Left Atria MRI* data-set employed in this research.

Bibliography

[1] Yoko Miyasaka, Marion E Barnes, Bernard J Gersh, Stephen S Cha, Kent R Bailey, Walter P Abhayaratna, James B Seward, and Teresa S M Tsang. Secular trends in incidence of atrial fibrillation in Olmsted County, Minnesota, 1980 to 2000, and implications on the projections for future prevalence. *Circulation*, 114(2):119–125, 2006. ISSN 1524-4539.

[2] Deepak Bhakta and John M Miller. Principles of Electroanatomic Mapping. *Indian Pacing And Electrophysiology Journal*, 8(1):32–50, 2008.

[3] Gregory F Michaud and Roy John. Percutaneous pulmonary vein isolation for atrial fibrillation ablation. *Circulation*, 123(20):e596—-601, 2011. ISSN 1524-4539.

[4] J L R M Smeets, S a. Ben-Haim, L.-M. Rodriguez, C Timmermans, and H J J Wellens. New Method for Nonfluoroscopic Endocardial Mapping in Humans : Accuracy Assessment and First Clinical Results. *Circulation*, 97(24):2426–2432, 1998. ISSN 0009-7322.

[5] Raymond W Sy, Aravinda Thiagalingam, and Martin K Stiles. Modern electrophysiology mapping techniques. *Heart, lung & circulation*, 21(6-7): 364–375, 2012. ISSN 1444-2892.

[6] Roy Beinart, Rajesh Kabra, Kevin E Heist, Dan Blendea, Conor D Barrett, Stephan B Danik, Ryan Collins, Jeremy N Ruskin, and Moussa Mansour. Respiratory compensation improves the accuracy of electroanatomic mapping of the left atrium and pulmonary veins during atrial fibrillation ablation. *Journal of interventional cardiac electrophysiology : an international journal of arrhythmias and pacing*, may 2011. ISSN 1572-8595. doi: 10.1007/s10840-011-9583-z.

[7] Daniel Reisfeld. Three-Dimensional Reconstruction of Intra-Body Organs., 2001.

[8] Patricia Chiang, Jianmin Zheng, Koon Hou Mak, Nadia Magnenat Thalmann, and Yiyu Cai. Progressive surface reconstruction for heart mapping procedure. *Computer-Aided Design*, 44(4):289–299, apr 2012. ISSN 00104485.

[9] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. *Eurographics Symposium on Geometry Processing 2006*, pages 61–70, 2006.

[10] Deyu Sun, Maryam E Rettmann, David R Holmes III, Cristian Linte, Bruce Cameron, Jiquan Liu, Douglas Packer, and Richard A Robb. Anatomic surface

reconstruction from sampled point cloud data and prior models. *Studies in health technology and informatics*, 196:387, 2014.

[11] Deyu Sun, Maryam E Rettmann, Douglas Packer, Richard A Robb, and David R Holmes. Simulated evaluation of an intraoperative surface modeling method for catheter ablation by a real phantom simulation experiment. In *SPIE Medical Imaging*, pages 94152N—-94152N. International Society for Optics and Photonics, 2015.

[12] Jasbir Sra and Masood Akhtar. Mapping techniques for atrial fibrillation ablation. *Current problems in cardiology*, 32(12):669–767, 2007. ISSN 0146-2806.

[13] Antonio R Porras, Gemma Piella, Oscar Cámara, Etelvino Silva, David Andreu, Antonio Berruezo, and Alejandro F Frangi. Cardiac Deformation from Electro-Anatomical Mapping Data: Application to Scar Characterization. In *FIMH*, pages 47–54. Springer, 2011.

[14] Ziv Yaniv, Emmanuel Wilson, David Lindisch, and Kevin Cleary. Electromagnetic tracking in the clinical environment. *Medical Physics*, 36(3):876, 2009. ISSN 00942405. doi: 10.1118/1.3075829.

[15] Niloy J. Mitra, An Nguyen, and Leonidas Guibas. Estimating Surface Normals in Noisy Point Cloud Data. *International Journal of Computational Geometry and Applications*, 14(4-5):261–276, 2004. ISSN 0218-1959. doi: 10.1145/777792.777840.

[16] Kurt S. Hoffmayer and Edward P. Gerstenfeld. Contact force-sensing catheters. *Current Opinion in Cardiology*, 30(1):74–80, 2015. ISSN 0268-4705. doi: 10.1097/HCO.0000000000000131.

Part III

Appendices

APPENDIX A

ERRATA AND CORRIGENDA

This appendix is a record of errata and corrigenda of the original publications as presented in Part II of the thesis document. None of the errors reported imply either variations on the methodology or significant deviation of results and conclusions presented in the original publications.

- Eq. (7.4) in page (92) should be: $\mathbf{T}(\mathbf{S}) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{T}(\mathbf{C}_{i,j}) B_{i,m}(u) B_{j,n}(v)$.
- Eq. (8.5) in page (122) (also used in Fig. 8.3) should be:

$$\mathbf{S}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{P}_{i,j} \overbrace{\underbrace{\binom{m}{i}}_{\mathbf{C^u}} (1-u)^{(m-i)} u^i}^{\mathbf{B^u}} \overbrace{\underbrace{\binom{n}{j}}_{\mathbf{C^v}} (1-v)^{(n-j)} v^j}^{\mathbf{B^v}}.$$

- The statistical analysis in Section 6.4.4 is based on the use of the *two-sample t-test*. However, the underlying data this operates on contains data-dependences which can increase the error of tests. A more appropriate test is the Wilcoxon signed rank test, which application yields:

$$\partial C_{best} / \partial C_{sim} \rightarrow p = 0.001027$$
$$\partial C_{best} / \partial P_{best} \rightarrow p = 0.000893$$
$$\partial P_{best} / \partial C_{sim} \rightarrow p = 0.8287$$