University of California

Los Angeles

Patient-Specific Interactive Ultrasound Image Simulation Based on the Deformation of Soft Tissue

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Computer Science

by

Kresimir Petrinec

2013

Abstract of the Dissertation

Patient-Specific Interactive Ultrasound Image Simulation Based on the Deformation of Soft Tissue

by

Kresimir Petrinec

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2013

Professor Demetri Terzopoulos, Chair

Ultrasound imaging systems provide a low-cost, real-time, noninvasive and safe way to examine soft tissues inside the human body. Yet, the ability of medical practitioners to understand and mentally register two-dimensional (2D) ultrasound image slices within the three-dimensional (3D) anatomy is a difficult task, so training is needed. Current ultrasound training methods are expensive, inefficient, and pose a major obstacle to the wide deployment of ultrasound imaging systems in routine clinical practice. In this thesis, we present a new approach to ultrasound training, where complex and expensive phantoms are replaced by a 3D virtual patient model, which represents the anatomy of any desired body part or organ, and is simulated on a standard laptop computer. The advantage of our system is not only its cost effectiveness, but also its ability to emulate different disease states or conditions in different virtual patients and to visualize the underlying body structures of interest through different examination procedures with a virtual ultrasound probe.

Conventional 3D models of the human body are purely geometric and do not model soft-tissue mechanics and deformation, which is an important factor in the practice of clinical ultrasound imaging. To address this limitation, we introduce

real-time interactive soft tissue simulation in our 3D patient model. For this purpose, we adapt and evaluate two well-known deformable model simulation methods: mass-spring-damper systems (MSDS), and the finite element method (FEM) with a quasistatic solution of isotropic linear elastic materials with Cauchy strain. We apply these methods to the simulation of ultrasound in soft tissues. The soft tissue model in its undeformed state is determined by static real-patient data captured by applying a linear ultrasound probe on the neck and on the left upper arm. A visual tracking system is used to control the virtual probe. We achieve real-time interactive simulation rates by carefully adapting the code to run efficiently on multicore personal computers.

Our real-time, interactive simulation of a 3D virtual patient with deformable soft tissues enables cheaper, more efficient, and more effective ultrasound training. Our ultrasound training system promises to facilitate the broader use of ultrasound in healthcare and reduce the number of medical procedure complications.

The dissertation of Kresimir Petrinec is approved.

Douglas S. Parker

Song-Chun Zhu

Eric Savitsky

Demetri Terzopoulos, Committee Chair

University of California, Los Angeles

2013

To my mother and father, and to my wife.

For their support, encouragement, and constant love.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to express the deepest appreciation to my adviser, Professor Demetri Terzopoulos, for his valuable and constructive suggestions, vast knowledge, encouragement for setting the highest standards, generous ideas, support, and understanding. Without his patient guidance and his help, this dissertation would not have been possible.

I am very grateful to my committee thesis member, Professor Eric Savitsky, MD, for giving me the opportunity to work on the ultrasound training system, for introducing me to the importance of ultrasound in emergency medicine and to the need for more cost-efficient ultrasound training, as well as for constant belief in the success of the project. Thanks also to Professors Douglas S. Parker and Song-Chun Zhu for agreeing to serve on my thesis committee; without them my defence might never have taken place.

Many thanks to Cheryl Hein, managing director of the Center for Advanced Surgical and Interventional Technology (CASIT), for thoughtful comments and guidance through the initial phase of the project. I am also grateful to Jyotsna Vitale, director of ultrasound operations for the UCLA Department of Radiology, who gave me access to ultrasound equipment and assisted me in my initial ultrasound data acquisitions.

In the early stages of this project, I was fortunate to receive a Rothman Family Seed Funding donation through the CASIT. The funding allowed me to begin work on the Ultrasound-Guided Procedural Training project, designed to revolutionize ultrasound education and training. It also gave me access to CASIT's valuable technical support on this project.

I am grateful to my friends and colleagues at the UCLA Computer Graphics and Vision Laboratory for sharing their knowledge and ideas with me (in no particular order): Gabriele Nataneli, Sung-Hee Lee, Shawn Singh, Billy Hewlett,

Mubbasir Kapadia, Brian Allen, Eftychios Sifakis, Weiguang Si (Justin), Wenjia Huang, Masaki Nakada, Gergely Klar, Eduardo Ribeiro Poyart, Matthew Wang, Konstantinos Sideras, and Garett Ridge.

Finally, no such effort can be attempted successfully without family support. I would like to thank my father, Professor Zdravko Petrinec, for being my role-model and encouraging me to pursue graduate studies; my mother, Marija Petrinec, for teaching me science since an early age; and my sister, Irena Petrinec, who inspired me with her perseverance and hard work. My parents-in-law, Sanja and Borislav Starcevic, helped to make Los Angeles a true home. Last, but not least, I am most grateful to my wife Maja Starcevic, for her love, compassion, patience, and optimism throughout my study. Our son Charlie has enriched my life in ways that cannot be described.

Vita

1999	B.Sc. (Electrical Engineering), Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia.
2004	M.Sc. (Electrical Engineering), Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia.
1999–2005	Teaching and Research Assistant, Department of Control and Computer Engineering, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. Member of the Laboratory for Robotics and Intelligent Control Systems (LARICS) group.
2005–2006	Research on renewable energy power systems and FPGA programming using Handel-C at Anglia Ruskin University, Cambridge, United Kingdom
2006–2007	Researcher and software developer (Systems Engineer), Software Imaging, Oxford, United Kingdom.
2007–2013	Graduate student researcher, Computer Science Department, UCLA, Los Angeles, California.
2007–2010	Researcher, Center for Advanced Surgical and Interventional Technology (CASIT), UCLA, Los Angeles, California.
2010	M.Sc. (Computer Science), Computer Science Department, UCLA, Los Angeles, California.
2010–present	Senior Software Engineer, Research and Development, SonoSim Inc., Santa Monica, California.

Publications

K. Petrinec, E. Savitsky, C. Hein, "Patient-specific cases for an ultrasound training simulator", *Studies in Health Technology and Informatics*, Vol. 163, pp. 447–53, 2011.

CHAPTER 1

Introduction

1.1 Motivation

Ultrasound imaging systems provide a low-cost, real-time, noninvasive and safe way to examine soft tissues inside the human body. The quality and applicability of ultrasound imaging has been increasing for the last decade so that, coupled with its non-invasive features, non-ionizing radiation, and declining price, it is the preferred imaging modality for a rapidly growing range of procedures, including diagnostic imaging and image-guided interventional procedures. However, the technician's and clinician's ability to understand and mentally register 2D image slices within the 3D anatomy is a difficult task and is a huge barrier to the widespread use of ultrasound imaging in daily clinical practice. Consequently, the demand for ultrasound training is expanding.

Current ultrasound training methods are expensive and inefficient. Training requires expert diagnostic medical sonographers to teach clinicians to use ultrasound machines that are shared with clinical practices and require patient models or physical tissue phantoms for practice. This conventional approach to training healthcare providers suffers from a number of limitations, including a) lack of exposure to an appropriate range of disease and injury cases; b) inability to relate ultrasound imagery directly to underlying patient anatomy; and c) inability to widely deploy training systems independent of access to ultrasound machines and patient or physical phantom models.

Typical computer-based training systems use expensive tissue phantoms equipped with complex systems for the tracking of ultrasound probe positions and orientations. They display ultrasound brightness scans (B-scans), which are either sampled from three-dimensional (3D) data, or synthesize an approximation of ultrasound data directly from computed tomography (CT) scans. The sampled B-scans suffer from a small range of motion due to the inability to capture large ultrasound volumetric data, and the synthesized B-scans are not able to truly replicate ultrasound, because ultrasound speckles, which are caused by the interference of the signal reflected by tissue micro-inhomogeneities, such as capillaries or blood cells, are below the resolution of the CT scan.

In practice, the deformation of soft tissues (e.g., tendons, ligaments, fascia, skin, fat, muscles, nerves, blood vessels, etc.) when the probe is pressed against the skin has a large diagnostic value. A good example is ultrasound-guided central venous catheter (CVC) placement (Fig. 1.1), where the difference between vein and artery can be easily determined by compressibility and shape. However, traditional blind techniques, which rely on anatomical landmarks to estimate the location of vessels, result in a high rate of complications. What makes the traditional landmark approach problematic is that many factors, such as intravenous drug use, cardiac arrest, or even body type (e.g., underweight or overweight), can alter the usual anatomic relationship, so physicians require multiple attempts to cannulate the vessel of interest. On the other hand, ultrasound allows for the real-time imaging of vessels during CVC placement making it safer, faster, and easier, and it is not a surprise that ultrasound guided CVC placement is becoming a standard. Another good example where ultrasound compression is useful is in the diagnosis of deep vein thrombosis (DVT). For example, when the probe is pressed against the skin of a normal patient, the veins collapse easily and the deformation is instantly visible in the ultrasound image. If the veins do not deform under pressure, which can also be clearly visible in the ultrasound image, than

Figure 1.1: Ultrasound guided central venous catheter (CVC) placement.

that may indicate a positive finding for venous occlusion (Crisp et al., 2010).

Real-time interactive ultrasound simulation with soft-tissue deformation poses a difficult problem. Most methods suitable for the simulation of soft tissue are computationally intensive and run slowly on large data sets. Our motivation is to bring the training to a higher level and develop an interactive ultrasound training system capable of simulating ultrasound B-scans of tissue under compression. The availability of such training would result in the wider use of ultrasound in healthcare and reduce the number of medical procedure complications.

1.2 Thesis Contributions

In this thesis, we present a new approach to ultrasound training. Our work draws on multiple disciplines and combines them in order to achieve real-time interactive ultrasound simulation with soft-tissue deformation. Knowledge of ultrasound physics is essential for ultrasound reconstruction; human anatomy and medicine for volume alignments and grasping major issues involved in ultrasound imaging of various organs and pathologies; structural engineering and mathematics for

soft-tissue simulation; software engineering for good design and architecture; and computer science to achieve robust, stable and efficient algorithms.

In particular, this thesis makes the following contributions:

- We develop a laptop-based ultrasound simulator with real-patient, case-specific data registered with a virtual patient model. This contribution has been published in (Petrinec et al., 2011). The simulator has become a commercial product of SonoSim, Inc.[1]

- We propose a model for the real-time interactive simulation of ultrasound in deformable tissue based on real-patient data. The focus is on the simulation of vessel compression in the body. In this model we embed 3D ultrasound data in a tetrahedral mesh, and simulate the deformation of the mesh using either the finite element method or mass-spring-damper systems. We also show how to efficiently sample slices from the deformed tissue in order to synthesize and display ultrasound B-scans.

- To handle large deformable tissue volumes, we introduce parallel deformation simulation on multiple sub-volumes and real-time volume stitching.

- For better visualization, we propose an interactive deformation of the skin of the virtual body model when in contact with the virtual ultrasound probe, and an interactive visualization of how the segmented surfaces (vessels) deform under compression applied to the tissue.

- We develop a pipeline for real-patient data acquisition and data processing, where we (i) acquire timestamped 2D ultrasound images, (ii) reconstruct 3D volumetric ultrasound images using a pixel-based method, and (iii) segment 3D data with vascular structures using a semi-automatic method. The

[1] See http://sonosim.com.

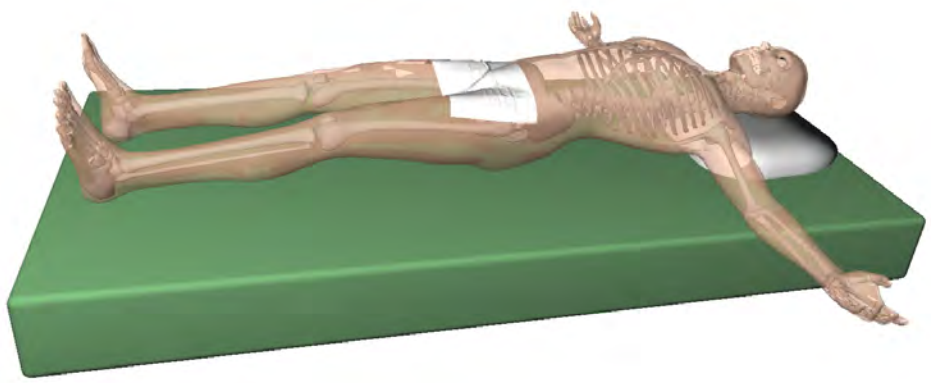


Figure 1.2: A virtual patient—the Ultimate Human Model, male cgCharacter.

method utilizes a shape-morphing algorithm, which allows for a quick segmentation from parallel B-scans.

- To track the orientation of the ultrasound probe, we propose a robust visual tracking system with ID-based fiducial markers.

The following sections overview the aforementationed contributions in greater detail.

Ultrasound Simulator with Real-Patient Data

We present a new approach to ultrasound training: a laptop-based ultrasound simulator with real-patient, case-specific data aligned with a virtual patient (Fig. 1.2). The simulator provides interactive simulation of ultrasound brightness scan images displayed on the screen simultaneously with the virtual patient and virtual probe. An external probe with a motion tracker embedded in the probe housing controls the orientation of the virtual probe. The user can select a number of real cases from a case list and experience the feel of a real ultrasound examination.

The simulator has been commercialized by SonoSim, Inc. It plays a major role in the SonoSim® Ultrasound Training Solution called "Hands-On Scanning", in

conjunction with didactic courses and knowledge assessment modules. SonoSim claims that the Solution improves ultrasound performance measures for basic scanning procedures and diagnostic window interpretation and that it is more effective than a live instructor in teaching users how to perform diagnostic ultrasound window interpretation.

Simulation of Ultrasound Compression

We develop a real-time interactive simulation of ultrasound compression. Ultrasound compression is useful in many procedures, such as ultrasound-guided central venous catheter (CVC) placement, or the diagnosis of deep vein thrombosis (DVT). To simulate ultrasound in soft tissues in our 3D patient model, we adapt and evaluate two well-known deformable model simulation methods—mass-spring-damper systems (MSDS) and the finite element method (FEM) with a quasistatic solution of isotropic linear elastic materials with Cauchy strain. The soft tissue model in its undeformed state is determined by static real-patient data.

For slice sampling from a deformable mesh, we present an efficient new method. Real-time interpolation of slices (or slice sampling) from the mesh is very challenging because there is no linear mapping between slice position/orientation and data embedded in a tetrahedral mesh. A naive approach samples the deformed state for every pixel, which is time inefficient.

Our virtual patient model, shown in Fig. 1.2, consists of the skin and bones. Muscles are excluded because they make it difficult to see veins, arteries, nerves and bones. Veins and nerves are patient-specific and they are dynamically created in the simulation. They deform as the user interacts with the system.

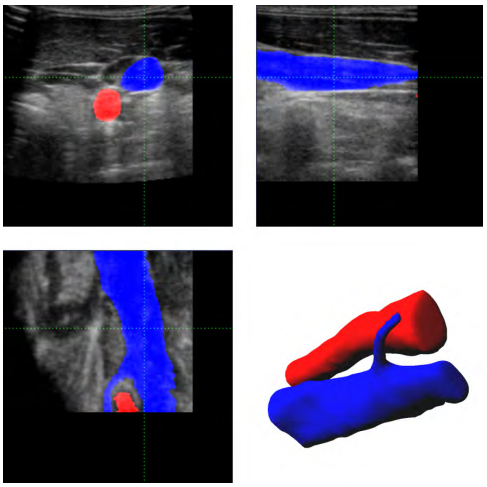


Figure 1.3: Vessels segmented using our semi-automatic segmentation method.

Ultrasound Data Acquisition and Processing Pipeline

We develop a data processing pipeline for our ultrasound simulator. First, we acquire real-patient data using a two-dimensional ultrasound probe with a three degree-of-freedom motion tracker attached to it. Second, we reconstruct volumetric data using pixel-based reconstruction method. Third, we segment the reconstructed data with a semi-automatic segmentation method. Finally, we manually align the processed data with our virtual human model.

Our tool for semi-automatic segmentation allows for a quick segmentation of 3D ultrasound volumes with vessels, nerves, and bones (Fig. 1.3).

Interactive Skin and Visualization of Deformable Isosurfaces

Conventional 3D models of the human body are purely geometric and do not model soft-tissue mechanics and deformation, which is an important factor in the practice of clinical ultrasound imaging. To address this limitation, we introduce real-time interactive deformation of the virtual body skin when in contact with the virtual ultrasound probe. The skin deformation is simulated using MSDS. In addition, we introduce real-time interactive deformation of segmented isosurfaces,

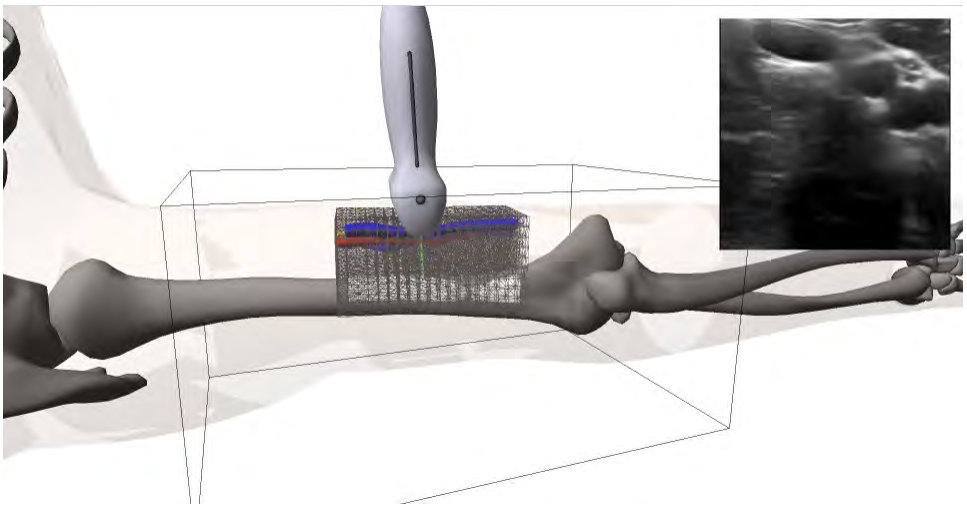


Figure 1.4: Vessels in the soft tissue mesh under the skin.

such as veins, arteries, and nerves. We achieve the deformation by embedding the isosurfaces into the soft tissue simulation mesh.

A snapshot of our simulator in Fig. 1.4 shows the skeleton of our virtul body, a virtual ultrasound probe, a mesh with the embedded isosurfaces of vessels and nerves, and a simulated ultrasound B-scan. The probe applies a pressure on the skin and soft-tissue, and the embedded vein is compressed. Simultaneously, the resulting simulated 2D ultrasound B-scan shows vein deformation.

Simulation of Large Volumes with Volume Stitching

We propose a real-time method suitable for simulating the deformation of large volumes of soft tissues that are partially anchored to a rigid structure such as bone. We divide the soft-tissue volume into smaller overlapping volumes, simulate the smaller volumes in parallel, and reconnect them.

Visual Motion Tracking

We present visual motion tracking with unique ID-based markers attached to the faces of a dodecahedron. Such an arrangement of makers allows for better visibility of multiple markers and allows for more robust marker tracking in single-camera visual motion tracking systems.

1.3 Thesis Outline

The remainder of the dissertation is organized as follows: Chapter 2 provides an introduction to ultrasound image formation, data acquisition and reconstruction, techniques for modeling soft tissue, and related work. Chapter 3 presents our ultrasound training simulator with static volumetric data. Chapter 4 presents our ultrasound training simulator with the simulation of soft tissue. Chapter 5 presents volume stitching. Finally, Chapter 6 concludes the dissertation, summarizing our work. Appendix A reviews the relevant mathematical basis of the theory of elasticity. Appendices B and C review the FEM and FVM, respectively. Appendix D describes different ways to track motion and introduces our optical tracking system.

CHAPTER 2

Background and Related Work

In this chapter, we will first cover the basics of ultrasound image formation and of real-patient ultrasound data acquisition and processing. We then review related work on ultrasound training and training systems. Finally, we will review deformable modeling techniques, which will be useful in the subsequent development of our ultrasound training system.

2.1 Ultrasound Image Formation

Ultrasound, ultrasonography, or ultrasound imaging is a technique which uses high frequency sound waves and their echos to determine the structure of an object or a body. High frequency sound waves are sent into a body and, as they travel through, some are reflected at the interfaces between tissues. The ultrasound image is formed using the intensity and depth of reflected waves, where the depth is computed from the return time of reflected waves.

Ultrasound imaging is performed with a hand-held ultrasound probe, which is also called transducer. The transducer sends a sequence of repetitive ultrasonic pulses into a body. When waves propagate, waves lose energy ("attenuation") to the medium of propagation and cause weak local heating. The absorption of energy depends mostly on the density of tissue (the higher the density, the more absorption), and on the frequency of the ultrasound beam (the higher the frequency, the more absorption). Echos (wave reflections) from different target

Table 2.1: The velocity of sound in tissue.

Material	Speed [m/s]
Air at STP	330
Blood	1570
Bone	4080
Fat	1450
Kidney	1560
Liver	1570
Water	1480

objects and boundaries are received and amplified.

The reflection depends on the difference in impedance of the two tissues. Basic imaging by ultrasound uses only the amplitude information in the reflected signal—the reflected signal is sampled continuously. As the velocity of sound in tissue is fairly constant (Table 2.1), the time between the emission of a pulse and the reception of a reflected signal is dependent on the distance; i.e., the depth of the reflecting structure. Different structures will reflect different amounts of emitted energy. The time before a new pulse is sent out is dependent of the maximum desired image depth.

The reflecting structures, also called scatterers, do not only reflect directly back to the transmitter, but scatter the ultrasound in other directions. It is important to realize that the actual amount of energy that is reflected back to the probe—i.e., the amplitude of the reflected signal—is not only dependent on the reflection coefficient, but also on the direction of the reflected signal. Thus, an irregular scatterer will reflect only a portion back to the probe, and a more regular scatterer will reflect more if the reflecting surfaces are perpendicular to the ultrasound beam. Thus, the apparent density of the tissue on the ultrasound

image is also dependent on the fiber direction.

An ultrasound brightness scan (B-scan) image is the result of a rather complicated set of physical phenomena, namely, the insonification and resulting absorption, reflection, and coherent scattering from a tissue medium of pulsed radio frequency ultrasonic pressure waves, and the electronic detection of the backscattered or echo pulses for display as an image. The resulting pictures have a granular structure variously described, as above, by the terms "texture" or "speckle".

Signal processing in an ultrasound scanner begins with the shaping and delaying of excitation pulses applied to each element of the array to generate a focused, steered and apodized pulsed wave that propagates into the tissue. Echoes resulting from the scattering of the sound by tissue structures are received by all elements within the transducer array. Processing of these echo signals routinely begins at the individual channel (element) level with the application of apodization (window) functions, and dynamic focusing or steering delays.

2.1.1 Ultrasound Image Synthesis

Ultrasound B-scans can be synthesized either by accurately replicating the propagation of ultrasonic waves through the tissue, or by interpolating B-scans from a volume formed from pre-acquired images. The former, also known as the generative approach, requires accurate models of tissue scatterers, the probe, and wave interaction. Jensen and Nikolov (2000) use linear acoustics and apply fully synthetic aperture imaging to an artificial kidney model and corresponding optical images. Dillenseger et al. (2009) synthesize abdominal ultrasound images from CT data. Reichl et al. (2009) synthesize ultrasound images from CT data on the Graphics Processing Unit (GPU), resulting in significantly decreased algorithm run time. The generative approach relies on very accurate models, which in practice are not possible to acquire or synthesize in real-time, so the synthetic images

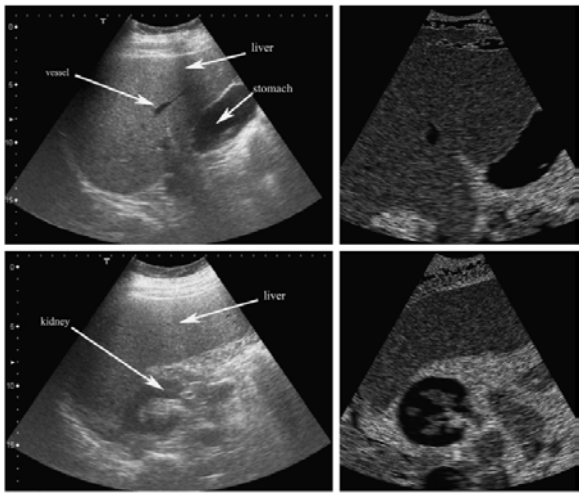


Figure 2.1: Comparison between two real 3.5 MHz ultrasound images (left) and the synthetic ones (right), from Dillenseger et al. (2009)

lack important fine details and have an artificial look, as shown in Fig. 2.1.

The later approach, also known as 3D ultrasound reconstruction, projects pre-acquired images into a regular 3D grid (volume). A thorough description and grouping of the various freehand 3D reconstruction algorithms can be found in (Solberg et al., 2007), where 3D volume reconstruction is classified into three categories based on implementation: Voxel-Based Methods (VBM), Pixel-Based Methods (PBM), and Function-Based Methods (FBM).

2.2 Ultrasound Data Acquisition and Processing

Data acquisition and processing in the context of three-dimensional (3D) ultrasound comprises the following steps: (i) determining the position and orientation of a sensor attached to the ultrasound probe, (ii) collecting ultrasound images, and (iii) timestamping all data collected for later post processing of each dataset, including the primary post processing action of volumetric reconstruction.

2.2.1 Acquisition

Methods for constructing ultrasound volumes from collected images, needed for ultrasound real-time simulation, can be classified in the following categories:

1. Constrained sweeping techniques
2. 3D probes
3. Sensorless techniques
4. Freehand techniques

Constrained sweeping techniques translate or rotate a 2D ultrasound probe over an area of interest by using an actuator (motor) (Shipley et al., 2005). A *3D probe* head contains either a 2D phased array of transducers, or mechanically or electronically steered 1D phased array of transducers. *Sensorless techniques* estimate the 3D position and orientation of a probe by analyzing speckle noise in the ultrasound images using decorrelation or linear regression (Prager et al., 2003; Rohling et al., 1998). *Freehand techniques* use tracking devices to track the ultrasound probe, permitting unconstrained movement (Barry et al., 1997; Sanches and Marques, 2002; Huang et al., 2005, 2009). This approach is more flexible than other methods, because data volume size is not limited by the mechanical design of the probe and the user has full control over the scanning direction (Gee et al., 2003).

2.2.2 Sensor Tracking

There are four common ultrasound sensor tracking technologies (Mercier et al., 2005):

1. Mechanical

2. Acoustical

3. Electromagnetic

4. Optical

Mechanical localizers, or articulated arms track angles of joints. The tips position and orientation are computed by solving a direct (forward) kinematic problem. *Acoustical position trackers* use detectors (microphones) to detect emitters (speakers) which emit ultrasound waves. Their position and orientation is computed by measuring the propagation time (time of flight) of sound waves or by measuring the phase difference to compute relative distance. *Electromagnetic systems* measure the electrical current that is induced when a sensor is moved within a magnetic field generated by either an alternating current (AC) or a direct current (DC) transmitter. *Optical trackers* detect markers placed on a rigid structure by using multiple cameras, where the structure geometry is known in advance.

2.2.3 Ultrasound Image Segmentation

Manual segmentation of two-dimensional (2D) ultrasound images is time consuming and labor intensive process with unrepeatable results; i.e., there is always a large variation between results, even among expert sonographers. On the other hand, fully automatic segmentation of ultrasound images is a very challenging task (if not impossible) not only because of low (degraded) image quality due to speckle noise, shadows and reflections, but also because of boundaries which may appear discontinuous due to the amplitude of echo, which depends on the orientation of the reflecting structure.

For that reason, most segmentation methods which are widely used for CT and MRI data fall apart on ultrasound data; for example, gray-level thresholding or region growing. Only in some cases when the target is known (e.g., the heart), the highly specialised segmentation algorithms can be applied and provide robust and

reliable results. Noble and Boukerroui (2006) provide a survey of ten influential papers in the ultrasound segmentation literature.

Semi-automatic methods provide the best of manual and automatic segmentation. Typically, the user guides automatic boundary detection to what appears to be the real boundary. The mostly used semi-automatic methods are active contours, and intelligent scissors algorithms.

Active contours, also known as "snakes", were introduced by Kass et al. (1988). Snakes are energy-minimizing parametric curves, whose energy depends on their shape and position within image. Snakes consist of abstract materials which make them resist stretching and bending. They are constrained to lie in the potential surface, which corresponds to image gradients, under the action of constant gravitational force. The snakes energy functional is the sum of internal energy due to stretching and bending, image potential energy, and external force energy. Depending on the energy, image forces can attract a snake to edges, lines, or terminations.

The intelligent scissors, also known as "live-wire", method was introduced by Mortensen and Barrett (1995, 1998). It defines a boundary via dynamic programming and formulates it as graph search for an optimal path. It allows the user to interactively select a start pixel (a seed point) and the most suitable boundary from a set of all optimal boundaries estimating from a seed point. On-the-fly training causes the boundary to adhere to the specific type of edge currently being followed.

Levienaise-Obadia and Gee (1999) present a semi-automated adaptive segmentation method which uses locally on-the-fly trained statistical models along the boundary to attract an active contour. They split the boundary in a number of spline segments and for each segment compute intensity gradients and first and second order grey level texture statistics along normal lines inside and outside the boundary. They train a classifier on one frame and apply it to the boundary in

other frames with small user interventions.

2.2.4 Ultrasound Volume Image Processing

Medical volumetric datasets, such as Computed Tomography (CT), Magnetic Resonance Imaging (MRI), or 3D/4D ultrasound, are typically stored in the Digital Imaging and Communications in Medicine (DICOM) standard format. Actual volumetric data inside a DICOM file are stored as arrays of samples, also called voxels, which lie on a regular spaced 3D grid. Voxels can be visualized directly by volume rendering, or by the extraction of isosurfaces. Volume rendering is a computationally intensive task where every voxel must be projected into screen plane. On the other hand, the isosurface is a surface which represents points of a constant value within a volume, so it can be rendered much faster than volume rendering due to small polygon count. For example, the isosurface may represent a bone in CT scan.

In this thesis we visualize segmented regions of vessels, nerves and bones with isosurfaces. We use isosurface rendering instead of volume rendering for two reasons—rendering isosurfaces is faster, and we need surfaces for the simulation of deformation.

A common technique for isosurface extraction is marching cubes (Lorensen and Cline, 1987). The algorithm 'marches' from cube to cube inside a volume, where each cube is formed from eight neighbor voxels, and creates tringles where the surface corresponding to a user-defined value intersects the cube. There are 256 ways a surface can intersect a cube. The algorithm enumerates those cases and stores them in a lookup table. By exploiting rotational and reflective symmetry, the table is reduced to 15 unique cases.

Later, it was discovered that the algorithm produces holes. Chernyaev (1995) showed that there is actually 33 topologically different configurations, and pre-

sented an algorithm which does not create holes. Schlei (2012) presents the volume-enclosing surface extraction (VESTA) technique which automatically detects and resolves potential topological ambiguities, and compares it to the marching cubes algorithm.

Isosurfaces produced by the marching cubes algorithm appear faceted. Gaussian filtering is often used to reduce the faceting. The new position of each vertex is computed as weighted average of the current vertex position and its first order neighbors (vertices which share an edge with the current vertex). However, Gaussian smoothing produces shrinkage because the convolution with a Gaussian kernel attenuates all frequencies (except the zero frequency). Taubin (1995) propose a solution to this problem by alternating Gaussian steps with positive and negative scale factors computed in a such a way that they produce a low-pass filter effect as a function of the natural frequencies of the shape.

2.3 Ultrasound Training Methods and Systems

The majority of current ultrasound training methods use real ultrasound machines on either phantoms or patients (other students, patient volunteers, or hired patient models). On the other hand, computer-based training systems are used only for the simulation and visualization of ultrasound images that correspond to probe position and orientation, and of phantoms equipped with complex systems for the tracking of probe positions and orientations (Terkamp et al., 2003; Maul et al., 2004; Ehricke, 1998; Heer et al., 2004; Varandas et al., 2004; Jensen, 1996).

Some previous ultrasound simulators synthesize an approximation of ultrasound data directly from computed tomography (CT) scans instead of from real-patient ultrasound data (Reichl et al., 2009; Hostettler et al., 2005; Dillenseger et al., 2009). However, those systems are not able to truly replicate ultrasound images, because ultrasound speckles, which come from interference of the signal

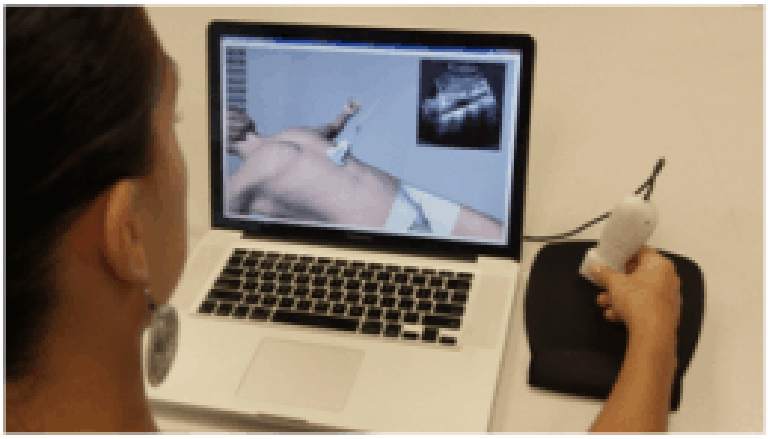


Figure 2.2: Ultrasound trainer.

reflected by tissue micro-inhomogeneities, such as capillaries or blood cells, are below the resolution of the CT scan (Dillenseger et al., 2009).

In (Petrinec et al., 2011), we present a new approach to ultrasound training. Complex and expensive phantoms are replaced with a virtual three-dimensional (3D) model, simulated on a standard laptop computer and representing the anatomy of any desired body part or organ, as shown in Fig. 2.2. The advantage of such a system is not only its lower price, but also its ability to simulate different procedures and disease states or conditions on different virtual patients and to visualize underlying body structures. Thus, the system enables quicker learning and helps improve hand-eye coordination required in ultrasound procedures. In addition to the virtual model, the system displays simulated ultrasound as well as CT or magnetic resonance imaging (MRI) B-scans, all based on real-patient data. The image on the trainer corresponds to a B-scan of the virtual probe placed on the virtual model. The image also corresponds to the peripheral probe, with an embedded orientation sensor, which is the primary user input device to the simulator. As the user moves the peripheral probe, the virtual probe moves accordingly and the image on the laptop (trainer) shows the image that would have been acquired in a real scan.

2.4 Deformable Models

Deformable models and related simulation techniques, which we review in this section, will provide the basis for simulating soft-tissue deformation in our augmented ultrasound training system.

2.4.1 The Theory of Elasticity

The theory of elasticity is a branch of continuum mechanics that studies the physics of continuous materials. More specifically, elasticity is a part of solid mechanics, which studies the physics of continuous materials with a defined rest shape, and which recover their rest shape after an applied stress is removed. Solid materials that permanently deform after a sufficient applied stress (greater than a yield value) are studied under plasticity. Another branch of continuum mechanics is fluid mechanics, which studies the physics of continuous materials that take the shape of their container.

The theory of elasticity is based on Hooke's law, which states that the deformation x of an object is proportional to the deforming force F. Simple linear springs obey Hooke's law: $F = kx$, where k is the spring constant. In continuum mechanics, multidimensional deformation (density) is characterized by strain ε, force (density) is characterized by stress σ, and material properties are characterized by moduli of elasticity, such as Young's modulus E. Appendix A reviews the mathematical formulation of these concepts.

The most commonly used methods for simulating deformable solids are the finite difference method (FDM), the finite element method (FEM), the boundary element method (BEM), and the finite volume method (FVM). When speed is more crucial than accuracy, a useful alternative is mass-spring-damper systems (MSDS).

2.4.2 Mass-Spring-Damper Systems

Mass-spring-damper systems (MSDS) are the most intuitive and simplest to implement deformable models. They comprise point masses connected together with massless springs and dampers—viscoelastic elements, also called Voigt elements (Fig. 2.3). The force acting on each mass is computed due to its spring connections with its neighbors, along with external forces. The motion (acceleration) of each particle is governed by Newton's second law. The force exerted by the Voigt element is

$$F = k\,(x_0 - x) - \gamma \frac{dx}{dt}, \tag{2.1}$$

where k is the spring constant, x_0 is the original (rest) length of the spring, x is the current length, and γ is damping constant. There are two categories of forces in MSDS—internal forces due to the tensions of springs and external forces due to gravity, collision, friction, etc. In equilibrium, the net force acting on every point mass is zero. The velocities and positions of the N point masses are computed by numerically solving a system of Lagrange equations of motion, the N second-order ordinary differential equations (Terzopoulos and Waters, 1990):

$$m_i \frac{d^2 x_i}{dt^2} + \gamma_i \frac{dx_i}{dt} + g_i = f_i; \quad i = 1, \ldots, N, \tag{2.2}$$

where m_i is the mass of i-th point mass, g_i is total spring force, and f_i is total external force on point mass i. These differential equations can be simulated by applying explicit, semi-implicit, or implicit techniques for numerical integration.

A simple technique for numerical integration, albeit limited in terms of stability, is the explicit Euler method:

$$a_i^t = \frac{1}{m_i}\left(f_i^t - \gamma_i v_i^t - g_i^t\right), \tag{2.3}$$

$$v_i^{t+\Delta t} = v_i^t + \Delta t a_i^t, \tag{2.4}$$

$$x_i^{t+\Delta t} = x_i^t + \Delta t v_i^{t+\Delta t}, \tag{2.5}$$

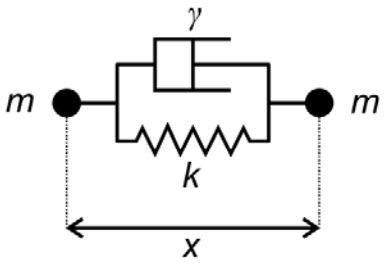


Figure 2.3: Voigt model.

where starting from given initial positions x_i^0 and velocities v_i^0 at time $t = 0$, accelerations a_i, velocities v_i, and positions x_i of point masses are computed at every time step $t = 1, \Delta t, 2\Delta t, \ldots$

The spring constants in MSDS are often chosen arbitrarily, and little can be said about the material being modeled. Although generalized springs may be employed to preserve areas and volumes, it is difficult to incorporate continuous material properties (Nealen et al., 2006). Even though they are not as accurate as other methods, MSDS are acceptable for many CGI applications in motion pictures and games. They have been successfully employed in various applications, such as cloth animation, facial animation, simulation of soft materials, and organic active bodies (Baraff and Witkin, 1998; Terzopoulos et al., 1991; Terzopoulos and Waters, 1990; Miller, 1988).

2.4.3 The Finite Element Method

The Voigt model can be regarded as a discrete (beam) "element" with which one can assemble deformable truss structures. In elastic continua, however, it does not suffice to apply the displacement method of analysis of beam and truss structures. Zienkiewicz and Taylor (2000) summarize the displacement formulation introduced by Clough (1960) which, approximates the continuum with "finite elements", such that the continuum is idealized as an assemblage of individual structural elements (Fig. 2.4); i.e., the structure is divided into small sections

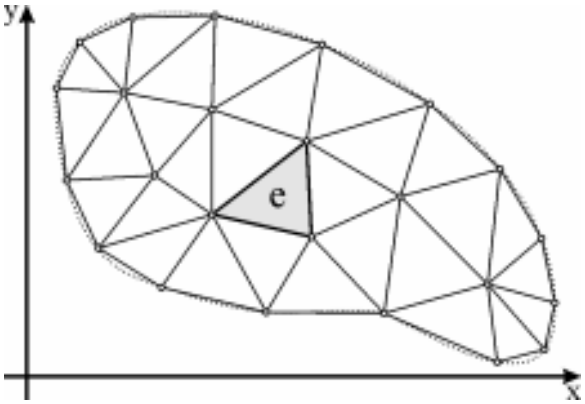


Figure 2.4: An example of a plane stress region divided into triangular-shaped finite elements.

represented by appropriate elements, such as beams, plates, shells, or solids. The elements are assumed to be interconnected at a discrete number of nodal points, as shown in the figure.

A set of (usually polynomial) functions is chosen to uniquely define the state of displacement within each element and on its boundaries in terms of its nodal displacements. On each element, the physical behavior is described with the element stiffness matrix K_e. The individual element stiffness matrices are assembled into a global stiffness matrix K. The displacement u of elements is characterized through the equilibrium equation, $Ku = f$, where f indicates nodal forces acting on the elements. After imposing boundary conditions, numerically solving the equilibrium equation for the system yields the element nodal displacements. The nodal displacements are then used in structural analysis to compute the stress within elements. For the relevant mathematical basis of the theory, see Appendix B.

Goksel and Salcudean (2009) used the FEM with linear-strain quasi-static elements to simulate the deformation of tissue mimicking a gelatin phantom with a soft cylindrical inclusion. For fixed nodes, they applied boundary conditions to the precomputed stiffness matrix by zeroing its corresponding rows and columns. They modeled probe interaction (indentation) as displacement constraints on the

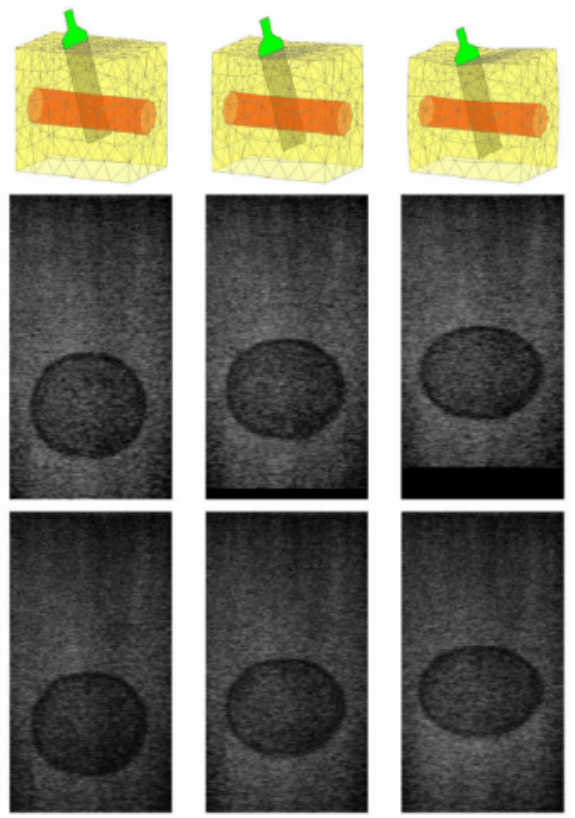


Figure 2.5: Simulated (upper) and acquired (lower) images with 0, 5, and 10 mm indentations for the probe tilted at 15 degrees; from (Goksel and Salcudean, 2009).

closest surface nodes. The mesh with 493 nodes and 1921 tetrahedra elements is generated by using off-the-shelf software. Real-time simulation of ultrasound images is achieved by mapping the 2D slice pixels back to the normal undeformed volume using their algorithm for fast image synthesis. An example of synthetic ultrasound images is shown in Fig. 2.5. In this example, in the FEM, the Young's moduli for the tissue and the cylinder were set to 15 KPa and 5 KPa, respectively, and a Poisson's ratio of 0.48 was used for all the elements.

2.4.4 Other Deformable Modeling Techniques

The FDM discretizes the problem's domain into a uniform grid of a finite number of points, and it approximates the derivative expressions in the differential

equations using finite differences (Strang, 1986). The solution (a pointwise approximation) is obtained using explicit, semi-implicit, or fully implicit schemes. Terzopoulos et al. (1987) in their seminal computer graphics work use the FDM to discretize continuous variational derivative of defined deformation energy functionals, and use semi-implicit integration to obtain the solution.

The BEM finds a solution to the system of governing equations which may be represented with boundary integral equations (BIEs). The boundary of the problem's domain is piecewise discretized into so-called boundary elements (El-Zafrany, 1993). The BEM is well-suited to the simulation of linear elastic isotropic and homogeneous materials (for which there exists a Green function) when the mesh topology remains fixed. Oftentimes in linear problems, the dimensionality of the problem is reduced by one, which results in data and CPU time reduction. In fact, the BEM has the important advantage over the FEM of not requiring the construction of a volumetric mesh, but is not well suited to simulating cutting.

The FVM was introduced to the computer graphics community by Teran et al. (2003). The method divides a continuum, in deformed configuration, into discrete regions around nodes. Nodal forces are computed from the surface integral, which reduces to the sum of simple products of element face area, stress tensor, and face normal, where the stress tensor is constant within an element when linear shape functions are used. The FVM relies on a geometrical framework, so it is intuitive and simple to understand. The relevant mathematical basis of the theory is reveiwed in Appendix C.

A comprehensive review of other deformable models, such as mesh-free and reduced deformation methods, can be found in Nealen et al. (2006).

2.4.5 Mesh Generation

Mesh generation represents the first step of any finite element method that engineers have to implement when the theoretical analysis of the problem is complete (Ciarlet and Lions, 1996). The mesh is responsible for the accuracy of the solution, so it is very important to capture the geometry of the domain and carefully approximate the boundaries of the domain. The definition of a mesh consists of connectivity between vertexes, its topology and of the coordinates of vertexes. A mesh can either be structured (also referred as a grid), or unstructured—its topology must be explicitly defined using a connectivity matrix.

Finite element solvers require a conforming mesh (without overlapping or intersecting elements and with some continuous properties at element interfaces). Ciarlet and Lions (1996) classified mesh generation methods in two main classes, and outlined the most popular methods accordingly. The first class corresponds to the algorithm complexity while the second corresponds to the field of applications in terms of the geometry to which the algorithms apply.

Automatic mesh generation (i.e., without user intervention) may be achieved with quadtree/octee type methods (Yerry and Shephard, 1985), advancing-front methods (Peraire et al., 1987) and Voronoi type methods (George et al., 1991). Molino et al., motivated by crystallography, use a body-centered cubic (BCC) mesh to generate a mesh whose connectivity is suitable for large deformations (Molino et al., 2003). The method is successfully applied in Teran et al. (2003).

In medical imaging, mesh generation typically requires image segmentation and labeling (marking voxels as inside or outside of some anatomical structure) (Archip et al., 2006). Goksel and Salcudean (2010) present variational modeling approach, which is more appropriate for soft tissue domains. They group voxels of similar intensities into elements while maintaining good element quality for the FEM. An example of optimized structured mesh overlaid on a synthetic phantom

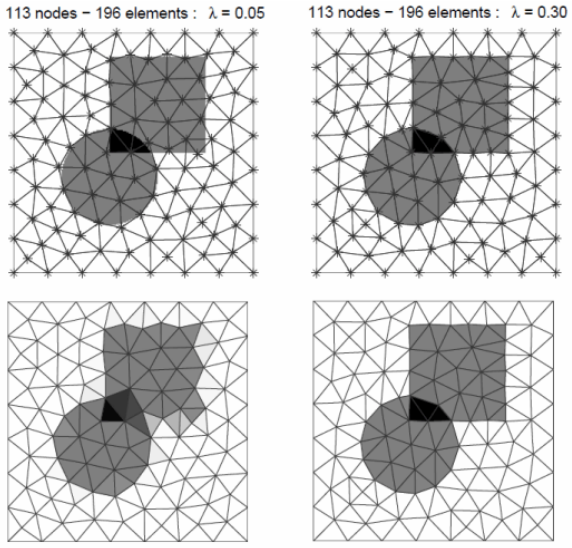


Figure 2.6: Synthetic phantom mesh optimized using $\lambda = 0.05$ (left column) and $\lambda = 0.3$ (right column), shown as meshes' overlaid in the image (top row) and the corresponding image approximations (bottom row); from (Goksel and Salcudean, 2010).

is shown in Fig. 2.6.

CHAPTER 3

Ultrasound Training Simulator

In this chapter, we will review the commercialized version of our simulator, and discuss how we acquire and process real-patient-based simulation data.

3.1 The Simulator

Our simulator, has been commercialized by SonoSim, inc., and is a part of the SonoSim® Ultrasound Training Solution product. The SonoSim® Personal Solution consists of hands-on scanning (the simulator), didactic courses, and knowledge assessment. Fig. 3.1 shows the SonoSim® laptop-based turnkey solutions that delivers an expansive array of courses, assessments, and hands-on training cases.

The hands-on scanning of the SonoSim® Ultrasound Solution provides the following basic feature set (Fig. 3.2):

1. Vitual human body

   A photorealistic three-dimensional human body model that accurately depicts anatomical structures.

2. Virtual ultrasound probe under interactive control and array beam

   This feature accurately depicts how the array beam traverses anatomical structures to create the ultrasound image in the display window.

3. Functional ultrasound unit display window

Figure 3.1: SonoSim® turnkey solution. Copyright SonoSim, Inc.

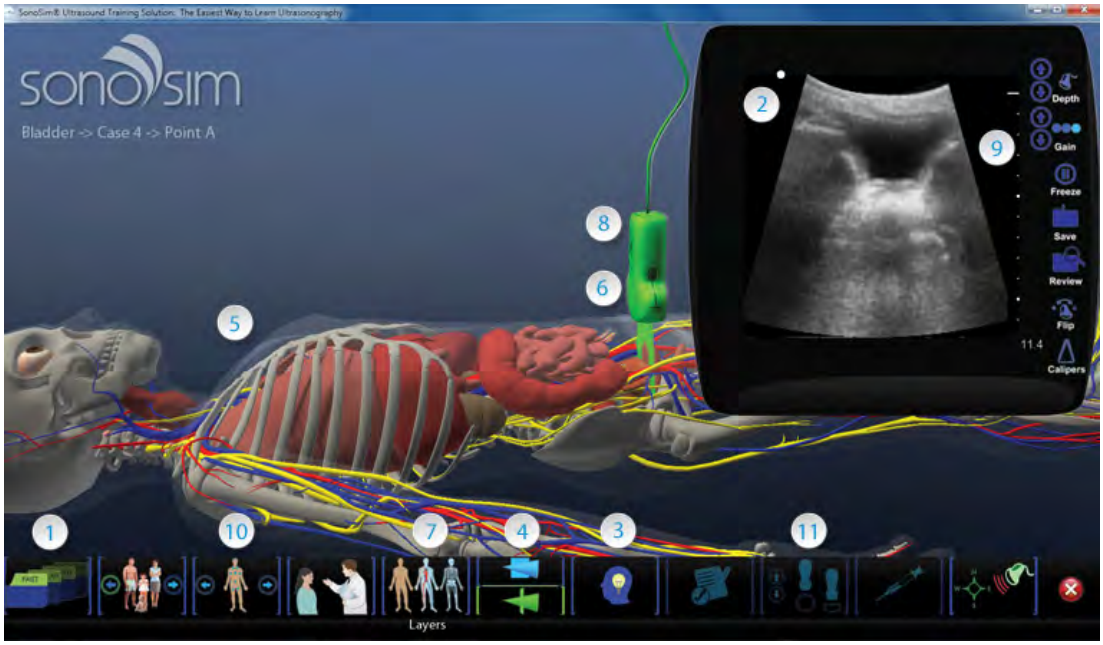


Figure 3.2: The SonoSimulator® features. Copyright SonoSim, Inc.

The ultrasound display window simulates a real ultrasound unit, including the ability to modify image gain and depth, save and review obtained images, and measure anatomic structures.

4. Ultrasound beam array penetration correleated with underlying layers of anatomy

   Anatomic Layers can be removed to reveal underlying anatomy, allowing the user to correlate the ultrasound image to relevant anatomic structures.

5. Appropriate ultrasound probe-type (i.e., physical footprint) and array beam

   This helps the user recognize which transducer to use when performing an actual scan.

6. Moving probe along different points of the body

   The navigation button allows a user to move along different regions or more discrete points along the body.

7. Teaches exact probe motion required for optimal ultrasound window acquisition

   The ultrasound display window depicts images that correspond to probe movements with extremely high fidelity and real-time performance.

8. Optimal ultrasound window acquisition guidance

   Probe-positioning assistance, for Longitudinal and Transverse window acquisition, allows for virtual hand-holding during image acquisition.

9. Apply compression over structures of interest

   The up and down compression buttons will allow you to compress or decompress the structure immediately beneath the transducer (e.g., compress a vein).

10. Vast library of real patient-based SonoSim® Cases

The Case List button provides immediate access to a wide-ranging and ever-growing SonoSim Case Library.

11. Individualized SonoSim® Case narrated tutorials

    The Findings button provides immediate, expert feedback. A narrated version of the original ultrasound clip describes what users should recognize while scanning a corresponding SonoSim® Case.

Recording patient-specific cases requires quick action. From the moment a patient arrives to a hospital for an exam there is very limited time when data acquisition can be performed. In addition, examination rooms are in most cases very small with no space for additional and cumbersome equipment such as conventional 3D ultrasound systems. Therefore, data acquisition must be performed using small and portable ultrasound systems. The image quality of portable systems nowadays is of comparable quality to standard ultrasound systems, but they lack the ability to capture 3D ultrasound. To overcome this problem, a very common approach is to track the position and orientation of the ultrasound probe using magnetic or optical motion trackers, and to reconstruct the volume offline.

Magnetic motion trackers utilize sensors placed on the body to measure the low-frequency magnetic field generated by a transmitter source. The sensors and source are cabled to an electronic control unit that correlates their reported locations within the field. They have limited working space and are very sensitive to magnetic fields. Optical trackers are sensitive to lightning conditions and require a clear line of sight between the camera and markers, which is hard to achieve in small exam rooms. Both magnetic and optical motion trackers require sophisticated calibration. The role of calibration is to find the mathematical transformation that converts the 2D coordinates of pixels in the ultrasound image into 3D coordinates in the frame of reference of a position sensor attached to the ultrasound probe (Mercier et al., 2005).

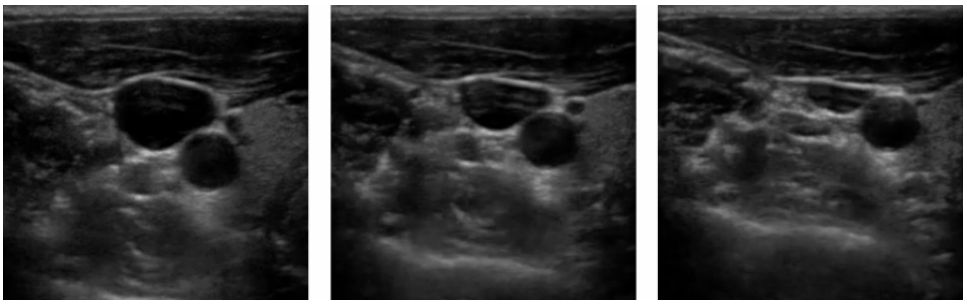


Figure 3.3: Compression of vessels (vein and artery) in the soft tissue.

On the other hand, 3-DOF motion trackers, such as the InterSense iCube3 motion tracker based on gyros and electromagnetic sensors, provide robust and precise orientation. It is small in size and easy to mount on any conventional ultrasound probe, and transmits data to PC via USB without any additional hardware box, all of which makes it suitable for data acquisition. The lack of information about the probe position can be compensated by introducing a scanning protocol where the sonographer is instructed to do a fan swipe by maintaining the position as much as possible, while the patient is instructed to hold his/her breath. However, the surface of the body is slippery and soft, so small movement is inevitable.

The virtual body and ultrasound data in our initial simulator are static and behave as a rigid object; i.e., the system does not model soft-tissue mechanics and deformation. This is very important in many medical care procedures, such as ultrasound guided central venous catheter (CVC) placement which is becoming a standard (Rothschild, 2001). The difference between a vein and an artery can be determined by compressibility and shape. While veins are completely compressible, have thinner walls, and ovoid shape, arteries are difficult to compress, have thicker walls, and are circular in shape (Fig. 3.3). Veins can collapse completely and may be difficult to identify if the patient is upright. Blind techniques which simply rely on anatomical landmarks to estimate location of vessels result with a complication rate of more than 15 percent (McGee and Gould, 2003).

3.2 Data Acquisition

In our system (Petrinec et al., 2011) the ultrasound volume is constructed using the freehand method. In other words, the ultrasound volume is acquired by sweeping a conventional 2D probe over the area of interest, and formed by stacking up the resulting brightness scans (B-scans). The data acquisition system consists of the following components (Fig. 3.4):

- SonoSite® M-Turbo™ portable ultrasound system with C60 5-2 MHz curved array transducer,
- InertiaCube3™ (InterSense, Inc.) motion tracker with three degrees of freedom (3-DOF),
- a laptop computer, and
- a digital camera.

The motion tracker is attached to the ultrasound probe and connected to a laptop computer through a USB port. The tracker detects a full 360 degree range of motion about each of 3 axes (roll, pitch, and yaw). Furthermore, a 180 Hz update rate virtually eliminates tracker-induced latency. Calibration is required to establish the rigid body transformation between the sensor and the B-scan. The rotational components of the transformation can be reduced to zero by attaching the axis of motion tracker parallel to the image of B-scan. In that case, the transformation consists of translation only, which can be manually measured. The motion of the probe is constrained to the rotation around a pivoting point preferably creating a fan scan, which is ideal for scanning through small acoustic windows such as the ribcage.

Figure 3.4: A view from camera which is used in data acquisition. A sonographer holds an ultrasound probe with motion tracker mounted on it. The ultrasound machine in the background shows captured B-scans. A laptop, which is not visible in the image, simultaneously captures the timestamps. Camera-captured video is used for the temporal registration of images and timestamps.

3.2.1 Data collection using a freehand fan scan method

The probe is moved steadily by hand, maintaining a constant contact pressure throughout the scan, with the extent of motion estimated by the user observation of the ultrasound B-scan screen until the region of interest is covered. An example of such a scan over the left upper quadrant (LUQ), capturing the left kydney and the spleen, is shown in Fig. 3.5. The computer asynchronously reads and stores the orientation of the probe and adds a timestamp, denoting the time at which the orientation is obtained. At the same time, ultrasound B-scans are being captured and stored in the Digital Imaging and Communications in Medicine (DICOM) standard format. The acquisition of ultrasound imagery and timestamps is started approximately at the same time. The precise delay between the two starts is determined from the video taken by the camera and used later for the precise registration of the dataset probe orientation and ultrasound imagery data streams.

One must keep in mind that there are many potential sources of error in the

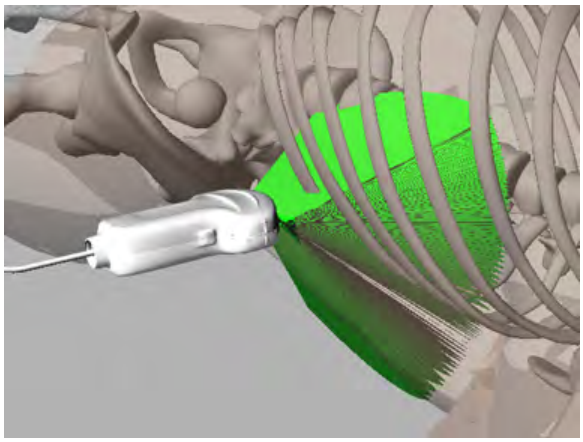


Figure 3.5: Visualization of a fan scan captured on PC using a motion tracker.

freehand 3D ultrasound system. Treece et al. (2003) divided errors in five groups: 1) errors in the B-scan images themselves, 2) the readings from the position sensor, 3) temporal matching of B-scans and positions, 4) location of the B-scan relative to the position reported by the sensor, and 5) errors in 3D reconstruction of B-scans. In addition to those errors, the final reconstruction may be distorted by the motion of internal structures of the body, such as when breathing or as the heart is beating, when the sweep motion is not fast enough to capture moving structure in a small approximately still period of time.

3.2.2 Data collection using freehand linear scan method

Despite many potential sources of error, the fan scan method previously described results with high quality 3D volumes when the sources of error are avoided. However, the size and shape of the volume are inadequate for the simulation of deformation when the virtual probe has to move away from the pivoting point. Then, instead of fanning around a fixed point, the 3D volume can created from a linear scan. For 3D reconstruction from the linear scan, it is not enough to know the position and orientation of sensor, but also the sensor must be calibrated; i.e., the position and the exact rigid body transformation between the sensor and

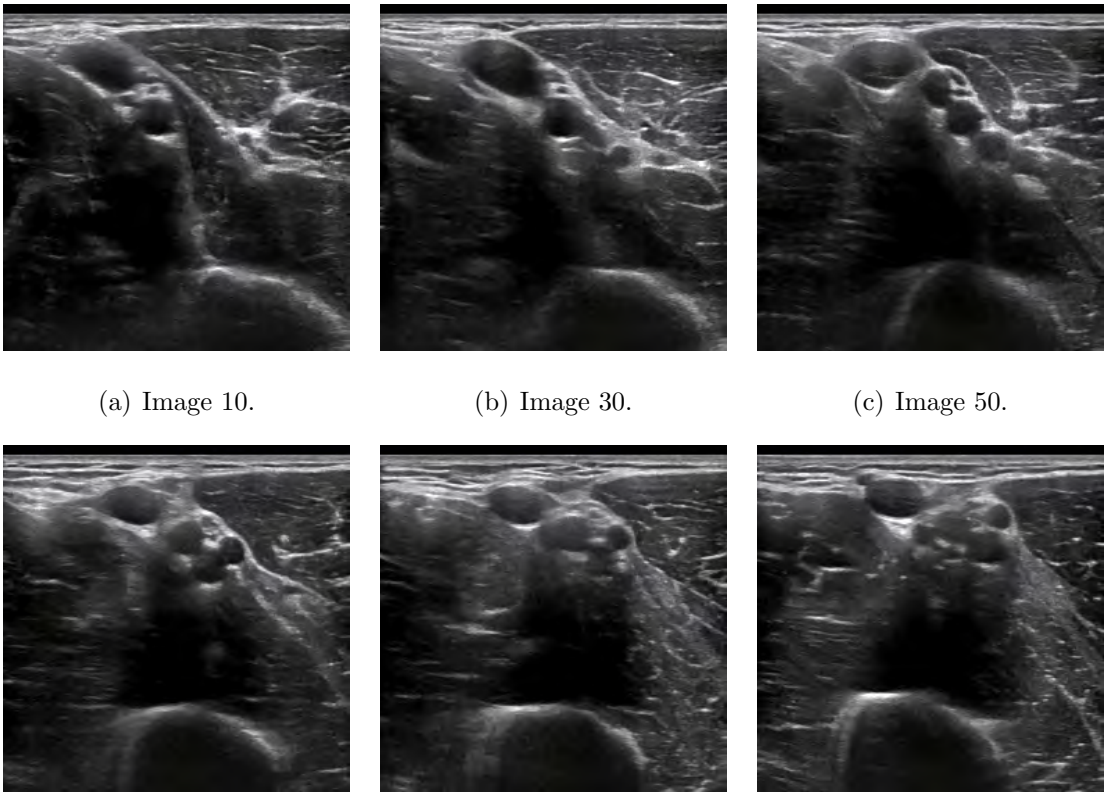
the B-scan must be known. Prager et al. (1998) use a cross-wire phantom of known geometry filled with water. First, they scan the crossing from multiple probe positions and orientation. Then they detect the position of the crossing in the acquired B-scans. Finally, they find the transformation matrix by solving an overdetermined system of nonlinear homogeneous equations using Levenberg-Marquardt algorithm (More, 1978).

Calibration processes require calibration rigs and are labor intensive and time consuming. However, rectangular 3D volumetric data can be acquired even without a sensor attached to the probe, and many ultrasound machines have that functionality embedded. This requires strict and precise probe motion; i.e., a linear scan must be acquired by sweeping linearly across the skin while making sure that 1) the orientation of the probe beam is perpendicular to the direction of probe, 2) the motion is in a straight line, 3) that the velocity of the motion is constant, and 4) the velocity of the motion tracker is such that the distance between acquired 2D ultrasound images is approximately equal to the resolution of the images. The latter constraint simply means that the scan must be slow enough to ensure good quality of slices sampled in the direction orthogonal to the direction of the linear scan.

Fig. 3.6 shows several slices from a linear scan acquired across the vessels, nerves and muscles of the left upper arm. The scan is acquired using a GE Logiq E9 machine with a linear transducer ML 6-15. The probe beam is 5 cm wide and the depth is set to 4.5 cm. The linear scan is 10 cm long and consists of 136 images.

3.3 Reconstruction of the Volumetric Data

In our approach, data reconstruction is based on the Pixel-Based Methods (PBM) described in (Solberg et al., 2007), and it consists of four stages:



(a) Image 10. (b) Image 30. (c) Image 50.

(d) Image 70. (e) Image 90. (f) Image 110.

Figure 3.6: Images from a linear sweep scan applied across the left upper arm.

1. *Preprocessing of images and timestamps.* A number of scaling, alignment, offset, and image modifications are applied to provide a consistent and organized baseline for further processing. This results in unique timestamps for each preprocessed image and an identified region of interest (ROI).

2. *Volume size computation.* The geometric relationship between the ROI and source data geometric parameters, including pivot points and orientation angles, is used to compute an affine transformation per image, which is then applied to each image to project it as the boundary of a slice in 3D Cartesian space in voxel units.

3. *Pixel-based reconstruction.* Two blocks of memory are utilized in the reconstruction process: one for volume, and the other for a mask which will keep track of which voxels were set and how many times each voxel was visited. For all images, two adjacent images are taken and projected into the volume by applying corresponding transformations to all image pixels. Also, all voxels on the line between corresponding pixels of the two images are linearly interpolated (Fig. 3.7). Voxels visited multiple times are averaged.

4. *Iterative reconstruction.* After pixel-based reconstruction there may be voxels that were not masked. Those voxels are set to the average value of adjacent voxels and the process is iterated until all voxels are assigned values.

In the simulation of ultrasound B-scans from 3D volumes, the virtual probe is positioned at the origin of the reconstructed 3D volume. Given the yaw, pitch, and roll angles of the probe, which are received from a motion tracker embedded in the peripheral probe held by the simulator user, the corresponding transformation is computed and applied to the coordinates of pixels in an image, which results in the coordinate of the corresponding voxel. The coordinate falls inside a voxel cube.

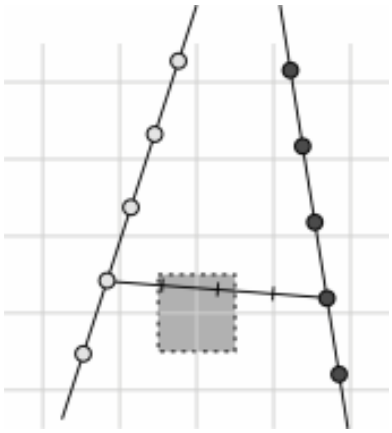


Figure 3.7: PBM reconstruction method. 3D voxel grid is shown as 2D grid symbolizing the centers of the voxels. 2D input images are illustrated as lines where the pixel centers are marked with circles.

To obtain the voxel intensity, trilinear interpolation is used, and the corresponding pixel is set to the value of that voxel.

Our approach naturally yields realistic ultrasound images, because images are sliced from volumes created from case-specific (real-patient) data. A comparison of an original and a simulated image is shown in Fig. 3.8.

An example of two slices, one in the scaning plane and the other in the orthogonal plane, from volumetric data reconstructed from the linear scan is shown in Fig. 3.9(a) and 3.9(b), respectively. Fig. 3.9(a) corresponds to Fig. 3.6(d). The orthogonal reconstructed slice in Fig. 3.9(b) demonstrates that high quality volumetric data can be acquired even using a sensor-less freehand linear scan method; i.e., without motorized transducers or probes equipped with 6-DOF sensors.

3.4 Segmentation

We will later need to quickly segment arteries, veins, nerves, and bones in volumetric ultrasound data. Vessels, nerves, and bones appear as smooth tubular

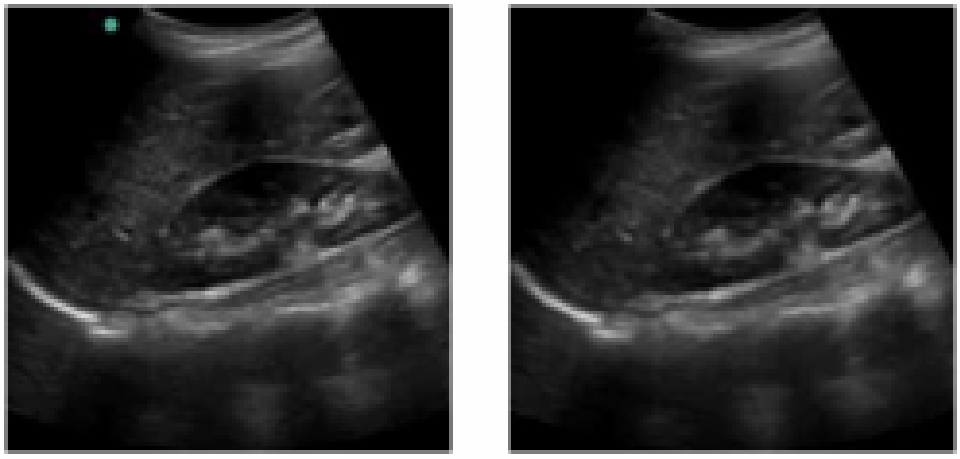
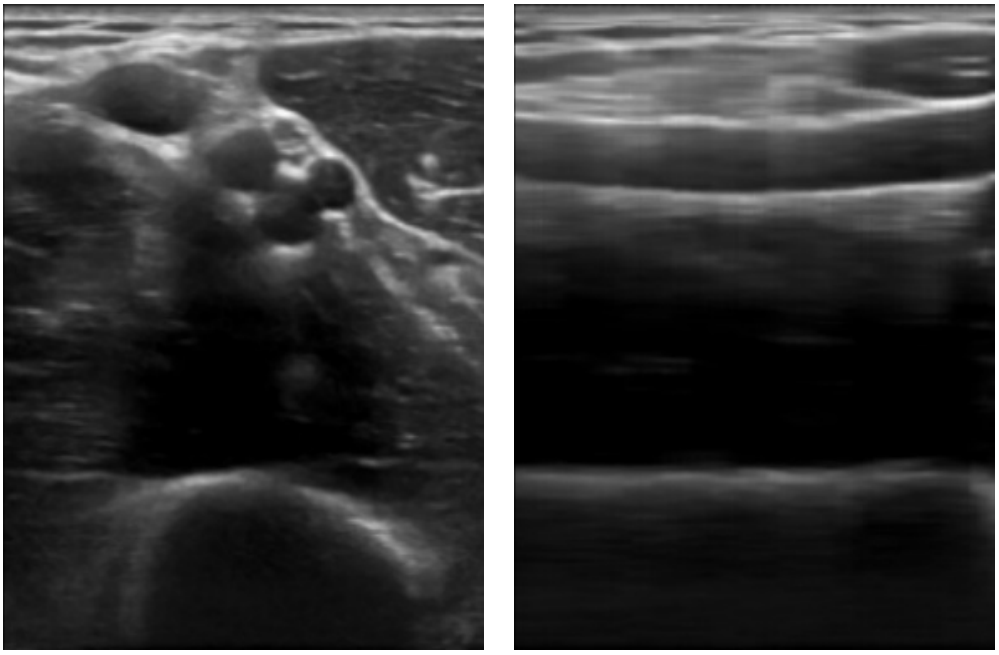


Figure 3.8: B-scans of Morison's Pouch. Comparison of an original scan used in volume reconstruction (left) and a scan simulated from a reconstructed volume (right).



(a) Slice from reconstructed volume.

(b) Orthogonal slice from reconstructed volume.

Figure 3.9: An example of 3D reconstruction from a linear scan of the left upper arm. Visible structures from top to bottom: basilic vein, median nerve, brachial veins, brachial artery, and the humerus.

structures mostly parallel with the skin, and cross-sections of those structures do not change abruptly from slice to slice. Driven by those two observations, we created a tool which allowed us to speed up segmentation simply by skipping slices and interpolating skipped slices between the skipped slices. In another words, the idea is that the user draws a curve on the boundary of a vessel in one slice, then skips several slices, draws a curve on the boundary of the same vessel in another slice, and then the tool automatically interpolates curves on slices between.

Given the two curves, a source curve and a destination curve, the interpolation is done as follows. First, we subdivide (resample) both curves into the same number of segments. Then, we find the center of curves and select the same starting points (points at the same arbitrary angle with respect to the center). Then, for each point in the starting curve, we find a corresponding point in the destination curve and use linear interpolation to find points for all in-between curves. Fig. 3.10 shows an example of shape morphing. A red triangle is morphed into a green square. Piecewise linear contours (polygons) represent interpolated shapes (irregular convex hexagons) where one of them in the middle is filled.

More sophisticated curve (shape) morphing, also known as shape blending, can be applied. For example, Yang and Feng (2009) interpolate the path along the curve based on feature correspondence. Similarly to our method, they resample the source and destination curves, but then they detect feature points which delimit each curve into meaningful parts of the curve. The algorithm automatically establishes one-to-one corresponence between feature points and computes the trajectory of corresponding features.

Fig. 3.11 illustrates our tool for semi-automatic segmentation. The user can draw a curve on the boundary of vessel in one slice, then skip slices that appear similar or change slowly, and then draw a curve on the boundary of a vessel in another slice. Then, when the user releases the mouse, all skipped slices are automatically filled by using linear curve morphing (curve interpolation).

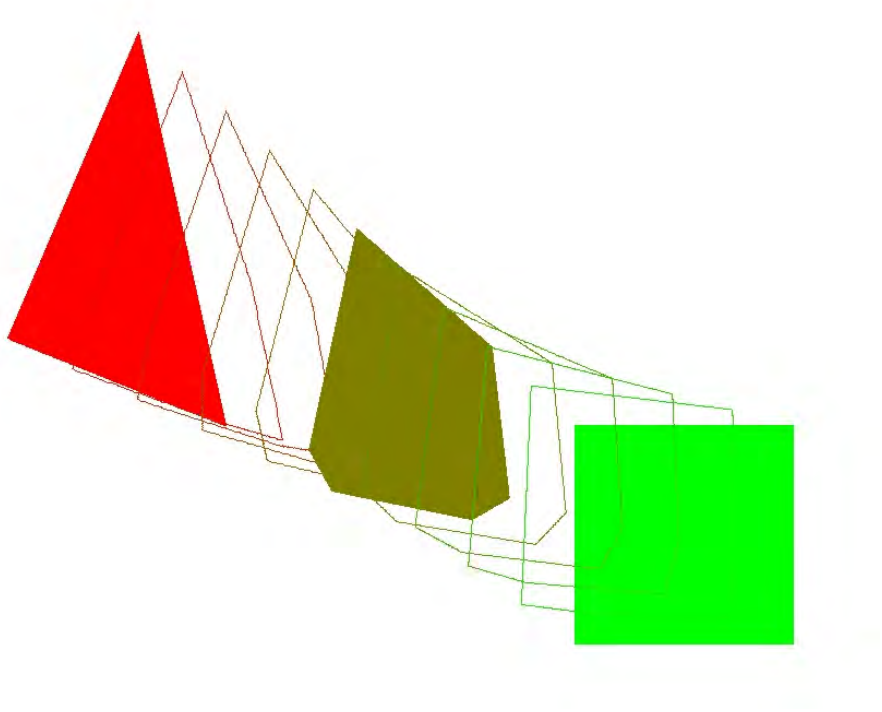
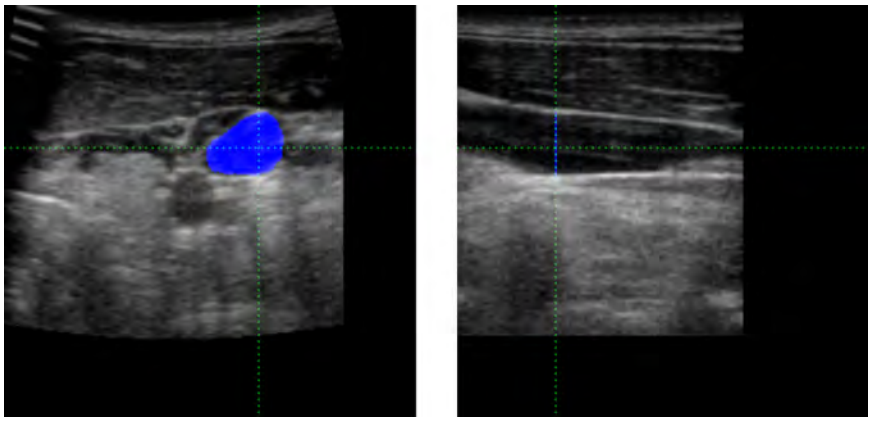


Figure 3.10: Shape morphing.

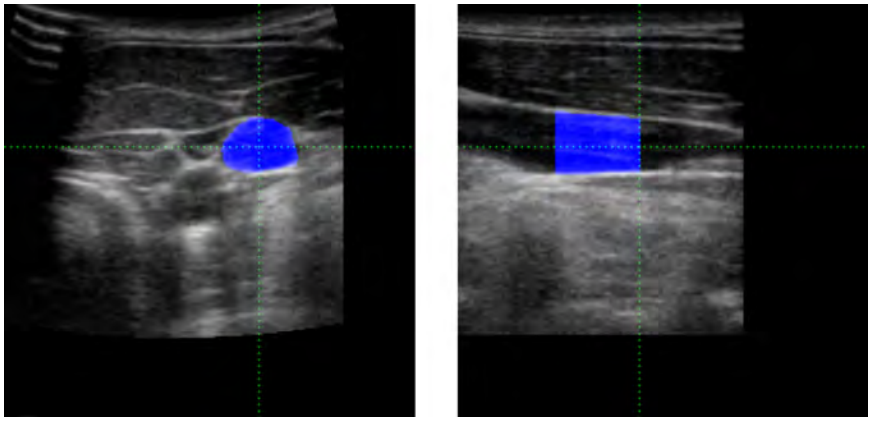
Using our method vascular structures can be quickly segmented from 3D ultrasound data. Fig. 3.12 shows three orthogonal slices from a 3D ultrasound image of the neck artery and vein. The veins are segmented in less than 10 minutes. The 3D reconstructed surface reveals branching of the vein.

3.5 Surface Reconstruction

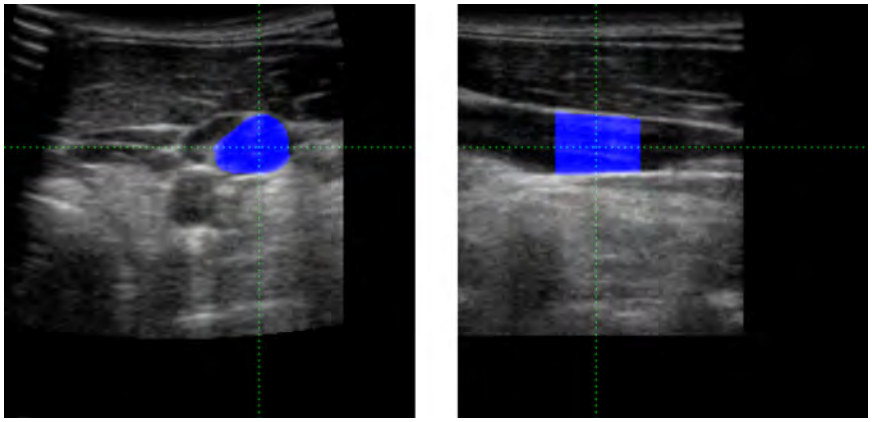
We have applied the marching cubes algorithm to a synthetic volumetric data of size $101 \times 101 \times 101$ with an embedded sphere. Fig. 3.13(a) shows the isosurface of a sphere extracted from a 3D volumetric data using the marching cubes algorithm. Initially the rough surface is refined by using Gaussian filtering (Fig. 3.13(b)). After 10 iterations the surface is still not entirely smooth. To obtain a smoother surface, more filtering iterations are necessary. However, the more iterations that are applied, the smaller the surface becomes. The shrinkage is demonstrated on



(a) First boundary drawn by the user.



(b) Second boundary drawn; in-between slices auto-interpolated.



(c) Interpolated slice.

Figure 3.11: An example of our semi-automatic segmentation with shape morphing. Two orthogonal slices from a 3D ultrasound data.

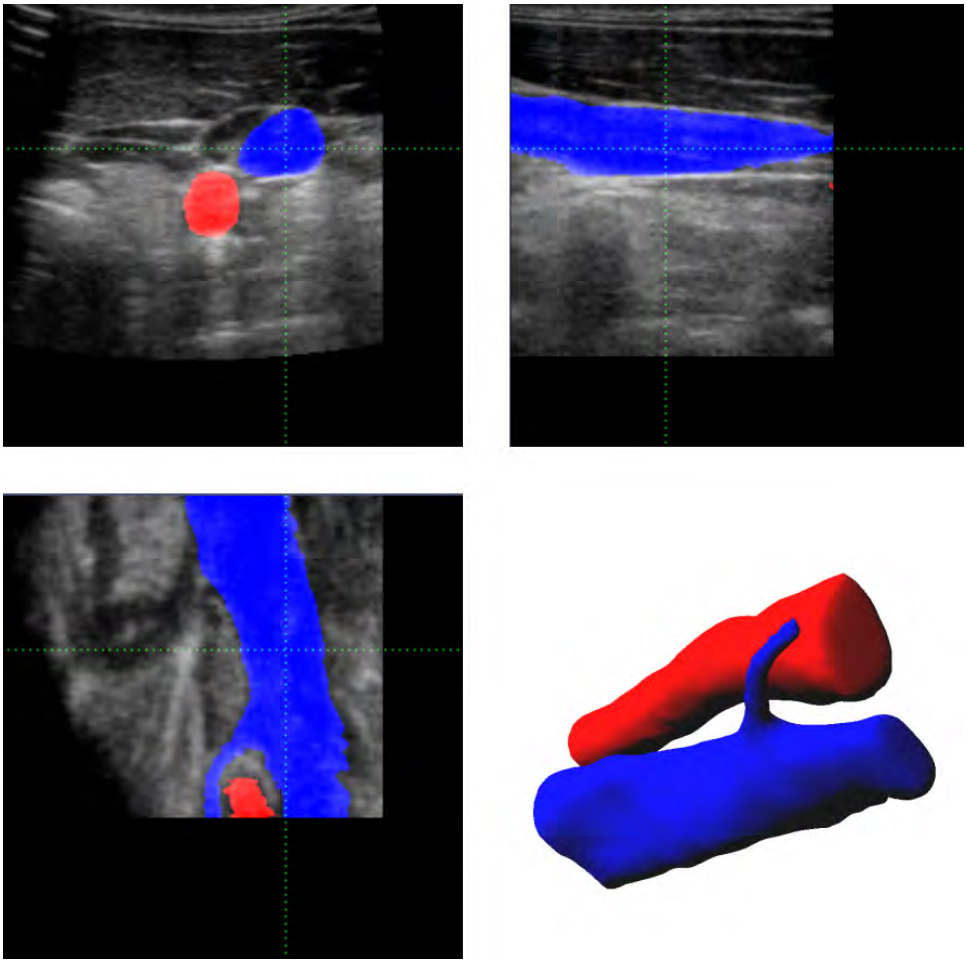


Figure 3.12: An example of the neck artery (red) and vein (blue) segmented using our semi-automatic segmentation with shape morphing. Three orthogonal slices from a 3D ultrasound data and 3D surface reconstruction.

(a) No filtering (b) 10 iterations of filtering

Figure 3.13: Isosurface of a sphere extracted from a 3D volumetric data without and with Gaussian filtering.

to overlapping spheres in Fig. 3.14. After 5000 iterations, the two spheres are barely visible.

Fig. 3.12 demonstrates the application of marching cubes and 10 iterations of filtering to the 3D reconstructed surface of the blood vessels of the neck (internal jugular vein and carotid artery).

(a) No filtering

(b) 10 iterations of filtering

(c) 100 iterations of filtering

(d) 1000 iterations of filtering

(e) 2000 iterations of filtering

(f) 3000 iterations of filtering

(g) 4000 iterations of filtering

(h) 5000 iterations of filtering

Figure 3.14: Surface shrinkage with Gaussian filtering.

CHAPTER 4

Simulation of Ultrasound Compression

In this chapter, we present our approach for the simulation of ultrasound compression and discuss the system components related to this functionality. We will also compare two deformable model simulation methods that we use for the simulation. Finally, we present our results.

4.1 Introduction

To develop a model for real-time simulation of ultrasound compression, we need to make a series of decisions. We need to decide on a geometric description of the object, a mathematical model of the elastic deformation, and a fast solution algorithm. We need a robust model with consistent and predictable behavior, and realistic simulation with visually pleasing results. To achieve those goals, we first evaluate two well-known deformable model simulation methods: mass-spring-damper systems (MSDS), and the finite element method (FEM) with a quasistatic solution of isotropic linear elastic materials with Cauchy strain.

The finite element method, which is described in Appendix B, is a well-known method for the simulation of volumetric solid soft bodies. Instead of reinventing the wheel, we use OpenTissue, an open-source library for physics-based animation. OpenTissue contains a collection of algorithms and data structures written in an object oriented style and optimized for interactive modeling and simulation. The interactive speed of the simulation is achieved by using quasi-static stress-strain

simulation; i.e., by using an iterative solver with the conjugate-gradient method. A comprehensive evaluation of other open-source interactive physics engines for simulation systems and game development is presented in Boeing and Bräunl (2007).

Mass-spring-damper systems, described in Section 2.4.2, are easy to implement and have a low computational complexity. We use explicit Euler numerical integration, which is fast, but unstable for large time steps. To ensure stability, we choose relatively small time steps and we overdamp the MSDS; i.e., we set the spring and damping constants so that the system remains stable even when a large external force is applied.

4.2 System Integration

The ultrasound training system for the simulation of compression is illustrated by the block diagram shown in Fig. 4.1. The system consists of six major components:

1. Database of real-patient volumetric ultrasound data

   The database contains a number of data sets acquired either with a 3D ultrasound probe, or data reconstructed from 2D ultrasound B-scans. The data, which is typically stored in the Digital Imaging and Communications in Medicine (DICOM) standard format, is preprocessed in such a way that only raw 3D data with spacial information is preserved. All other information, such as patient ID, name, age, or pathology, is stripped from the data sets. In addition to real-patient data, each data set contains segmentation data created using the semi-automated segmentation method described in 3.4.

2. Deformation of volumetric data simulator

   The simulator encapsulates two deformable model simulation methods: mass-spring-damper systems (MSDS), and the finite element method (FEM) with

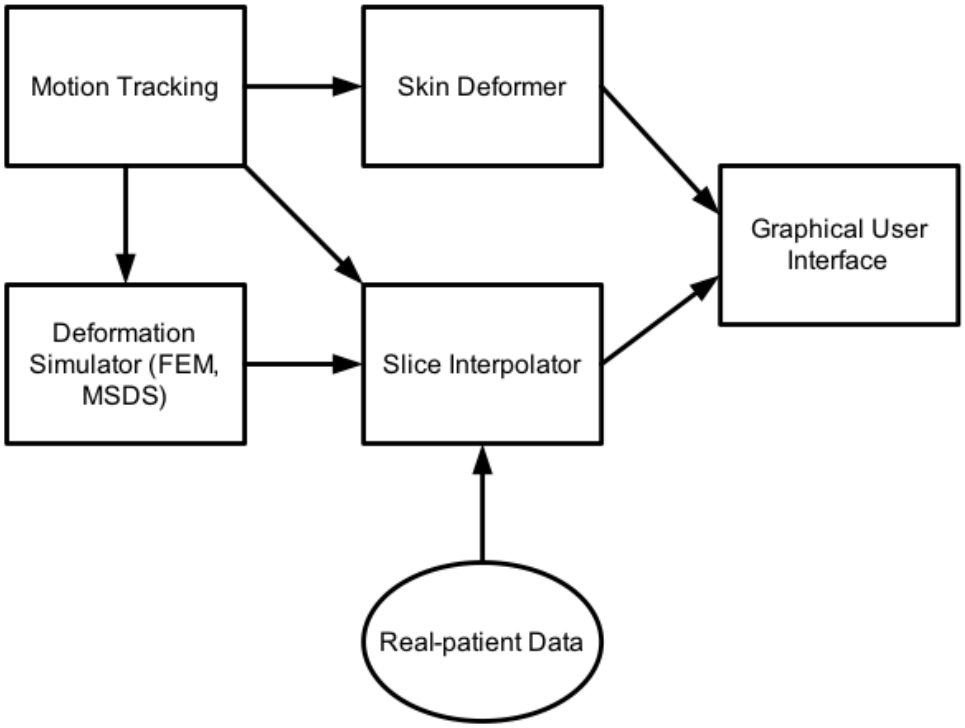


Figure 4.1: System block diagram.

a quasistatic solution of isotropic linear elastic materials with Cauchy strain. A unique interface created for the simulation libraries permits the real-time switching and comparison of the methods. The simulator takes the position and orientation of a collider, handles the collision, and computes the new state of the nodes in the mesh.

3. Slice interpolation module

   The slice interpolation module samples and interpolates the ultrasound image over the deformed mesh. It takes the probe position and orientation, probe beam parameters, volumetric data, and mesh in the undeformed and deformed state, to create a 2D ultrasound image that is displayed on the screen.

4. Skin deformation simulator

   The skin deformation simulator uses a mass-spring-damper system (MSDS)

to simulate a visually pleasing surface deformation. The edges of the triangular mesh are replaced with springs. To provide resistance and to ensure that surface returns to initial position when the collider is removed, all nodes of the mesh are anchored with zero-length springs.

5. Interactive user interface

   The user interface consists of an interactive 3D display with a deformable virtual body, virtual probes, and a simulated ultrasound slice; and dialogs for setting the FEM, MSDS, and simulation parameters (Figs. 4.2, 4.3, and 4.4). The user can rotate or zoom in/out the display, or rotate and move the probe with a mouse. The user can also select a case using a case list dialog.

   The deformable virtual body has a rigid skeleton, deformable skin, and simulated deformable structures embedded into the tissue (isosurfaces of vessels and nerves). The triangular mesh of the skin deforms when the virtual probe (collider) attempts to penetrate, and returns back to its original shape when the collider is not in contact. Isosurfaces of the segmented regions of real-patient data are embedded into the mesh and follow the deformation of the mesh. The user can look under the skin and examine how segmented structures move. For example, unlike arteries, veins under pressure will deform and even completely collapse.

6. Motion tracking system

   The motion tracking system controls the position and orientation of the virtual probe. Different systems may be used to control the probe; for example, a single camera optical tracking system, a 6-DOF mouse, or a haptic device.

The arrows in the block diagram (Fig. 4.1) show the flow of information. The motion tracker moves the virtual probe which interacts with the tissue deformation

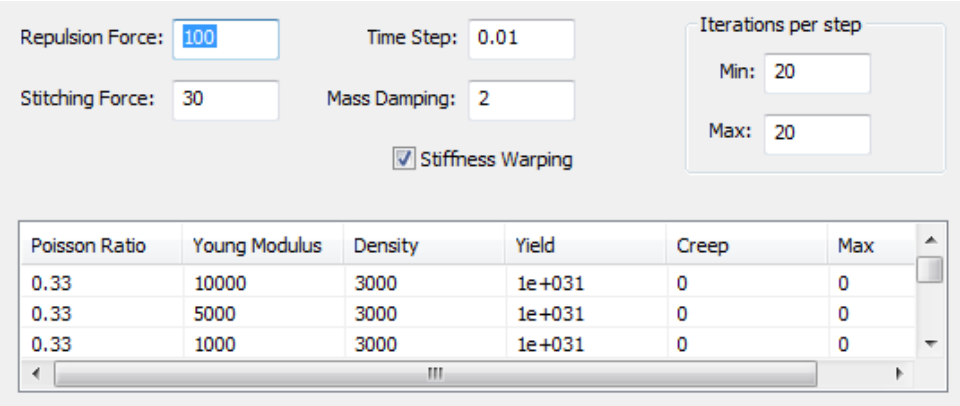


Figure 4.2: Settings dialog for the FEM parameters.

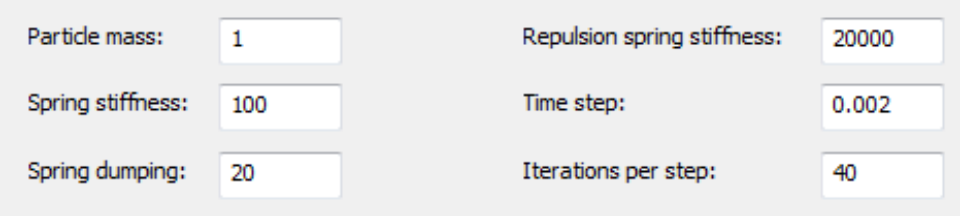


Figure 4.3: Settings dialog for the MSDS parameters.

simulator and the skin deformer. The probe, approximated by a sphere, applies force to mesh nodes in the collision volume (inside the sphere). The (un)deformed meshes are passed to the slice interpolator. In addition to the tissue mesh, the interpolator also takes the probe position and orientation from the motion tracker, and case-specific real-patient data from a data base. Then it projects slice pixels into the deformed mesh, takes corresponding points in the undeformed state, and sets pixels intensities to voxel intensities interpolated from the volumetric real-patient data. The interpolated slice is shown in the graphical user interface display as a 2D image. The skin deformer modifies the nodes and surface normals of the skin mesh.

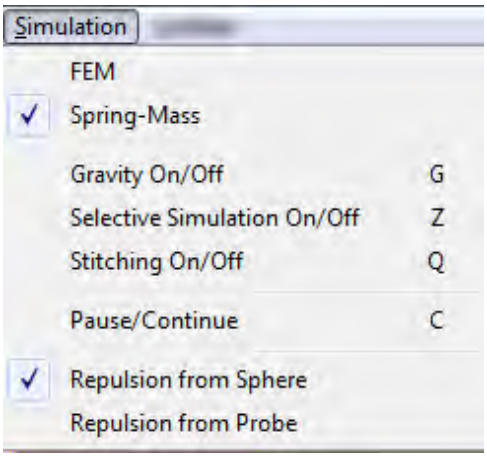


Figure 4.4: Settings menu for the simulation parameters.

4.3 Simulation of Skin Deformation

In our initial implementation, we embedded the skin of the virtual patient directly into the soft-tissue deformation mesh. When we applied pressure to the mesh, it caused the skin to bulge on the edges where the skin intersected the mesh. Even though the deformation of the skin inside the mesh looked satisfactory, the abrupt changes in the skin exposed the rectangular shape of the mesh, which was not visually pleasing.

To prevent sudden and unnatural skin deformation around the intersection with the tissue mesh, we decided to simulate the skin separately from the tissue. The skin is simulated as a MSDS with a single-layer mesh. We use the same collision model for both systems; i.e., when the probe is in contact, the collider, which approximates the shape of the probe, applies a repulsion force to the nodes of both meshes—the skin and the soft-tissue mesh. As a result, the skin is nicely deformed under contact with the probe, and there are no visually unpleasing deformations where the skin crosses the tissue mesh.

4.4 Comparison between FEM and MSDS

In real-time soft-tissue simulations, the speed of deformation simulation algorithms is more important than accuracy. Rather than having scientifically accurate soft-tissue simulation which may take minutes or hours per frame, we are more interested in less accurate methods which can run in real-time and which can provide visually pleasing results. The FEM with a quasistatic solution of isotropic linear elastic materials with Cauchy strain is often used in real-time systems, as well as the MSDS method.

In order to evaluate the two methods for ultrasound simulation in deformable tissue, we must make sure that both simulations have approximately the same dynamic response. Thus, the problem is how to achieve similar dynamic response with the two fundamentally different systems.

To achieve similar dynamic response, we create FEM and MSDS of the same size with the same number of nodes as follows: First, we create an FEM system and set its material properties to produce a stable, over-damped dynamic response. Then we apply the following algorithm to create an MSDS with a similar dynamic response:

Algorithm FEMToMSDS(*tets*,*springs*)

Inputs:

- *tets*: List of FEM tetrahedra

Outputs:

- *springs*: List of MSDS springs

Algorithm:

1. For all tetrahedra T in *tets*
2. For all edges E in T
3. Let p_A and p_B be the vertex coordinates of edge E
4. Let S_E be a spring between p_A and p_B
5. Set the spring constant of S_E: $k_s(S_E) := T_{young} * 0.002$

```
6.          Set damping factor of S_E: k_d(S_E) := 0.1 * k_s(S_E)
7.          Find the spring Q in springs between p_A and p_B
8.          If Q does not exist in springs then
9.             Add S_E to springs
10.         Else
11.            Update the spring constant of Q: k_s(Q) := k_s(Q) + k_s(S_E)
12.            Update the damping constant of Q: k_d(Q) := k_d(Q) + k_d(S_E)
13.         EndIf
14.      EndFor
15.   EndFor
```

The algorithm converts all the edges of the FEM mesh tetrahedra into springs. Spring constants are set to the Young modulus of the tetrahedra multiplied by a constant value, which we empirically found to yield similar dynamic response. Damping constants are set to the spring constant multiplied by an empirically found constant which results in stable, over-damped dynamic response. Duplicate springs are merged by summing spring and damping constants. Fig. 4.5 illustrates the process on two adjacent tetrahedra with connected faces.

In addition to yielding a similar dynamic response, the algorithm creates an MSDS of the same topology as the FEM, with the same number of nodes, and preserves the size.

4.5 Slicing the Deformed Mesh

Given a regular mesh and volumetric data embedded in the mesh, the challenge is to synthesize 2D images (slices) from the deformed mesh in real-time. For example, Fig. 4.6(a) shows a regular $5 \times 5 \times 5$ mesh in undeformed configuration. It comprises 125 nodes and 320 conforming tetrahedrons. Fig. 4.6(b) shows the mesh deformed when a probe applies pressure on the top side of mesh. A slice plane cuts a number of tetrahedrons in the deformed mesh. The cut consists of

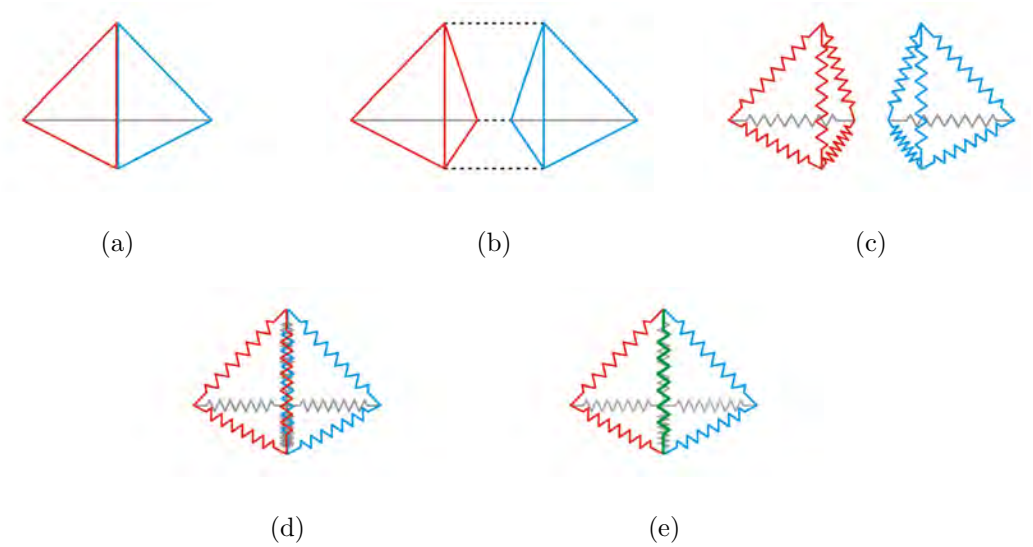
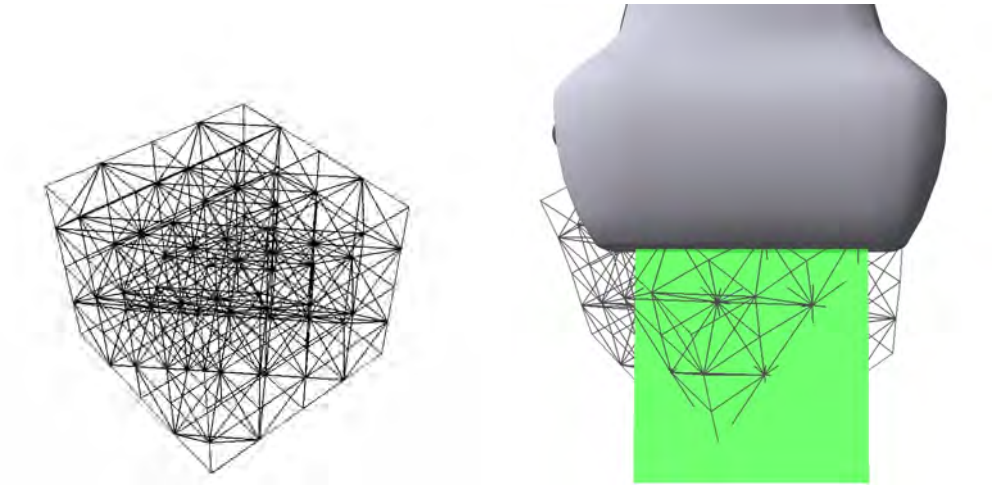


Figure 4.5: Converting FEM into mass-spring-damper system.

triangles and rectangles, as shown in various colors in Fig. 4.6(c).

For each pixel in the slice, we can easily find the corresponding voxel intensity. First, we must find the exact 3D position of the pixel in the deformed mesh. Once can precompute the slice transformation and apply it to the pixel. Then, one must find which tetrahedron contains the pixel, compute barycentric coordinates, and use these barycentric coordinate to find a 3D coordinate in the undeformed mesh. Finally, one must sample the volumetric data to obtain the voxel intensity and set the pixel to that intensity.

Doing this for every pixel is inefficient and too slow for real-time applications. Goksel and Salcudean (2009) proposed an algorithm for fast ultrasound image simulation. They exploit the fact that mesh tetrahedrons are much bigger than image pixels, so numerous image pixels are enclosed by a tetrahedra cut by the image. They build a data structure where for each pixel they store the index of the tetrahedron which intersects the image. In the first pass, they look for all intersections of faces with the image. A pixel may be intersected by multiple faces of the tetrahedra. To assign the correct element to the pixel, they topologically sort all tetrahedra cross sections, top to bottom, or column-by-column. Subse-



(a) Undeformed mesh.

(b) Mesh deformed by probe.

(c) Slice cut from deformed mesh.

Figure 4.6: Slice cutting.

quently, they use a scan-line approach to find tetrahedra of pixels with unassigned tetrahedra.

Our approach

In our approach, we first create a list of tetrahedrons which intersect bounding boxes of deformed tetrahedrons. Then, we process slice pixels row-by-row. In each row, we take a pixel P_a at the far left and P_z at the far right. If they fall into same tetrahedron, then we linearly sample voxels from interval $[P_a, P_z]$ in undeformed configuration. Otherwise, we take a pixel P_m in the middle and recursively apply the same to intervals $[P_a, P_{m-1}]$ and $[P_m, P_z]$ while $a < z - 1$. To speed up the search for tetrahedrons, we split the slice into M regions, and for each region we create a list of tetrahedrons crossing the region. We empirically found that we achieve the best results when M equals 32.

To summarize, the major differences between our approach and the approach of Goksel and Salcudean (2009) is that we scan and set the image pixels row-by-row, we use row subdivision to find and set pixels inside the same tetrahedron, and we do not use topological sorting.

4.6 Deformable skin

Pressing the ultrasound probe against the skin may cause significant and visible deformation to the skin and underlying tissue. In order to make the simulation of compression as realistic as possible, it is desirable that the skin of the virtual body also be deformable.

The skin of the virtual body is a surface which consists of a number of triangular faces. The faces share edges and vertices with adjacent faces. We simulated the deformation using MSDS. Vertexes of the triangles are represented with particles, and edges of the triangles are represented with springs. The virtual probe,

when in contact with the surface, applies external force to particles in collision, forcing them to move away. However, this forces the whole surface to drift away from initial state if it is not properly constrained.

One way to prevent the drift is by anchoring the surface to the skeleton with a mesh of springs. Terzopoulos and Waters (1990) create a physically-based 3D model of the human face with three layers of mass-spring elements representing the muscle layer and two layers of skin (dermis and epidermis). The bottom surface of the muscle layer is fixed in bone. Facial expressions are controlled by muscle contraction. A benefit of this approach is that the surface deforms together with the skeleton and muscles; e.g., when the skeleton (jaw) moves, the skin (face) follows it.

In our work, the skeleton is rigid and the muscles are not active, so modeling the skin with an underlying mesh fixed in bone would introduce unnecessary complexity and take more processing time. Instead, we prevent the remainder of the skin from drifting by anchoring the entire surface to the initial position by attaching zero-length springs to all particles. This allows the surface to move away from a colliding object and ensures that the entire surface returns to its initial state when the external forces are removed. For bodies with rigid skeletons, this results in realistic simulation and allows for simple and efficient implementation.

Fig. 4.7 demonstrates the interaction between a sphere and a surface. An external (repulsion) force is applied to the surface particles that are in collision with the sphere. Zero-length springs resist the external forces, causing the surface to deform away from the sphere. This is also demonstrated in Fig. 4.8. A part of the surface stays in contact. The large repulsion force will push away the surface from the sphere surface, but may cause unstable behavior (oscillations or divergence).

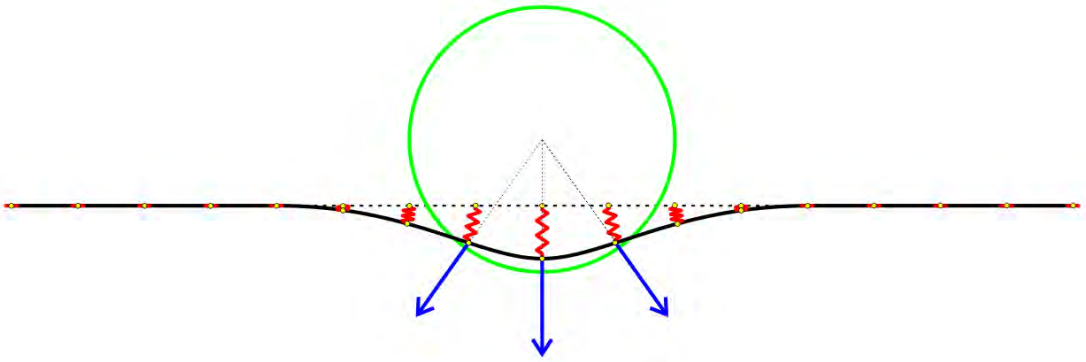


Figure 4.7: A cross-section of a surface (black) deformed by a sphere (green). External forces (blue) are applied to nodes in collision. Zero-length springs (red) pull the surface back to its initial shape (horizontal dashed line).

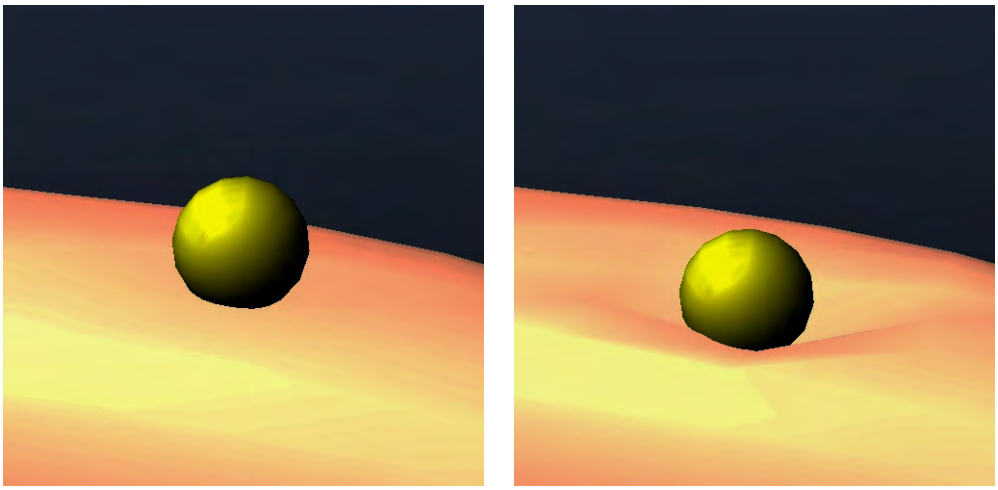


Figure 4.8: A 3D example of the interaction between the skin and a spherical object.

4.7 Simulation Results and Discussion

All the results described in this section were simulated and rendered in real-time on a laptop with a 2.8 GHz Intel® Core™ 2 Duo CPU, and an nVidia® GeForce® 9600 M GT graphics card. We generated the tetrahedral regular meshes for the FEM and MSDS off-line. To create the surfaces of vessels and nerves, we used the approach described in Sections 3.4 and 3.5.

The experiment was in two parts. In both parts we used the MSDS and FEM, for the integration and we used a time-step of 0.01 seconds, and we ran 20 iterations per frame; i.e., the FEM ran 20 conjugate gradient iterations per frame, and the MSDS ran 20 explicit Euler iterations per frame. The parameters of the MSDS were empirically set to match the dynamics of the FEM as described in Section 4.4. We empirically found that similar dynamics is achieved when the spring constant is set to 0.002 times the Young modulus of the corresponding tetrahedron material and when the spring damping constant is set to 0.1 times the spring constant.

In the first part of the experiment, we simulated ultrasound images in deformable tissue using the MSDS and FEM. We measured the average time to simulate mesh deformation, to respond to collision, and to sample the B-scan image against four different mesh sizes, as shown in Table 4.1. The results show that the deformation simulation time grows linearly with the number of nodes in the mesh for the MSDS, and exponentially for the FEM. The collision detection and response took less than one percent of the time and it is negligible. The volume sampling time also increases with the mesh size, but the ratio of the time versus the number of nodes declines as the number of nodes increases.

In the second part of the experiment, we investigated how the visualization of segmented data (e.g., vessels, nerves, bones, etc.) affects the overall simulation time. Similarly as in the first part, we simulated ultrasound images in deformable

Table 4.1: Simulation times.

Mesh	20x10x10	24x12x12	28x14x14	32x16x16
Number of nodes	2000	3456	5488	8192
Number of tetrahedra	7695	13915	22815	34875
Simulation using MSDS (FEM) [ms]	30 (85)	50 (180)	80 (380)	130 (550)
Collision detection [ms]	0.3	0.4	0.6	0.9
Volume sampling [ms]	40	56	65	90

Table 4.2: Simulation frame rates, in frames per second (FPS).

Mesh	20x10x10	24x12x12	28x14x14	32x16x16
Without vessels using MSDS (FEM)	7.6 (5.8)	6.4 (3.5)	5.9 (1.9)	4.1 (1.4)
With vessles using MSDS (FEM)	4.6 (3.6)	4.2 (2.8)	3.5 (1.6)	2.9 (1.3)

tissue using the MSDS and FEM, but we measured the number of frames per second (fps) versus four different mesh sizes, and against the simulation with and without visualization of the segmented data. Fig. 4.9 shows an example of isosurfaces of nerves, vessels, and bone in the upper arm, which we used in the simulation. It took about 4 seconds to create five isosurfaces with the total of 243224 triangles. The isosurfaces are embedded in the mesh and they deform when mesh deforms. The results in Table 4.2 show average simulation frame rates with and without the visualization of vessels.

In both parts, we ran simulations on a laptop with a 2.8 GHz Intel® Core™ 2 Duo CPU, and an nVidia® GeForce® 9600 M GT graphics card. The results demonstrate our method yields visually pleasing simulation of soft-tissue deformation in real-time.

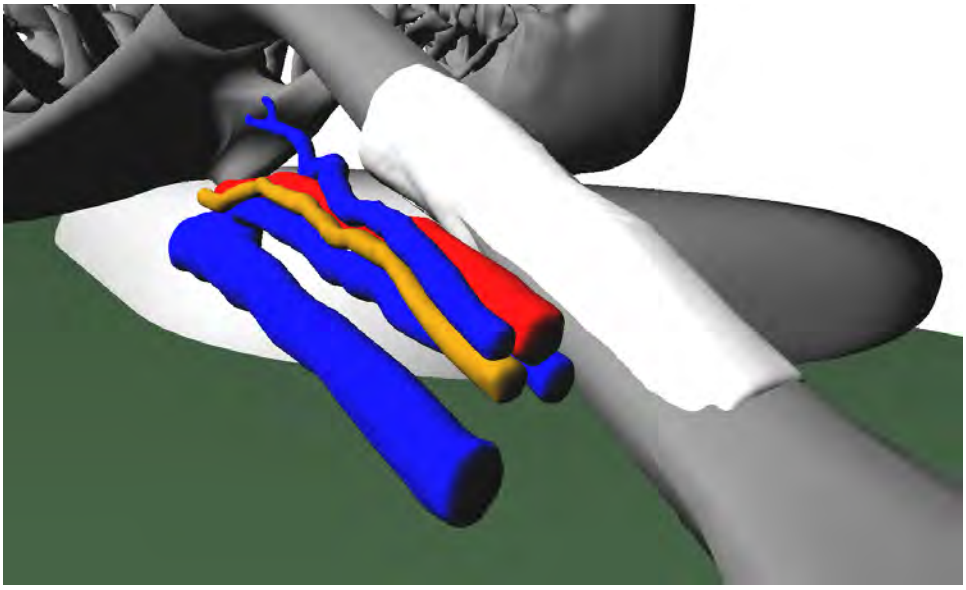


Figure 4.9: Isosurfaces of ultrasound data roughly aligned with the skeleton of the virtual model.

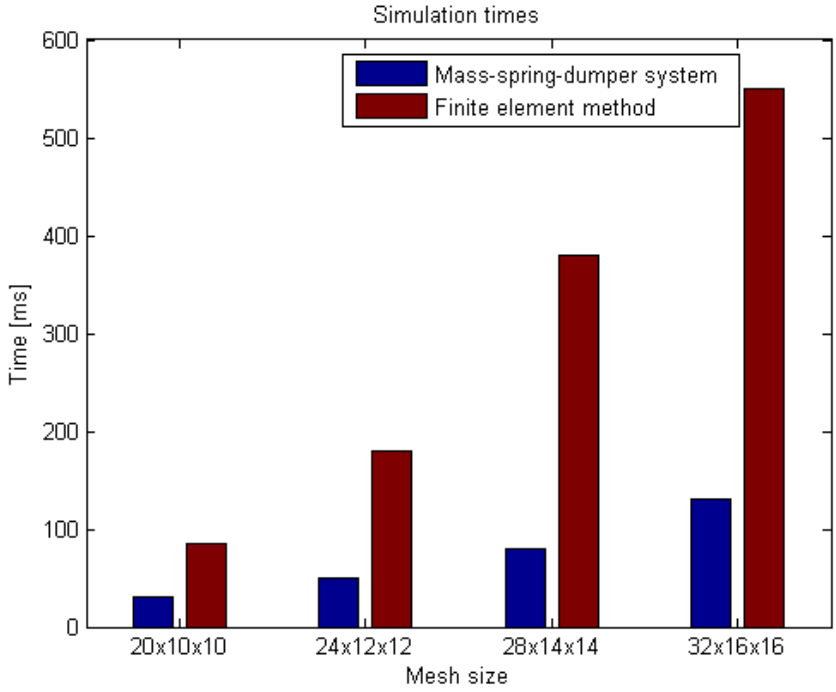


Figure 4.10: Comparison of simulation times: mass-spring-damper system vs. the finite element method.

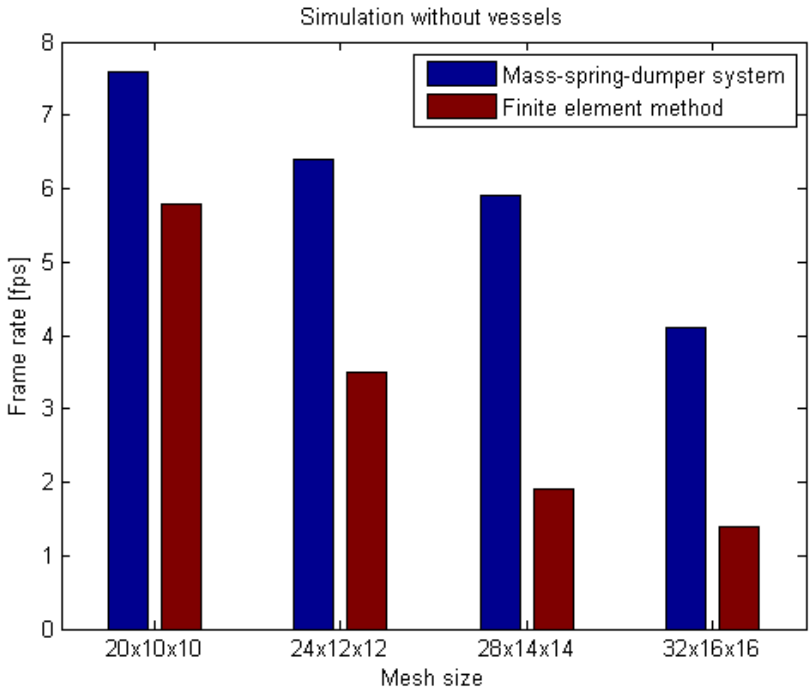


Figure 4.11: Comparison of frame rates in the simulation without vessels: mass-spring-damper system vs. the finite element method.

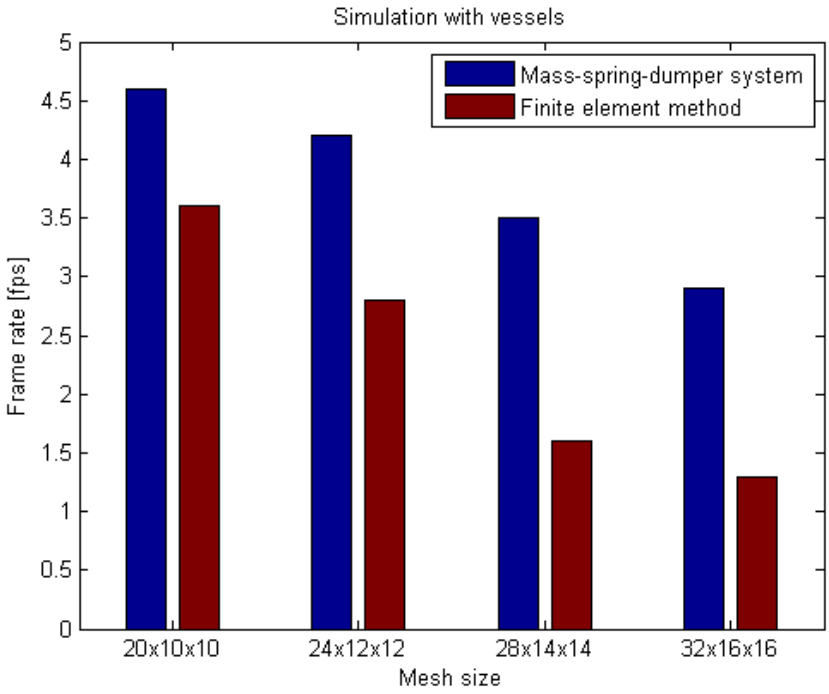


Figure 4.12: Comparison of frame rates in the simulation with vessels: mass-spring-damper system vs. the finite element method.

CHAPTER 5

Volume Stitching

Ultrasound scans of body parts, such as arms or legs, always have rigid areas, such as bones, which do not deform under pressure. As a consequence, nodes of the finite element mesh that fall inside those areas can be fixed. Moreover, external force or pressure applied to one part of the body typically does not deform other parts of body when the skeleton is fixed. For example, a small pressure applied to upper arm deforms the skin and soft tissue of the upper arm, but does not affect tissue in the lower arm, so it seems natural to simulate deformation only in areas where the external force is applied.

5.1 Volume Stitching Idea

The idea behind volume stitching is simple: We divide a large mesh into smaller overlapping meshes, attach zero-length springs to the overlapping nodes, and simulate deformations on the smaller meshes in parallel. Fig. 5.1 illustrates the idea on a 2D mesh, which consists of 70 nodes arranged in a square grid with 14 columns and 5 rows. Nodes are connected with springs in horizontal, vertical, and diagonal directions (Fig. 5.1(a)). First, the mesh is split in two parts, left and right (Fig. 5.1(b)). The last two columns of the left part overlap with the first two columns of the right part. Then, the overlapping nodes are connected with zero-length springs (Fig. 5.1(c)). Note that we overlap two columns instead of one in order to preserve the continuity.

The task of zero-length springs is to attract separated nodes. When two overlapping nodes connected with a zero-length spring are separated, then the zero-length spring applies a force proportional to the distance between the nodes. The farther the nodes are, the bigger attraction (stitching) force.

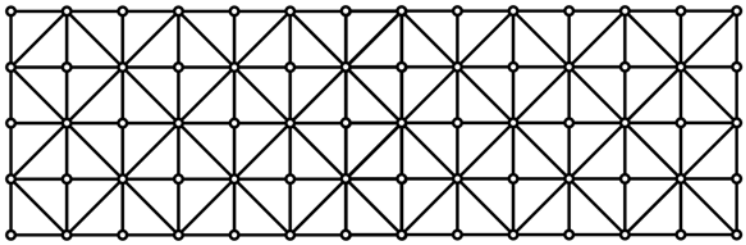
The simulation of deformation with volume stitching is done in two alternating steps:

1. Simulation of deformation
2. Recalculation of stitching forces

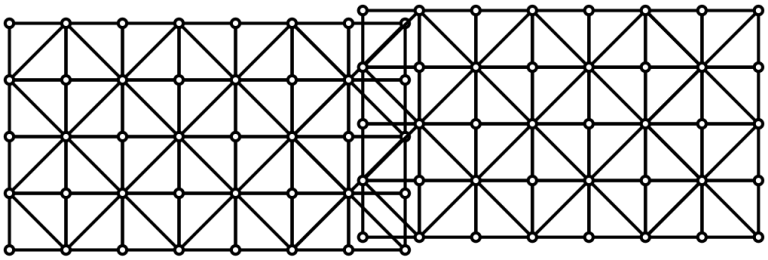
The deformation of meshes can be simulated sequentionally (mesh-by-mesh), or in parallel (in multiple threads). Each mesh deformation is simulated iteratively by using the FEM or the MSDS. In the odd step, run several iterations of the mesh deformation simulation. In the even step, compute stitching forces and set them as external forces acting on overlapping nodes. Then, repeat the steps.

At equilibrium, the sum of internal, external, and stitching forces acting on the overlapping nodes is zero. As a result, overlapping nodes will not fully overlap when the external force is applied to the mesh. This is illustrated in Fig. 5.2(a), where a spherical collider is forcing the right mesh to deform, and zero-length springs are pulling the meshes together. The bottom nodes of both meshes are anchored. There is a gap between the meshes because the stitching force is not strong enough to pull the meshes together. A stronger force will pull the meshes closer, but never to zero distance, because of linear dependence on the distance. The gap can be removed by displaying nodes in the middle; i.e., mid-points of the overlapping nodes, as illustrated in Fig. 5.2(b).

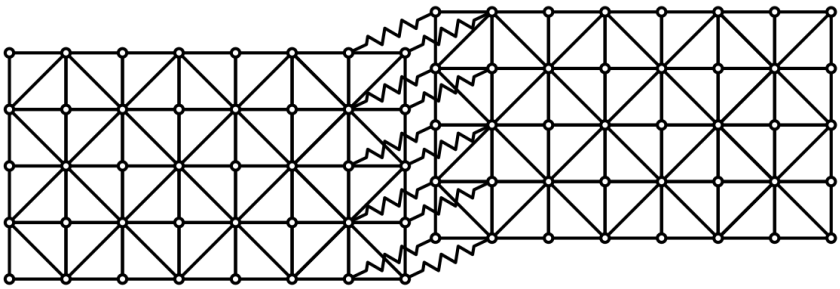
For a large number of meshes, we can automatically disable the simulation of meshes where no external force or force from zero-length springs is applied, and that can significantly reduce the simulation time. We can also select inactive meshes at random and allow them to come to fully to steady-state.



(a) Oginal mesh with 14 x 5 nodes.

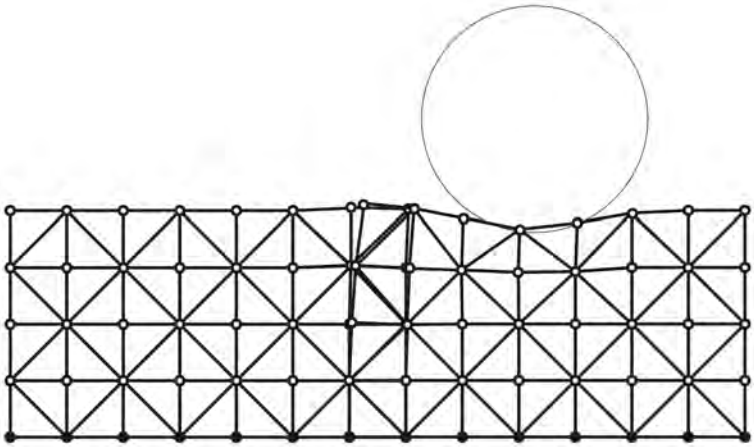


(b) Original mesh divided in two with 8 x 5 nodes each.

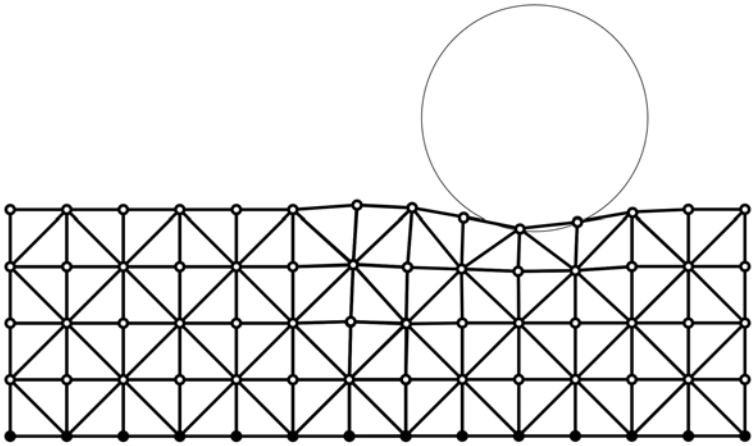


(c) Meshes connected with zero-length springs.

Figure 5.1: Volume stitching.



(a) Gap between meshes



(b) Gap removed by using mid-points

Figure 5.2: Volume stitching gap fix.

5.2 Results and Discussion

The volume stitching algorithm described in this chapter has been implemented and tested on a laptop with a 2.8 GHz Intel® Core™ 2 Duo CPU, and an nVidia® GeForce® 9600 M GT graphics card.

In our first test, we first simulated a uniform mesh using the FEM and MSDS. The size of the mesh was $52 \times 52 \times 12$. As shown in Table 5.1, the FEM simulation took 2200 ms, and the MSDS simulation took 500 ms. Then, we divided the mesh into 25 overlaping meshes, each of size $12 \times 12 \times 12$, arranged in a 5 by 5 grid. When we applied an external force to one mesh, then at most nine neighbors had to be updated in each iteration step, and only one of the rest was simulated per step, resulting with the simulation of a total of 10 out of 25 meshes. As a result, the total simulation time for the 25 meshes was 800 ms for the FEM and 220 ms for the MSDS simulation.

Even though the total number of nodes and tetrahedra in the stitched meshes was much larger than the number of nodes and tetrahedra in a single large mesh, the simulation time of stitched meshes was 2 to 3 times smaller.

It the test, we selected the 10th mesh at random. A better strategy is to find the maximum stitching force between pairs of overlapping volumes and select the volume with the largest maximum stitching force.

In our second test, we interactively increased the number of meshes, as shown in Fig. 5.3, and measured the FEM and MSDS simulation times with and without volume stitching. Table 5.3 compares the simulation step times with and without selective stitching. Without selective stitching, the simulation step time increased with the number of meshes. On the other hand, the simulation step time with selective stitching increased until the number of stitched meshes was 10, and then remained constant above 10. The FEM simulation took 3-4 times longer than the MSDS simulation.

Table 5.1: Simulation times with and without volume stitching. 25 meshes of size $12 \times 12 \times 12$ are compared with one equivalent $52 \times 52 \times 12$ mesh.

Mesh	25 x (12x12x12)	52x52x12
Total number of nodes	43200	32448
Total number of tetrahedra	166375	143055
Sim. time using FEM [ms]	800	2200
Sim. time using MSDS [ms]	220	500

Table 5.2: Simulation time with (and without) selective volume stitching. Each volume mesh size is $12 \times 12 \times 12$. Simulation using FEM and MSDS.

Number of volumes	1	4	9	16	25
Total number of nodes	1728	6912	15552	27648	43200
Total number of tetrahedra	6655	26620	59895	10648	166375
Sim. time using FEM [ms]	90	335	755	800 (1300)	800 (2080)
Sim. time using MSDS [ms]	25	90	190	220 (340)	220 (530)

Table 5.3: Simulation time with (and without) selective volume stitching. Each volume mesh size is $15 \times 15 \times 15$. Simulation using FEM and MSDS.

Number of volumes	1	4	9	16	25
Total number of nodes	3375	13500	30375	54000	84375
Total number of tetrahedra	13720	54880	123480	219520	343000
Sim. time using FEM [ms]	140	590	1900	2200 (2950)	2200 (4600)
Sim. time using MSDS [ms]	50	180	380	450 (650)	450 (1250)

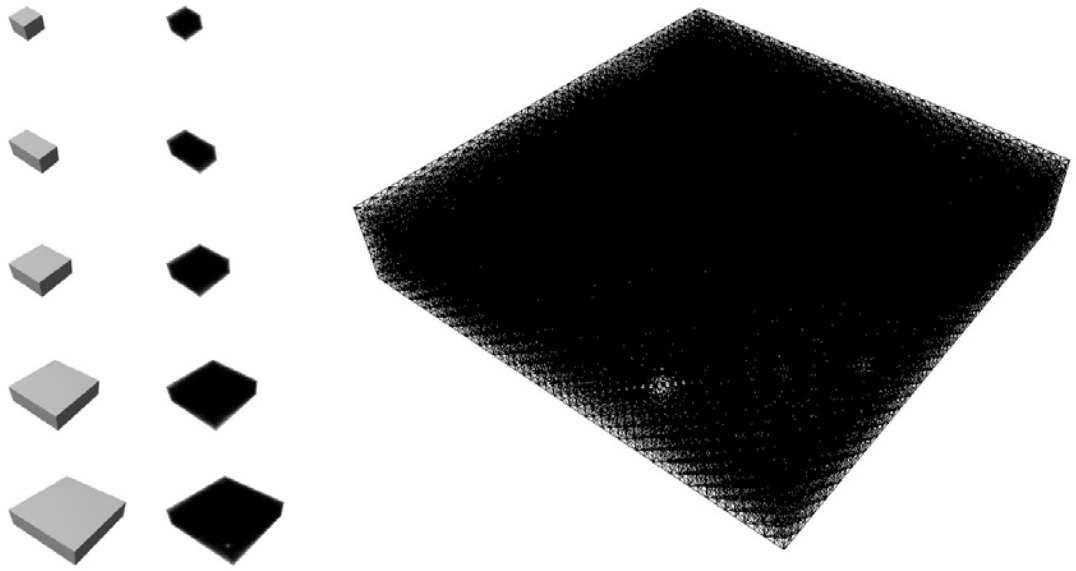


Figure 5.3: Interactive adding and stitching volumes. Adding volume meshes of size $12 \times 12 \times 12$ into a 5×5 grid (left). Overlapped meshes form one $52 \times 52 \times 12$ mesh (right).

CHAPTER 6

Conclusion

6.1 Summary

In this dissertation, we presented a real-time interactive ultrasound simulation with soft-tissue deformation. First, we presented a new approach to ultrasound training—a laptop-based ultrasound simulator with static real-patient, case-specific data registered with a virtual patient. We developed our simulation system, which is currently being sold as a commercial product, and argued that it enables realistic and cost-effective ultrasound training. Next, to augment the realism of our ultrasound training system, we focused on the simulation of soft-tissue deformation. We applied mass-spring-damper systems (MSDS) and the finite element method (FEM) to the simulation of ultrasound compression and evaluated their performance. In our system, the skin of virtual body model deforms at interactive rates when in contact with the virtual ultrasound probe, and the user can visualize in real-time how the vessels deform under compression applied to the skin. To handle large soft-tissue volumes, we presented a technique that simulates multiple sub-volumes in parallel while performing real-time volume stitching. We also presented our data acquisition and processing pipeline, which includes a semi-automatic segmentation method. Finally, we demonstrated that real-time interactive simulation is possible by carefully adapting the algorithms and the code to run efficiently on multicore personal computers.

6.2 Future work

In our work to date, we assumed that the skeleton is static. However, skeletal motion is an important factor in ultrasound examination. For example, in the musculoskeletal (MSK) ultrasound examination of the hand, the hand's motion can be used to detect a defect, a break, a cause of pain, or it can be used to identify structures such as tendons and ligaments. In our future work, we would like to simulate ultrasound imaging in deformable tissues coupled to a skeleton with mobile joints. This would be of immediate benefit to the training of MSK ultrasound of the hand, the knee, and the elbow, among other jointed structures of the body.

APPENDIX A

The Theory of Elasticity

A body subjected to external load may deform. If the relative position of any two points within the body is changed, then the body is said to be strained (Ugural and Fenster, 2003). Let (x, y, z) be the standard Cartesian coordinates. The displacement at every point within the body can be represented by u, v and w in the x, y, and z coordinate directions, respectively, as shown in Fig. A.1. Hence, the displacement constitutes a displacement field, $u = u(x, y, z)$, $v = v(x, y, z)$, and $w = w(x, y, z)$. For linearly elastic materials, the displacement field is a linear function of the loads that produce it.

Normal strains, denoted by ε_x, ε_y, and ε_z, and shearing strains, denoted by γ_{xy}, γ_{yz}, and γ_{zx}, are defined by

$$\begin{aligned} &\varepsilon_x = \frac{\partial u}{\partial x}, \qquad \varepsilon_y = \frac{\partial y}{\partial y}, \qquad \varepsilon_z = \frac{\partial w}{\partial z}, \\ &\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}, \qquad \gamma_{yz} = \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}, \qquad \gamma_{zx} = \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}. \end{aligned} \tag{A.1}$$

By examining (A.1) it may be deduced that $\gamma_{xy} = \gamma_{yx}$, $\gamma_{yz} = \gamma_{zy}$, and $\gamma_{zx} = \gamma_{xz}$, meaning that shearing strains are symmetric.

The infinitesimal strain-displacement relationships in (A.1) can be summarized as

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \qquad i, j = x, y, z, \tag{A.2}$$

where $u_x = u$, $u_y = v$, $u_z = w$, $x_x = x$, $x_y = y$, and $x_z = z$. Equation (A.2) is

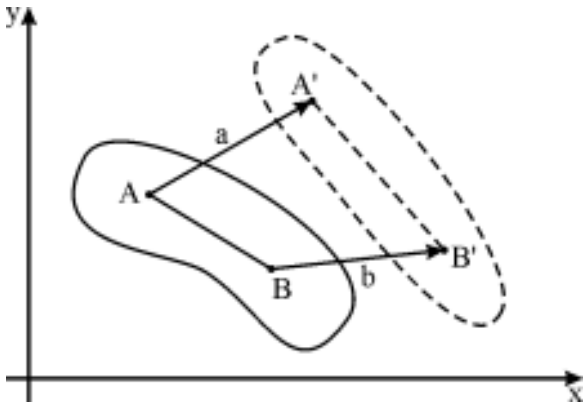


Figure A.1: A two-dimensional case of displacement and strain in a body, where $a = [u(x_A, y_A), v(x_A, y_A)]^T$ and $b = [u(x_B, y_B), v(x_B, y_B)]^T$.

also known as Cauchy strain tensor. In 3D the strain matrix is given as

$$[\varepsilon] = \begin{bmatrix} \varepsilon_{11} & \varepsilon_{12} & \varepsilon_{13} \\ \varepsilon_{21} & \varepsilon_{22} & \varepsilon_{23} \\ \varepsilon_{31} & \varepsilon_{32} & \varepsilon_{33} \end{bmatrix} = \begin{bmatrix} \varepsilon_x & \frac{1}{2}\gamma_{xy} & \frac{1}{2}\gamma_{xz} \\ \frac{1}{2}\gamma_{xy} & \varepsilon_y & \frac{1}{2}\gamma_{yz} \\ \frac{1}{2}\gamma_{xz} & \frac{1}{2}\gamma_{yz} & \varepsilon_z \end{bmatrix}. \tag{A.3}$$

The Cauchy strain tensor is a first-order approximation to the Green strain tensor

$$\varepsilon = \frac{1}{2}\left(\frac{\partial u}{\partial x_i}\frac{\partial u}{\partial x_j} + \frac{\partial u_i}{\partial x_i} + \frac{\partial u_j}{\partial x_i}\right), \qquad i, j = x, y, z \tag{A.4}$$

The Green strain tensor contains a nonlinear quadratic term, which complicates analysis of large displacements. Fortunately, the quadratic term becomes negligible if the displacements are small; consequently, the Cauchy strain tensor can be used.

If we consider a small area element within a still body subjected to an external load distributed through the body, then the force applied in that area is called stress. For relatively small deformations, most materials behave elastically and linearly according to Hooke's Law. To be precise, stress is directly proportional to strain

$$\sigma_x = E\varepsilon_x, \tag{A.5}$$

where σ_x and ε_x denote stress and strain, respectively, both acting in the x direction, and E is called the modulus of elasticity, or Youngs modulus.

Similarly, the elasticity of shear follows Hooke's law in shear:

$$\tau = G\gamma, \tag{A.6}$$

where G is the shear modulus of elasticity, or the modulus of rigidity. For all components, $\tau_{xy} = G\gamma_{xy}$, $\tau_{yz} = G\gamma_{yz}$, and $\tau_{zx} = G\gamma_{zx}$. Both E and G are constant for a given material, and are associated by

$$G = \frac{E}{2\,(1+\upsilon)}. \tag{A.7}$$

It was experimentally found that axial stress, say in the x direction, is proportional to lateral strain, in the y and z direction:

$$\varepsilon_y = \varepsilon_z = -\upsilon\frac{\sigma_x}{E}. \tag{A.8}$$

Here υ is known as Poisson's ratio, which can range from approximately 0 for cork to approximately 0.5 for rubber. Negative Poisson's ratio indicates an auxetic material, which becomes thicker perpendicular to the applied stretching force.

The stress state at each point in the body can be represented by an infinitesimal cube with three stress components on each of its six sides (one direct component and two shear components). Since each point in the body is under static equilibrium, only nine stress components from three planes are needed to describe the stress. These nine components can be organized into matrix form, as follows:

$$[\sigma] = \begin{bmatrix} \sigma_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz} \end{bmatrix}, \tag{A.9}$$

which is called stress tensor (or stress matrix). The stress tensor is a second-order tensor,[1] and it can be shown that $\tau_{yx} = \tau_{xy}$, $\tau_{zx} = \tau_{xz}$ and $\tau_{zy} = \tau_{yz}$; i.e., that this matrix is symmetrical.

[1] A second-order tensor S is a linear mapping that associates a given vector u with a second vector v as $v = Su$.

In other words, the strain and stress are defined uniquely in terms of displacement functions.

APPENDIX B

The Finite Element Method

The term "finite element method" was coined by Clough in a 1960 paper on plane elasticity problems (Clough, 1960); however, the ideas of finite element analysis have roots in applied mathematics, physics, and engineering. For example, Courant used the term "elements" in 1943 for piecewise continuous functions defined over triangular domains to find approximate upper and lower bounds for eigenvalues (Courant, 1943).

The finite element method can be formulated in many different ways. For the solution of practical problems, the displacement-based finite element method is considered to be the most important, mainly because of its simplicity, generality and good numerical properties. Other formulations include the "strong" or classical and "weak" or variational. An example method based on the variational formulation is (Bubnov-) Galerkin's approximation method of obtaining approximate solutions to boundary-value problems, which is a type of so-called weighted residual method (Bathe, 1982).

Displacement u at any point within a finite element e can be approximated as a vector $\hat{u}$, as follows:

$$u \approx \hat{u} = \sum_{k=1}^{n_{en}} N_k a_k^e = N a^e, \tag{B.1}$$

where e is defined by the number of nodes n_{en}, a^e represents element nodal displacements, and components of N are prescribed functions of position called shape functions.

Strain ε at any point within the element is given by

$$\varepsilon \approx \hat{\varepsilon} = Su, \tag{B.2}$$

where S is a second order tensor, which can be approximated as

$$\varepsilon \approx \hat{\varepsilon} = Ba, \tag{B.3}$$

$$B = SN. \tag{B.4}$$

The linear Cauchy strain matrix in (A.2) and (A.3) is symmetric, and in 3D its independent components can be written as (Erleben et al., 2005)

$$\varepsilon = \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{33} \\ \gamma_{12} \\ \gamma_{13} \\ \gamma_{23} \end{bmatrix} = \begin{bmatrix} \frac{\partial u_1}{\partial x_1} \\ \frac{\partial u_2}{\partial x_2} \\ \frac{\partial u_3}{\partial x_3} \\ \frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1} \\ \frac{\partial u_1}{\partial x_3} + \frac{\partial u_3}{\partial x_1} \\ \frac{\partial u_2}{\partial x_3} + \frac{\partial u_3}{\partial x_2} \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1} & 0 & 0 \\ 0 & \frac{\partial}{\partial x_2} & 0 \\ 0 & 0 & \frac{\partial}{\partial x_3} \\ \frac{\partial}{\partial x_2} & \frac{\partial}{\partial x_1} & 0 \\ \frac{\partial}{\partial x_3} & 0 & \frac{\partial}{\partial x_1} \\ 0 & \frac{\partial}{\partial x_3} & \frac{\partial}{\partial x_2} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = Su, \tag{B.5}$$

where $\gamma_{ij} = 2\varepsilon_{12}$, and the components of the strain ε measure stretching and shearing.

Due to the symmetry of the stress matrix in (A.9), just as for the linear Cauchy strain matrix, the nine components can be reduced to six, written in vector form as

$$\sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{yy} & \sigma_{zz} & \tau_{xy} & \tau_{xz} & \tau_{yz} \end{bmatrix}^T, \tag{B.6}$$

which can be related to the strain ε by Hooke's Law generalized to elastic solids as

$$\sigma = D\varepsilon. \tag{B.7}$$

Matrix D is called the isotropic elasticity matrix and it is defined as

$$D = \frac{E}{(1+\upsilon)(1-2\upsilon)} \begin{bmatrix} 1-\upsilon & \upsilon & \upsilon & 0 & 0 & 0 \\ \upsilon & 1-\upsilon & \upsilon & 0 & 0 & 0 \\ \upsilon & \upsilon & 1-\upsilon & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\upsilon}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\upsilon}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\upsilon}{2} \end{bmatrix}, \tag{B.8}$$

where E is Young's modulus and υ is the Poisson' ratio.

The Principle of Virtual Displacement

Assume that the external forces acting on the body are given and that we want to find resulting displacements, strains and stresses. Then the body response can be calculated by establishing the governing differential equations of the equilibrium, subject to boundary and compatibility conditions. An equivalent approach is to use the principle of virtual displacement (Bathe, 1982). This principle states that the equilibrium of the body requires that the total internal virtual work is equal to the total external virtual work for any small virtual displacement imposed onto the body; i.e., external virtual work is equal to internal virtual work when equilibrated forces and stresses undergo unrelated but consistent displacements and strains. Thus, we have

$$\int_{\Omega} W_\sigma dV = W_q + \int_{\Gamma} W_t dA. \tag{B.9}$$

This equation is also known as static equilibrium, or the work balance equation, where Γ represents the surface of a finite element, Ω represents points inside the surface, W_σ is the internal work performed by stress forces, W_q is the external work performed by the nodal forces q^e, and W_t represents the external work performed by the external distributed load t. Assuming a virtual displacement δu^e of the

nodes of finite element e, such that $(u^e)' = u^e + \delta u^e$, then the displacement and strain for every point inside e can be found using linear interpolations (B.1) and (B.2) as

$$\delta u = N\delta u^e, \tag{B.10}$$

$$\delta\varepsilon = B\delta u^e. \tag{B.11}$$

From this the internal and external work equations are given as:

$$W_\sigma = (\delta\varepsilon)^T\sigma, \tag{B.12}$$

$$W_q = (\delta u^e)^T q^e, \tag{B.13}$$

$$W_t = (\delta u)^T t. \tag{B.14}$$

Substituting (B.10) and (B.11) into (B.12), (B.13), and (B.14), and then the result into (B.9) gives:

$$\int_\Omega (B\delta u^e)^T\sigma dV = (\delta u^e)^T q^e + \int_\Gamma (N\delta u^e)^T t dA. \tag{B.15}$$

After rearrangement, we obtain

$$(\delta u^e)^T \int_\Omega B^T\sigma dV = (\delta u^e)^T \left(q^e + \int_\Gamma N^T t dA \right). \tag{B.16}$$

This must hold for any virtual displacement δu^e, so we can conclude that

$$\int_\Omega B^T\sigma dV = q^e + \int_\Gamma N^T t dA. \tag{B.17}$$

To evaluate the terms under integrals, we substitute (B.7) followed by (B.3) to find:

$$\int_\Omega B^T DBu^e dV = q^e + \int_\Gamma N^T t dA. \tag{B.18}$$

The nodal displacement u^e as well as the matrices B and D are constant w.r.t. integration; thus, we obtain

$$B^T DBu^e \int_\Omega dV = q^e + \int_\Gamma N^T t dA. \tag{B.19}$$

This results in

$$B^T DBu^e V^e = q^e + \int_\Gamma N^T t dA, \tag{B.20}$$

where V^e is the volume of finite element. Now, this may be written as a simple matrix equation,

$$K^e u^e = q^e + f^e, \tag{B.21}$$

where

$$K^e = B^T DBV^e, \tag{B.22}$$

and

$$f^e = \int_\Gamma N^T t dA. \tag{B.23}$$

The term K^e is the element stiffness matrix, u^e is the nodal displacement, q^e is the nodal force, and f^e is the element surface force.

The element matrix equation (B.21) describes the characteristics of an element in the system, and, in general, it can be written in the standard form:

$$K^e u^e = F^e, \tag{B.24}$$

where the superscript e denotes the local coordinate space.

Assembly of Elements

The next step in the finite element method is to combine all the element equations into a complete set governing the element composite. The assembly procedure is based on compatibility at the element nodes; i.e., on the balance of all forces that act upon a node in static equilibrium. If we impose this equilibrium condition at a particular node i, we find that the sum of all the nodal forces in one direction equals the resultant external load applied at the node. This process will assemble all the local stiffness matrices into one global stiffness matrix, and the final

equation will have the form

$$Ku = f, \tag{B.25}$$

where u is nodal displacement vector, and f is a vector of external forces. The K matrix is banded, sparse, and usually symmetric and positive definite ($u^T Ku > 0$ for any nonzero vector u). Moreover, the element stiffness matrix and the assembled stiffness matrix, are always singular (their determinants are zero and they have no inverses). Luckily, K becomes invertible after applying boundary conditions to (B.25), which reduces the number of nodal unknowns and the number of equations to be solved.

Huebner et al. (2001) describe four ways of imposing boundary conditions. The easiest way to implement on a computer is to multiply by a large number the diagonal term of K, which is associated with a specified nodal variable, and replace the corresponding term in f by the product times the specified nodal variable, as shown in the following example:

$$\begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} \\ k_{21} & k_{22} \times 10^{15} & k_{23} & k_{24} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} \times 10^{15} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} F_1 \\ \beta_2 k_{22} \times 10^{15} \\ F_3 \\ F_4 \\ \beta_5 k_{55} \times 10^{15} \end{bmatrix}, \tag{B.26}$$

where β_2 and β_5 are constraints on u_2 and u_5, respectively. It is easy to verify that $u_2 = \beta_2$ under assumption that $k_{22} \cdot 10^{15} \gg k_{2j}$, $j = 1, 3, 4, 5$. Similarly, it can be shown that $u_5 = \beta_5$.

Solution of the Stiffness Equations

The nodal displacement u can be found from the system of linear equations in B.25 with imposed boundary conditions. Since the matrix K is usually symmetric and positive definite, a unique solution is guaranteed and can be found by applying

the matrix decomposition method. The matrix K is decomposed into a lower-triangular matrix L with unit diagonal elements and a diagonal matrix D, such that $K = LDL^T$. First we solve the linear system $Ly = F$, where $y = DL^T$, and then we find u from $L^T u = D^{-1} y$.

Another way to solve (B.25) with imposed boundary conditions is to use an iterative solver. A widely used iterative solver is the preconditioned conjugate gradient (PCG) method. The conjugate gradient method computes the solution from an initial guess that is updated through a successive approximation and error reduction approach. Convergence is faster with the addition of an appropriate preconditioner matrix. The preconditioner may require more calculations, but will likely reduce the number of iterations enough to improve the overall cost of computing the solutions. Jacobi, incomplete factorization, multigrid, and domain decomposition are some of the most popular preconditioning methods. Special care must be taken in the selection of the convergence criterion in order to be able to handle a variety of problems with minimal user intervention.

APPENDIX C

The Finite Volume Method

The Finite volume method starts from a mesh, where each node in the mesh is surrounded by a discrete region as shown in Fig. C.1. The internal force per unit area with respect to a plane can be found as

$$\boldsymbol{f} = \sigma \boldsymbol{n}, \tag{C.1}$$

where σ is the stress tensor and $\boldsymbol{n}$ is normal vector to the plane. Thus, the force on the node $\boldsymbol{x}_i$ surrounded by the region Ω can be found as

$$\boldsymbol{f}_i = \int_{\partial\Omega} \sigma \boldsymbol{n} dS. \tag{C.2}$$

The boundary $\partial\Omega$ of the region transects all elements incident to node $\boldsymbol{x}_i$. With linear basis shape functions, the stress tensor σ is constant within elements. Using the divergence theorem, Teran et al. (2003) showed that the integral of $\sigma\boldsymbol{n}$ can be replaced with an integral over the edges (or faces in three spatial dimensions) incident to the node $\boldsymbol{x}_i$. Then, the nodal forces in the tetrahedral mesh can be found simply by looping through all element faces and adding one sixth of $-\sigma$ times the cross product of the two edges.

When the first or second Piola-Kirchoff stresses are considered in the FVM, the number of multiplications and storage requirements are reduced by a factor of five compared to the FEM. Furthermore, there are no storage requirements in the Cauchy stress case. In order to illustrate the FVM technique, Teran et al. simulated a bouncing torus using simple isotropic linear elasticity to calculate the stress (Teran et al., 2003), as shown in Fig. C.2.

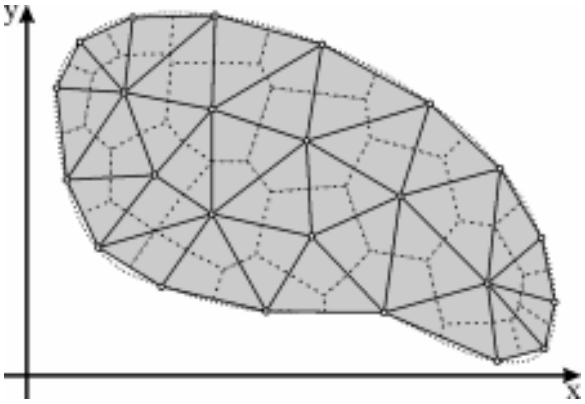


Figure C.1: An example of FVM integration regions.

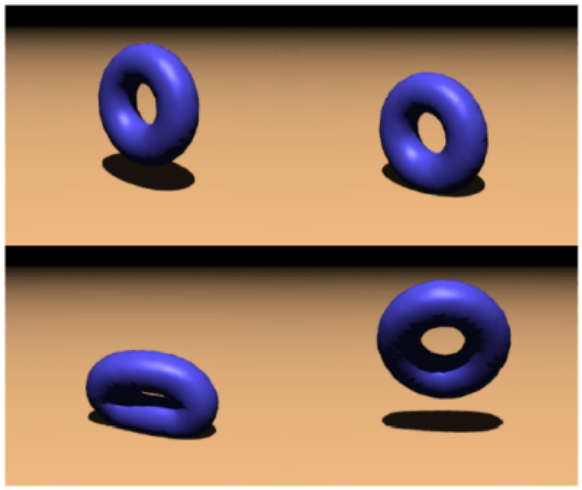


Figure C.2: Deformable torus simulated with FVM, from Teran et al. (2003).

APPENDIX D

Motion Tracking

A very important aspect of ultrasound scanning is hand-eye coordination. A sonographer presses the probe onto the body and must be able to map a 2D ultrasound image to corresponding human anatomy. Moving the probe along the body towards patient's left side with the probe indicator pointing to the left will produce completely different images when the probe indicator is pointing towards the patient's head. In fact, sonographers are trained to follow certain conventions that depend on the procedure they are performing or on the body part they are scanning. For example, in ultrasound examination of the heart parasternal long axis view, they must aim the probe indicator down towards the patient's left elbow in order to find the optimal window.

In order to closely imitate the interaction in a simulation environment, it is necessary to use a motion tracking device. There are many motion tracking devices, which may be categorized into mechanical, optical, inertial, and acoustical.

Visual tracking

Estimation of the position and orientation of objects placed in a scene is a difficult problem. It is often solved, especially in augmented reality applications, using fiducial markers. In augmented or virtual reality, the fiducial markers, which are designed to be easily identifiable in the picture, are often manually applied to objects. For example, a marker can be a red light emitting diode, or a black circle

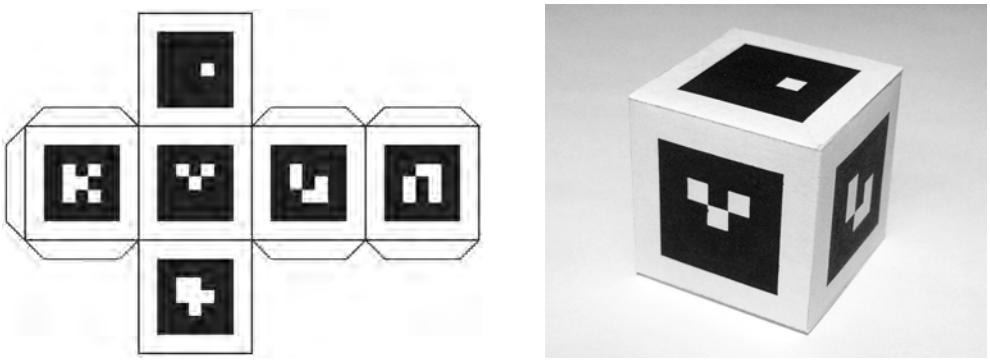
on a white background.

ARToolKit is a widely used computer vision library for maker tracking in augmented reality. It uses square markers with manually defined patterns to track the relative camera position and orientation in real-time. Poorly chosen patterns have a negative impact on tracking performance. Flohr and Fischer (2007) present a method for the automatic generation of a large number of markers, which are ID-based and consist of large monochrome patches. The markers are easily identifiable even in poor lightning conditions, which yields an improved recognition rate and tracking performance.

Compared to standard markers, the use of ID-based markers leads to improved marker recognition as long as the angle between the camera and the marker is approximately between 10 and 80 degrees (zero degrees corresponds to a marker side view, and 90 degrees to a marker top view). When the angle approaches 90 degrees, unstable calculations occur (jittering) due to an algorithm which uses four points to compute camera-marker affine transformation. Slight improvements are made by using image information from the previous frame. This gives stable, but not necessarily accurate results.

This limited range of motion is acceptable for some augmented reality application, but not for the simulation of ultrasound training where the range of motion may be up 180 degrees for yaw, pitch, and roll angels. To overcome this limitation, ID-based markers can be applied to sides of a cube. Only six unique markers are required to cover all sides of the cube. In addition to markers, five affine transformations must be defined do describe the relative position and orientation of the markers. A simple cube with six markers can be made out of paper. Markers must be carefully placed on a flat surface that will fold into a unit cube. Fig. D.1 shows an example of a cube net with six unique markers and the corresponding cube folded out of paper.

Although the detection of markers on the cube yields a full range of motion,



(a) Cube net. (b) Folded cube.

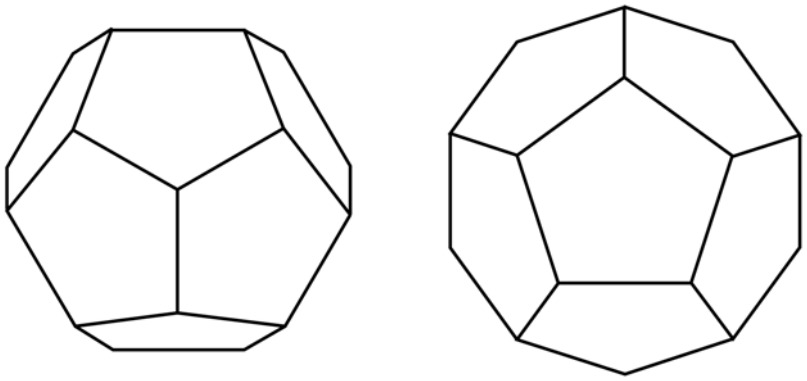
Figure D.1: An example of a cube with six unique markers.

the problem of jittering remains when the camera directly faces one side of the cube. In this situation, other markers are not visible and the problem of tracking multiple markers is reduced to a single marker detection.

One way to overcome this problem is to put markers on a convex polyhedron. In order to ensure that two or more faces are visible at all times, the angle between faces (dihedral angle) must be more than 90 degrees. One polyhedron which satisfies that requirement is the regular dodecahedron. It is composed of 12 regular pentagonal faces, and the dihedral angle is approximately 116.565 degrees. Another good characteristic of the dodecahedron is that at least three faces are always visible with an angle greater than 10 degrees (Fig. D.2(a)). In the situation where the cube with markers fails; i.e., when the camera directly faces one side of the cube, the six sides of the dodecahedron are visible simultaneously (Fig. D.2(b)). This ensures a full range of motion.

Choice of patterns

In low lighting conditions, markers may be incorrectly identified. The selection of markers such that they differ as much as possible may increase the detection rate, simply by pruning outliers. For example, if two markers differ by only one



(a) Good visibility of three faces. (b) Good visibility of six faces.

Figure D.2: Visibility of faces of a dodecahedron. At least three faces have good visibility.

patch, then an incorrect reading of that one patch gives us a valid marker ID and incorrect estimated position and orientation of the marker. On the other hand, if two markers differ by two or more patches, then the incorrect reading of one patch results with an invalid marker ID and that one is pruned from the estimation of position and orientation. Further improvements are possible by checking which marker ID may be visible simultaneously. Markers on opposite sides of the dodecahedron cannot be seen at the same time, so if such markers are detected then one of them must be ignored.

We want as large marker patches as possible and as many combinations as possible. With 2-by-2 marker patches it is possible to generate $2^4 = 16$ combinations, but only 3 are unique because ambiguous (which look the same when 90, 180, or 270 degree rotation is applied) must be excluded. Similarly, with 3-by-3 patches, it is possible to generate $2^9 = 512$ combinations, but only 120 unique ones (Fig. D.3). With 4-by-4 patches, it is possible to generate 65536 combinations, out of which 16320 are unique.

The next challenge is to select markers which are least similar. For a dodecahedron, we must select 12 out of 120 unique 3-by-3 patterns. There is a total

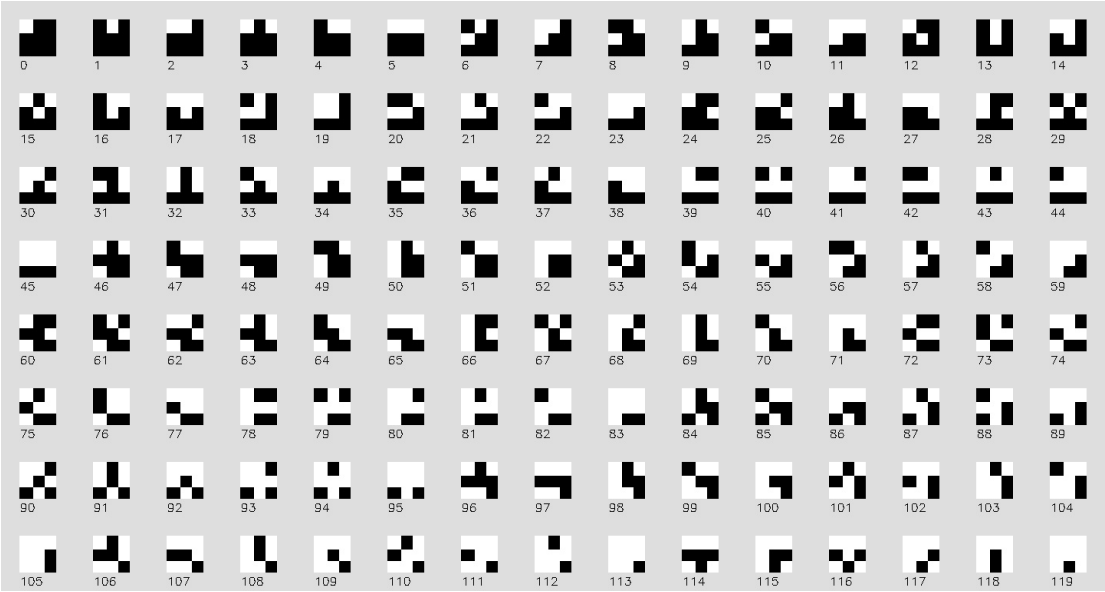


Figure D.3: 120 unique 3-by-3 patterns.

of $C(n, k)$, or "n choose k" k-combinations, where $n = 120$ and $k = 12$, which is approximately 10^{16}. Testing all combinations would take too much time. One could try random sampling, but finding the best combination is very unlikely even after 2 billion trials.

However, if we split unique patterns in two groups, positives (Fig. D.4) and negatives (Fig. D.5), then instead of $C(120, 12)$ we have $C(60, 6) = 50,063,860$ k-combinations. In that case, the best combination can be found in a reasonable amount of time on a standard computer. The observation is that a positive and corresponding negative are least similar so if we find a set of six least similar positives, then they will form a set of twelve least similar patterns when we include their negatives.

In order to find the least similar 3-by-3 patterns, first we create a 120-by-120 similarity matrix, $\boldsymbol{M}$. The value in row r and column c, $\boldsymbol{M}(r, c)$, corresponds to the similarity between patterns r and c shown in Figs. D.4 and D.5. A zero value indicates that patterns are equal, and one or more that patterns differ in one or more patches. We make sure that marker r is compared with not only

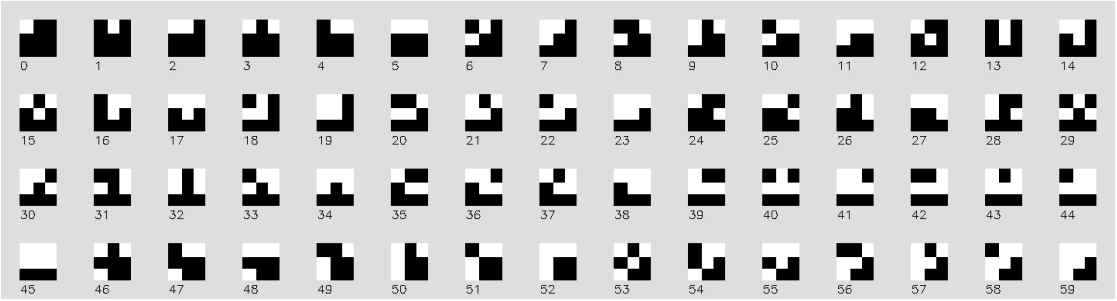


Figure D.4: Positives of 120 unique 3-by-3 patterns.

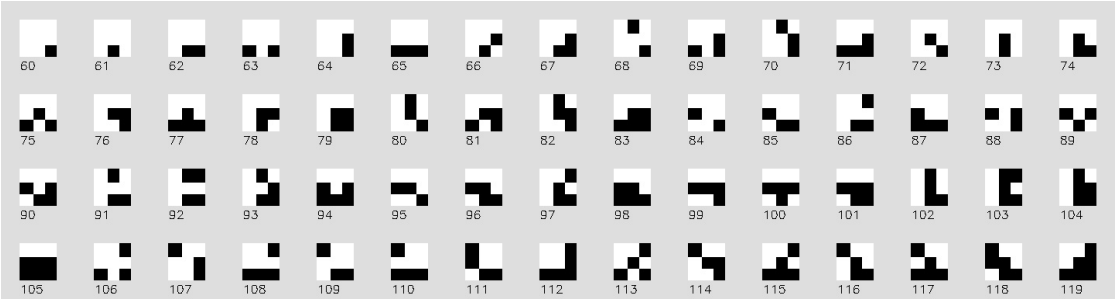
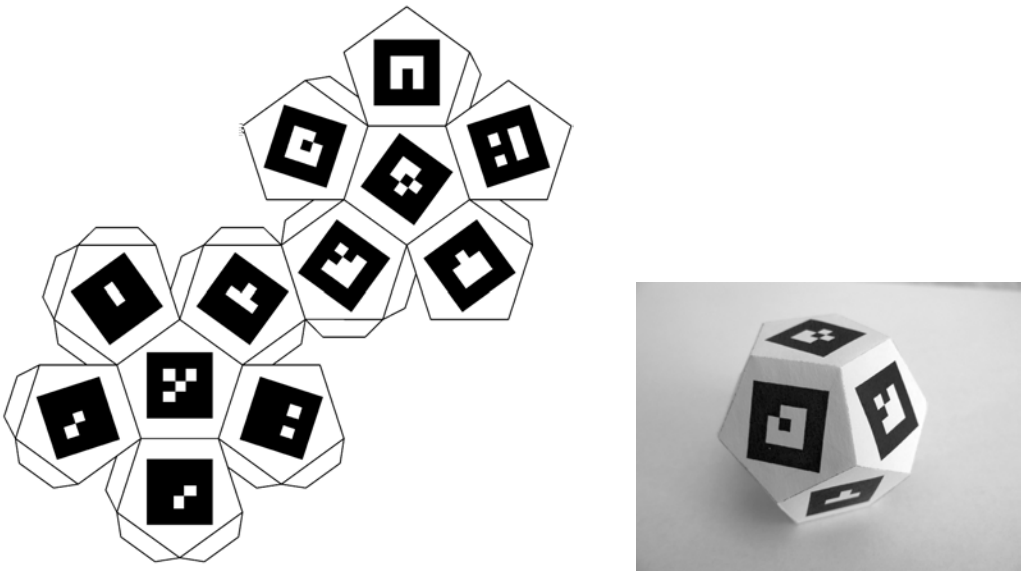


Figure D.5: Negatives of 120 unique 3-by-3 patterns.

marker c but also with all other possible orientations of marker c (90, 180 and 270 degrees). A marker rotated by 90 degrees is still equal to a non-rotated marker so the similarity is zero. As a consequence, the similarity matrix is symmetric with zero diagonal elements, and all other elements positive integer values.

Then, we run all $C(60, 6)$ k-combinations. For each combination, which is a six-dimensional vector of the positives, we add the indexes of corresponding negatives, forming a 12-dimensional vector v. Then we create a 12-by-12 similarity matrix, $\boldsymbol{S}$. Each element $\boldsymbol{S}(i, j)$ in matrix $\boldsymbol{S}$ is set to a precomputed corresponding value $\boldsymbol{S}(i, j) = \boldsymbol{M}(v(i), v(j))$ in matrix $\boldsymbol{M}$. Then we find a minimum non-diagonal value m_S and sum of elements s_S. We keep combinations with largest m_S and largest s_S. One such combination is: 53, 40, 13, 12, 6, 3, 113, 100, 73, 72, 66, 63. The smallest similarity of that combination is 2 (all patterns differ by at least two patches), and the sum is 568.



(a) Dodecahedron net.

(b) Folded dodecahedron.

Figure D.6: An example of a dodecahedron with 12 unique markers.

Finally, we print the best combination markers on the dodecahedron net as shown in Fig. D.6(a). The net is automatically generated, with markers centered on each face of dodecahedron, the net is printed on paper, and the dodecahedron is folded out of the paper (Fig. D.6(b)).

Results

Results show improved marker detection and range of motion—we can track yaw, pitch, and roll angles with a full range of motion.

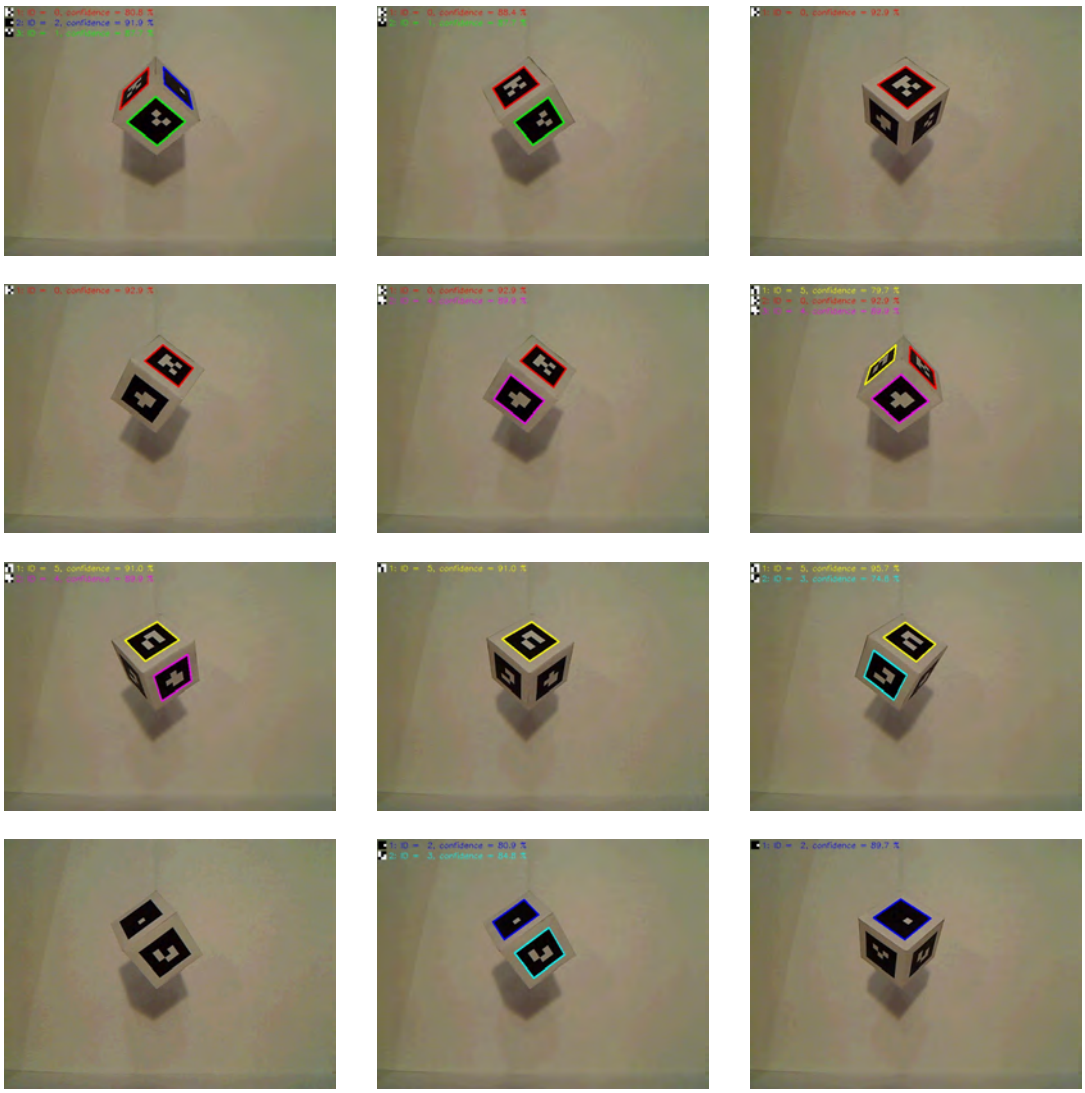


Figure D.7: Cube tracking. The top left corner of each image shows the pattern detected and the detection confidence.

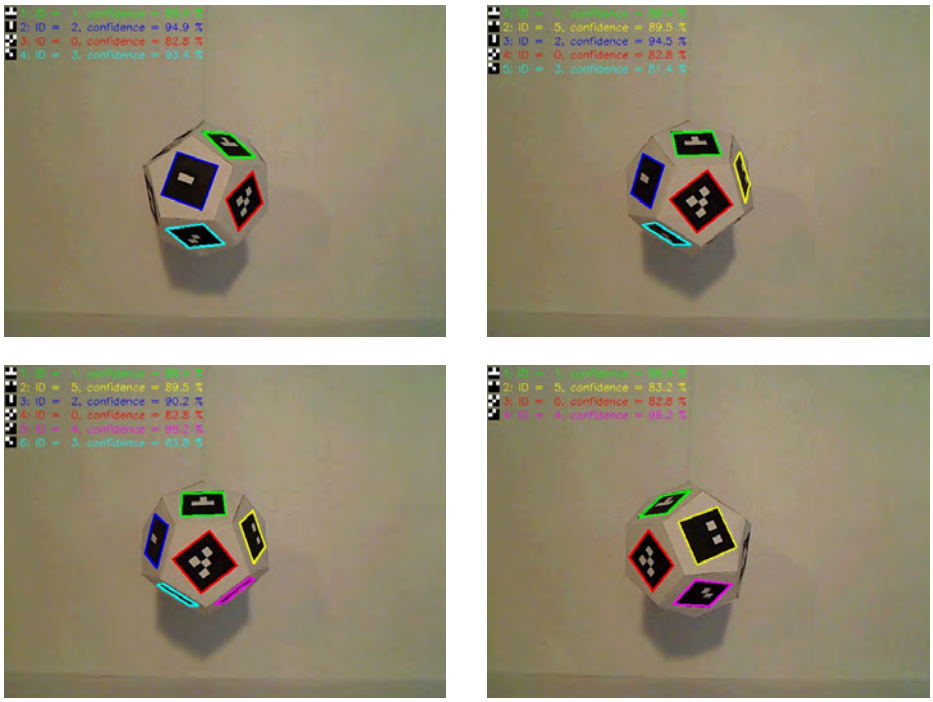


Figure D.8: Dodecahedron tracking. The top left corner of each image shows the pattern detected and the detection confidence.

Bibliography

Archip, N., Rohling, R., Dessenne, V., Erard, P., and Nolte, L. (2006). Anatomical structure modeling from medical images. *Computer Methods and Programs in Biomedicine*, 82(3):203 – 215. 26

Baraff, D. and Witkin, A. (1998). Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 43–54, New York, NY, USA. ACM. 22

Barry, C., Allott, C., John, N., Mellor, P., Arundel, P., Thomson, D., and Waterton, J. (1997). Three-dimensional freehand ultrasound: Image reconstruction and volume analysis. *Ultrasound in Medicine and Biology*, 23(8):1209 – 1224. 14

Bathe, K. (1982). *Finite Element Procedures in Engineering Analysis*. Prentice-Hall. 78, 80

Boeing, A. and Bräunl, T. (2007). Evaluation of real-time physics simulation systems. In *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, GRAPHITE '07, pages 281–288, New York, NY, USA. ACM. 48

Chernyaev, E. V. (1995). Marching cubes 33: Construction of topologically correct isosurfaces. Technical report, CERN. 17

Ciarlet, P. and Lions, J., editors (1996). *Automatic mesh generation and finite element computation*, volume IV. Elsevier Scinece B.V. 26

Clough, R. (1960). The finite element method in plane stress analysis. In *Proceedings, 2nd Conference on Electronic Computation, A.S.C.E. Structural Division*, Pittsburgh, Pennsylvania. 22, 78

Courant, R. (1943). Variational methods for the solution of problems of equilibrium and vibrations. *Bull. Amer. Math. Soc.*, 49(1):1 – 23. 78

Crisp, J. G., Lovato, L. M., and Jang, T. B. (2010). Compression ultrasonography of the lower extremity with portable vascular ultrasonography can accurately detect deep venous thrombosis in the emergency department. *Ann Emerg Med*, 56(6):601–610. 3

Dillenseger, J., Laguitton, S., and Delabrousse, E. (2009). Fast simulation of ultrasound images from a CT volume. *Computers in Biology and Medicine*, 39(2):180 – 186. 12, 13, 18, 19

Ehricke, H. (1998). Sonosim3D: a multimedia system for sonography simulation and education with an extensible case database. *European Journal of Ultrasound*, 7(3):225 – 230. 18

El-Zafrany, A. (1993). *Techniques of the Boundary Element Method*. Ellis Horwood. 25

Erleben, K., Sporring, J., Henriksen, K., and Dohlman, H. (2005). *Physics-based animation*. Charles River Media. 79

Flohr, D. and Fischer, J. (2007). A lightweight id-based extension for marker tracking systems. In *Eurographics Symposium on Virtual Environments (EGVE) Short Paper Proceedings*, pages 59–64. 88

Gee, A., Prager, R., Treece, G., and Berman, L. (2003). Engineering a freehand 3D ultrasound system. *Pattern Recognition Letters*, 24(4-5):757 – 777. 14

George, P., Hecht, F., and Vallet, M. (1991). Creation of internal points in voronoi's type method. control adaptation. *Advances in Engineering Software and Workstations*, 13(5-6):303 – 312. 26

Goksel, O. and Salcudean, E. (2010). Image-based variational meshing. *Medical Imaging, IEEE Transactions on*, PP(99):1 –1. 26, 27

Goksel, O. and Salcudean, S. (2009). B-mode ultrasound image simulation in deformable 3-D medium. *Medical Imaging, IEEE Transactions on*, 28(11):1657 –1669. 23, 24, 55, 57

Heer, I. M., Middendorf, K., Mller-Egloff, S., Dugas, M., and Strauss, A. (2004). Ultrasound training: the virtual patient. *Ultrasound in Obstetrics and Gynecology*, 24(4):440 – 444. 18

Hostettler, A., Forest, C., Forgione, A., Soler, L., and Marescaux, J. (2005). Real-time ultrasonography simulator based on 3D CT-scan images. In *Medicine Meets Virtual Reality 13: The Magical Next Becomes the Medical Now*, volume 111, pages 191 – 193. IOS Press. 18

Huang, Q., Lu, M., Zheng, Y., and Chi, Z. (2009). Speckle suppression and contrast enhancement in reconstruction of freehand 3d ultrasound images using an adaptive distance-weighted method. *Applied Acoustics*, 70(1):21 – 30. 14

Huang, Q. H., Zheng, Y. P., Lu, M. H., and Chi, Z. R. (2005). Development of a portable 3D ultrasound imaging system for musculoskeletal tissues. *Ultrasonics*, 43(3):153 – 163. 14

Huebner, K., Dewhirst, D., Smith, D., and Byrom, T. (2001). *The Finite Element Method for Engineers (4th Edition)*. Wiley-Interscience. 83

Jensen, J. (1996). Field: A program for simulating ultrasound systems. In *10th Nordic-Baltic Conference on Biomedical Imaging*, volume 4, pages 351 – 353. 18

Jensen, J. and Nikolov, I. (2000). Fast simulation of ultrasound images. In *Ultrasonics Symposium, 2000 IEEE*, volume 2, pages 1721 –1724 vol.2. 12

Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: Active contour models. *INTERNATIONAL JOURNAL OF COMPUTER VISION*, 1(4):321–331. 16

Levienaise-Obadia, B. and Gee, A. (1999). Adaptive segmentation of ultrasound images. *Image and Vision Computing*, 17(8):583 – 588. 16

Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3d surface construction algorithm. *COMPUTER GRAPHICS*, 21(4):163–169. 17

Maul, H., Scharf, A., Baier, P., Wstemann, M., Gnter, H. H., Gebauer, G., and Sohn, C. (2004). Ultrasound simulators: experience with the sonotrainer and comparative review of other training systems. *Ultrasound in Obstetrics and Gynecology*, 24(5):581 – 585. 18

McGee, D. and Gould, M. (2003). Preventing complications of central venous catheterization. *The new england journal of medicine*, 384(12):1123 – 1133. 32

Mercier, L., Lang, T., Lindseth, F., and Collins, D. (2005). A review of calibration techniques for freehand 3-d ultrasound systems. *Ultrasound in Medicine and Biology*, 31(4):449 – 471. 14, 31

Miller, G. (1988). The motion dynamics of snakes and worms. *SIGGRAPH Comput. Graph.*, 22:169–173. 22

Molino, N., Bridson, R., Teran, J., and Fedkiw, R. (2003). A crystalline, red green strategy for meshing highly deformable objects with tetrahedra. In *In 12th Int. Meshing Roundtable*, pages 103–114. 26

More, J. J. (1978). The levenberg-marquardt algorithm: Implementation and theory. In Watson, G., editor, *Numerical Analysis*, volume 630 of *Lecture Notes in Mathematics*, pages 105–116. Springer Berlin Heidelberg. 36

Mortensen, E. N. and Barrett, W. A. (1995). Intelligent scissors for image composition. In *Proceedings of the 22nd annual conference on Computer graphics and*

interactive techniques, SIGGRAPH '95, pages 191–198, New York, NY, USA. ACM. 16

Mortensen, E. N. and Barrett, W. A. (1998). Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing*, 60(5):349 – 384. 16

Nealen, A., Mller, M., Keiser, R., Boxerman, E., and Carlson, M. (2006). Physically based deformable models in computer graphics. *Computer Graphics Forum*, 25(4):809–836. 22, 25

Noble, J. and Boukerroui, D. (2006). Ultrasound image segmentation: a survey. *Medical Imaging, IEEE Transactions on*, 25(8):987 –1010. 16

Peraire, J., Vahdati, M., Morgan, K., and Zienkiewicz, O. (1987). Adaptive remeshing for compressible flow computations. *Journal of Computational Physics*, 72(2):449 – 466. 26

Petrinec, K., Hein, C., and Savitsky, E. (2011). Patient-specific cases for an ultrasound training simulator. In *Studies in Health Technology and Informatics*, volume 143. 4, 19, 33

Prager, R., Rohling, R., Gee, A., and Berman, L. (1998). Rapid calibration for 3-d freehand ultrasound. *Ultrasound in Medicine and Biology*, 24(6):855 – 869. 36

Prager, R. W., Gee, A. H., Treece, G. M., Cash, C. J. C., and Berman, L. H. (2003). Sensorless freehand 3-d ultrasound using regression of the echo intensity. *Ultrasound in Medicine and Biology*, 29(3):437 – 446. 14

Reichl, T., Passenger, J., Acosta, O., and Salvado, O. (2009). Ultrasound goes gpu: real-time simulation using cuda. In *Proceedings of SPIE*, volume 7261, pages 726116–726116–10. 12, 18

Rohling, R. N., Gee, A. H., and Berman, L. (1998). Automatic registration of 3-d ultrasound images. *Ultrasound in Medicine and Biology*, 24(6):841 – 854. 14

Rothschild, J. (2001). Ultrasound guidance of central vein catheterization. *Making Health Care Safer: A Critical Analysis of Patient Safety Practices, Evidence Report on Technology Assessment*, 1(43):245 – 253. 32

Sanches, J. M. and Marques, J. S. (2002). A multiscale algorithm for three-dimensional free-hand ultrasound. *Ultrasound in Medicine and Biology*, 28(8):1029 – 1040. 14

Schlei, B. (2012). Volume-enclosing surface extraction. *Computers & Graphics*, 36(2):111 – 130. 18

Shipley, J., Duck, F., Goddard, D., Hillman, M., Halliwell, M., Jones, M., and Thomas, B. (2005). Automated quantitative volumetric breast ultrasound data-acquisition system. *Ultrasound in Medicine and Biology*, 31(7):905 – 917. 14

Solberg, O., Lindseth, F., Torp, H., Blake, R., and Nagelhus Hernes, T. (2007). Freehand 3D ultrasound reconstruction algorithms–a review. *Ultrasound in Medicine and Biology*, 33(7):991 – 1009. 13, 36

Strang, G. (1986). *Introduction to Applied Mathematics*. Wellesley-Cambridge Press. 25

Taubin, G. (1995). Curve and surface smoothing without shrinkage. In *Proceedings of the Fifth International Conference on Computer Vision*, ICCV '95, pages 852–, Washington, DC, USA. IEEE Computer Society. 18

Teran, J., Blemker, S., Hing, V. N. T., and Fedkiw, R. (2003). Finite volume methods for the simulation of skeletal muscle. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, pages

68–74, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association. 25, 26, 85, 86

Terkamp, C., Kirchner, G., Wedemeyer, J., Dettmer, A., Kielstein, J., Reindell, H., Bleck, J., Manns, M., and Gebel, M. (2003). Simulation of abdomen sonography. evaluation of a new ultrasound simulator. *Ultraschall in der Medizin (Stuttgart, Germany : 1980)*, 24(4):239 – 244. 18

Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. (1987). Elastically deformable models. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, pages 205–214, New York, NY, USA. ACM. 25

Terzopoulos, D., Platt, J., and Fleischer, K. (1991). Heating and melting deformable models. *The Journal of Visualization and Computer Animation*, 2(2):68–73. 22

Terzopoulos, D. and Waters, K. (1990). Physically-based facial modeling, analysis, and animation. *The Journal of Visualization and Computer Animation*, 1(2):73–80. 21, 22, 58

Treece, G. M., Gee, A. H., Prager, R. W., Cash, C. J. C., and Berman, L. H. (2003). High-definition freehand 3-d ultrasound. *Ultrasound in Medicine and Biology*, 29(4):529 – 546. 35

Ugural, A. and Fenster, S. (2003). *Advanced Strength and Applied Elasticity (4th Edition)*. Prentice Hall. 74

Varandas, J., Baptista, P., Santos, J., Martins, R., and Dias, J. (2004). Volus–a visualization system for 3D ultrasound data. *Ultrasonics*, 42(1-9):689 – 694. Proceedings of Ultrasonics International 2003. 18

Yang, W. and Feng, J. (2009). 2d shape morphing via automatic feature matching and hierarchical interpolation. *Computers and Graphics*, 33(3):414 – 423. 41

Yerry, M. and Shephard, M. (1985). Automatic mesh generation for three-dimensional solids. *Computers and Structures*, 20(1-3):31 – 39. 26

Zienkiewicz, O. and Taylor, R. (2000). *The Finite Element Method: Volume 1, Fifth Edition*. Butterworth-Heinemann. 22