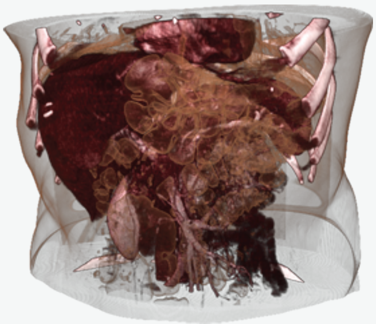
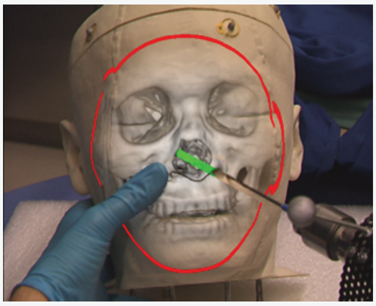
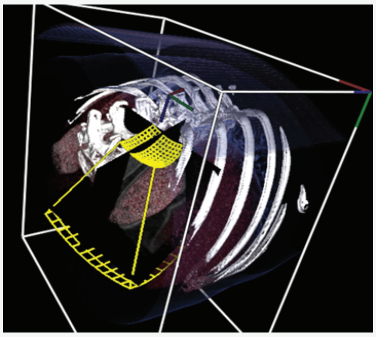
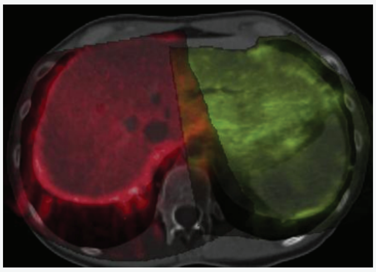
Computer Aided Medical Procedures
(CAMP)
Prof. Dr. Nassir Navab









Dissertation

Visual Computing for Computer Assisted Interventions

Oliver Kutter

Fakultät für Informatik
Technische Universität München

TECHNISCHE UNIVERSITÄT MÜNCHEN

Fakultät für Informatik
Chair for Computer-Aided Medical Procedures & Augmented Reality

Visual Computing for Computer Assisted Interventions

Oliver Kutter

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende: Univ.-Prof. G. J. Klinker, Ph.D.
Prüfer der Dissertation:
1. Univ.-Prof. N. Navab, Ph.D.
2. Prof. T. Peters, Ph.D.
The University of Western Ontario, Canada

Die Dissertation wurde am 04.03.2010 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 18.06.2010 angenommen.

Abstract

Continuous improvement in medical imaging technology provides an ever increasing amount of high resolution, peri-operative and multi-modal image data. Fusion and visualization of multiple datasets of the same patient has been shown to improve diagnosis as well as therapy guidance for many medical interventions.

However, the sheer amount of image data acquired by today's imaging modalities, and their objective fusion and real-time visualization for diagnosis and therapy postulates the request for efficient image processing, visualization and data presentation techniques.

This thesis focuses on developing visual computing solutions for improving and accelerating computer assisted medical interventions. Advanced simulation, registration and visualization algorithms have been developed and implemented on Graphic Processing Units (GPU) for optimal efficiency, merging computation of the result and real-time visualization of the data. The methods are investigated in detail for three, closely related, medical applications.

In the first application GPU-accelerated medical Augmented Reality (AR) visualization on a stereo video see-through Head Mounted Display is investigated. A key challenge for any medical AR system supporting medical navigation tasks is direct, pre-processing free, integration of image data and natural embedding into the AR scene. In this thesis I present a series of techniques for optimizing perception, performance and quality of medical AR visualization. Furthermore, occlusion problems of virtual and real objects are addressed. The methods are evaluated in a series of phantom and in-vivo experiments in close collaboration with our clinical partners.

The second application is simulation of medical ultrasound (US) image data from Computed Tomography (CT) data for patient-based ultrasound acquisition training and patient-specific registration with CT. Here, I present a framework for ray-based ultrasound simulation on GPUs, supporting different simulation models of varying complexity.

The third application is multi-modal registration of US with CT. GPU-accelerated US simulation is an essential part of the registration algorithm. In this thesis we present two novel, efficient, multi-modal registration applications. (1) Simultaneous registration of multiple 3D ultrasound scans with one CT scan. (2) Purely intensity based, dense deformable registration of 3D US and CT scans using a variational approach. The methods are validated for a series of real patient US and CT scans.

Keywords:
GPU, Medical Image Processing, Real-time Visualization, Ultrasound Simulation from CT, Multi-modal Registration, Medical Augmented Reality, Visual Computing

Zusammenfassung

Der kontinuierliche Fortschritt medizinischer bildgebender Modalitäten resultiert in immer grösseren Mengen von hochaufgelösten, peri-operativen und multi-modalen Bilddaten. Der Mehrwert von Fusion und Visualisierung von mehreren Datensätzen des gleichen Patienten für Diagnose und therapeutische Navigation wurde bereits für viele medizinische Anwendungen demonstriert. Die enormen Datenmengen, ihre Fusion und Visualisierung fordern effiziente Algorithmen für Bildverarbeitung, Visualisierung sowie fortgeschrittene Präsentationstechnicken um eine verbesserte Diagnose und Therapie für den Patienten zu erreichen.

Der Kern dieser Arbeit liegt in der Entwicklung von Visual Computing Lösungen für Computer unterstützte Eingriffen in der Medizin. Fortgeschrittene Algorithmen für Visualisierung, Simulation und Registrierung wurden entwickelt und für höchste Effiziens auf Graphik Prozessoren (GPU) implementiert. Die Berechnung des Resultats und Echtzeit Visualisierung erfolgt dabei in einem Arbeitsschritt. Die Methoden werden für drei, eng zusammenliegende, medizinische Anwendungen untersucht.

Die erste Anwendung untersucht GPU-beschleunigte medizinische Augmented Reality (AR) Visualisierung auf einem stereo Video see-through Head Mounted Display. Eine der Hauptherausforderungen für medizinische AR Systeme ist die direkte Integration von medizinischen Bilddaten und eine möglichst natürliche Einbettung in die AR Szene in Echtzeit. Diese Arbeit stellt eine Reihe von Techniken zur Optimierung der Wahrnehmung, Systemleistung und Qualität der AR Visualisierung vor, und zur Lösung von Verdeckungsproblemen von echten und virtuellen Objekten. Die Methoden werden in mehreren Phantom und in-vivo Experimenten in enger Zusammenarbeit mit unseren klinischen Partnern evaluiert.

Die zweite Anwendung ist Simulation von medizinischen Ultraschall (US) aus Computer Tomographie (CT) Daten für patienten-basiertes US Untersuchungstraining und patienten-spezifische Registrierung mit CT Daten. Ein generisches Framework für GPU-beschleunigte Simulation von US aus CT Daten, mittels strahlen-basierter Modelle unterschiedlicher Komplexität, wird vorgestellt.

In der dritten Anwendung wird multi-modale Registrierung von US und CT Daten untersucht. GPU-beschleunigte US Simulation spielt dabei eine essentielle Rolle. In dieser Arbeit werden zwei neue multi-modal Registrierungsanwendungen vorgestellt: (1) Simultane Registrierung von mehreren 3D US Scans mit einem 3D CT Scan. (2) Intensitätsbasierte, deformierbare Registrierung von 3D US und CT Daten mittels eines variationellen Ansatzes. Die Methoden werden anhand einer Reihe von Patienten US und CT Scans validiert.

Schlagwörter:
GPU, Medizinische Bildverarbeitung, Echzeit Visualisierung, Ultraschall Simulation aus CT, Multi-modale Registrierung,Medizinische Augmented Reality, Visual Computing

Acknowledgments

First of all, I would like to thank my PhD advisor Nassir Navab, for accepting and supporting me as one of his PhD students. I had the fortune to already get to know him and join his group as a diploma student and continue to work with him as PhD student. In these last five years he not only provided motivation, ideas, guidance, support and his opinion for my scientific work and this thesis, but also taught me so much more.

I would like to express my special thanks to Jörg Traub, for not only his help on proof-reading this thesis, but also for his great support and unconditional friendship over the last five years. I highly appreciate him sharing his *senior PhD* experience with me, and his feedback and insight he provided to any question and issue we discussed. The same way, I owe many thanks to Wolfgang Wein and Ramtin Shams. I was fortunate to have closely worked with them on multiple research projects in the last years. I am especially thankful for them providing me with many good advices, ideas, productive discussions and joint research work resulting in many publications.

I would like to thank Martin Horn, Virginie Fite Georgel and Martina Hilla for their support on organizational and administrative issues. Especially Martin for always providing me and our GPU group with the latest GPUs for conducting our research. Many thanks to André Aichert, Matthias Wieczorek, Darko Zikic, Jörg Traub and Wolfgang Wein for proof-reading parts of this thesis.

This work would not have been possible without the support of a large team of colleagues and students from the chair for computer aided medical procedures. I would like to thank all of them, especially Jörg Traub, Tobias Sielhorst, Marco Feuerstein, Wolfgang Wein, Stefanie Demirci, Stefan Hinterstoisser, Thomas Wendler, Tobias Lasser, Tobias Blum, Darko Zikic, Ben Glocker, Tobias Reichl, Christian Wachinger, Pierre Fite Georgel, Christoph Bichlmeier, Martin Groher, Ahmad Ahmadi, Hauke Heibel, Andreas Keil, Tassilo Klein, André Aichert, Matthias Wieczorek, Markus Vill, Athanasios Karamalis, Philipp Stefan, Aliaksei Maistrov, Marina Berkovic, Ya Chen, Andreas Kirsch, Benedikt Schultis, Christian Harrer, for the great joint work, joint publications, fruitful discussions and ideas on research, over the last three years.

I would also like to express my thanks to my medical collaboration partners, Sonja

Kirchhof at Radiology Department Grosshadern, Munich. Gernot Brockman, Eva Braun, Robert Bauernschmitt and Rüdiger Lange at German Heart Center, Munich and Sandro Heining, Ben Ockert, Jürgen Landes and Eckehard Euler at Chirugische Klinik und Poliklinik Innenstadt, Munich. My special thanks go to Gernot Brockmann for providing me his expert knowledge on cardiac surgery, cardiac catheter interventions, and 3D echocardiography, as well as being a constant source of ideas and having become a close friend. I would like to thank Jürgen Landes, Sandro Heining and Ben Ockert for setting up and conducting the experiments with the head mounted display based Augmented Reality (AR) system.

I want to thank Wolfgang Wein, Razvan Ionasec, Ali Kamen, Patric Ljung, Guillaume Stordeur, Frank Sauer and Gianluca Paladini from Siemens Corporate Research, Princeton for joint work and publications and providing valuable input for this work. I am very indebted to Gianluca Paladini and Frank Sauer, for supporting my work in Munich by SCR in the last year of this thesis and encouraging me to join the Imaging Architectures group at SCR to continue my work on GPU-accelerated medical image processing and visualization.

Many thanks to all institutions and organizations providing their support for conducting this thesis, German Heart Center Munich, Passport project partly funded by the European Commission, and Siemens Corporate Research.

Finally I owe uncountable thanks to my wife Vendula, for constantly supporting me in these last three, sometimes stressful, years. She always knew how to balance our work and private life, and motivated me to exceed myself many times during the three years and preparation of this thesis. I also want to thank my parents and my parents-in-law for their patience and great support during my thesis.

Contents

Outline of the Thesis

This thesis focuses on developing novel visual computing methods for computer assisted interventions. Following a short motivation and introduction, chapters 2 to 4, introduce the reader to underlying concepts, technology and methods used pervasively in this thesis. Chapter 5 to 7, each present the application of the concepts to a specific medical computing problem.

Each of the following paragraphs provides the reader with a brief abstract description of the individual chapters of the thesis and the underneath thematic interconnections.

Chapter 1: Introduction

Chapter 1 serves as the introduction into this thesis. We briefly summarize the reasons that motivated the work presented in this thesis: the concurrent developments in medical imaging and computing, trend to interventional imaging for many medical procedures and recent progress in high performance computing on Graphics Processing Units (GPUs).

Chapter 2: Medical Imaging

In chapter 2 we provide a comprehensive overview on medical imaging modalities and the areas of medical imaging and computing, interventional imaging and navigation. This chapter introduces the basic concepts of imaging modalities, algorithms, tracking, data acquisition and visualization devices used throughout this thesis.

Chapter 3: High Performance Computing on GPUs

Chapter 3 introduces the reader to the field of High Performance Computing (HPC) on GPUs. We review the recent evolution of GPUs from fixed function processors to fully programmable, many-core, high-performance co-processors. Furthermore, we provide a brief overview and introduction to the GPU programming models and available Application Programmer Interfaces (APIs) for General Purpose GPU (GPGPU) programming. The chapter is concluded by a short summary and an outlook on recent development and trends for HPC on GPUs. The concepts and paradigms presented in this chapter are the foundation for the methods and applications presented in the following chapters of this thesis.

Chapter 4: GPU-Accelerated Volume Visualization

In chapter 4 we present an overview on the underlying concepts for Direct Volume Rendering (DVR). We discuss in detail GPU-accelerated ray-casting, and extensions of the original algorithm optimizing rendering quality and performance. The chapter is concluded by a presentation of advanced rendering techniques for rendering sequences of volumetric data, intuitive volume exploration using focus and context rendering techniques, and fused rendering of multi-modal datasets of the same patient. The rendering techniques presented in this chapter are used throughout many applications presented in this thesis and many more beyond the scope of it.

Chapter 5: Advanced Medical Augmented Reality Visualization

In this chapter we present a rendering pipeline for advanced medical AR visualization. The rendering pipeline heavily uses the methods presented in the previous chapter to implement real-time, high-quality, pre-processing free in-situ AR visualization of volumetric data. Besides real-time DVR for AR, we address the problem of embedding the virtual objects naturally into the scene and handle occlusion problems of virtual and real objects using simple yet efficient and effective methods. The presented methods and techniques are evaluated in a series of phantom and in-vivo experiments. In the evaluation we emphasize on guaranteeing real-time performance for medical procedures as well as improving navigation and guidance by advanced rendering techniques.

Chapter 6: Ultrasound Simulation from CT

In chapter 6 we present a framework for GPU-accelerated Ultrasound (US) from Computed Tomography (CT) data, supporting different ray-based models of US wave propagation in human tissue. US simulation from actual patient CT data is of interest for two applications in medicine: (1) Patient-based US simulation and visualization for US acquisition training and education of novice users. (2) Patient-specific simulation of US imaging effects for multi-modal fusion of pre- or post-operative CT data and peri-operative US data of the same patient. In this chapter we discuss in detail the aspects of ray-based modeling and efficient GPU implementation for simulating 2D, 2D over time, 3D and 4D US data from CT.

Chapter 7: GPU-accelerated Registration

In this chapter we present two novel methods for multi-modal fusion US and CT data. (1) Simultaneous fusion of multiple 3D US scans with one CT scan. (2) Efficient, automatic, intensity-based, dense deformable fusion of 3D US and CT volumes using a variational approach. Both methods build on the methods for GPU-accelerated simulation of US data from CT to facilitate an automatic, robust, image based fusion using variations of the Linear Correlation of Linear Combination (LC^2) similarity metric. The presented methods are evaluated for a series of real patients' US and CT data set pairs.

Chapter 8: Conclusion

Chapter 8 concludes the thesis with a summary of the outcome and discussion of benefits, current limitations and possible future research directions.

Appendix

A: Further Applications

Each section of this appendix chapter presents an application that is not treated in detail in this thesis, but was covered during the time of this thesis and is closely related in terms of methodology or medical application.

B: Further Ultrasound Simulation Results

A selection of additional simulated US images for US simulation in Virtual Reality (VR) and AR environments, and detailed performance charts.

Authored and Co-Authored Publication

All publications contributed to the scientific community during this work.

Abstracts of Major Publications not Discussed in the Dissertation

All abstracts of major publications not discussed in this thesis.

Glossary and Acronyms

The glossary and listing of acronyms used throughout this thesis.

CHAPTER 1

Introduction

Methods, techniques and technology in medicine are constantly evolving to provide the best possible health care to the patient. Advancement in medicine and medical technology is an interactive process driven by both technological and medical research. Technological improvements and innovations enable the development of novel operation and therapy techniques, that were simply not possible or imagined a few years ago. Vice versa, the never ceasing request for novel and improved technical solutions supporting diagnosis and therapy in medicine constantly stimulates engineers and researchers to innovate new solutions.

1.1 Motivation of this Work

The foundation for the motivation to this work has been the recent developments in medical imaging technology providing more and more data, the need for real-time interventional imaging, navigation and visualization in minimally-invasive procedures in medicine and the technological developments in the field of high-performance computing on Graphic Processing Units (GPU). In the following paragraphs a short overview of the specific challenges and motivation for each field is given.

1.1.1 Developments in Medical and Interventional Imaging

Medical Imaging Technology Development

Various medical imaging modalities have been developed and introduced into clinical practice over the last decades. Two-dimensional (2D), three-dimensional (3D) and four-dimensional, or three-dimensional over time (4D) imaging data generated by anatomical imaging modalities, e.g. X-ray Imaging (X-ray), Computed Tomography (CT), Magnetic Resonance Imaging (MRI) and Ultrasound (US) and from functional modalities, e.g. Positron Emission Tomography (PET), Single Photon Emission Computed Tomography (SPECT) are frequently used throughout diagnosis and treatment stages for many pathologies. For a detailed overview on the different medical imaging modalities we refer to section 2.1. Existing modalities are constantly evolving with every generation. New scanners with improved acquisition hardware and protocols enable, e.g. to acquire higher spatial resolution data in less time (Multi-slice CT), record data over time (4D Ultrasound, 4D CT), and measure multivariate data (Dual Intensity CT, B-Mode + Doppler US). Concurrently new imaging modalities are developed, e.g. optical imaging. Furthermore, there is a general trend to adapt existing modalities for interventional imaging to directly support medical procedures in the intervention or operating room.

Interventional Imaging Trend

Interventional Imaging is an enabling key technology for existing and future applications towards minimally-invasive interventions in medicine. Over the last years minimally-invasive procedures have gained more and more interest and importance in the medical community, and have already or are candidates to become standard therapy for many different pathologies.

For instance in cardiology, coronary heart disease is routinely treated by X-ray image guided stenting of the coronary arteries [181], replacing traditional coronary by-pass therapy and the need for open surgery. In cardio-thoracic and vascular surgery, aortic dissections and aneurysms are similarly treated by implanting a stent graft inside the aorta under X-ray control [64, 131]. A novel, related procedures is the endovascular implantation of artificial heart valves [57, 210] for high-risk patients, who would not survive the stress of the traditional surgical treatment approach. Likewise in abdominal surgery, laparoscopic interventions offer accepted minimally invasive treatment alternatives for many clinical indications requiring surgical intervention, e.g. laparoscopic cholecystectomy [91] or appendectomy [205]. Minimally invasive treatment of cancer, e.g. liver cancer ablation [26] or sentinel lymph node biopsy for early stage breast cancer patients [127, 128] are two more examples for today already successfully used minimally invasive interventions.

The major common challenge of all minimally-invasive procedures is the need for interventional therapy guidance, supported by medical image processing and real-time visualization technology. Without an open situs, as in traditional surgery, the physician is forced solely to rely on pre- and intra-operative image data for navigation and decision guidance. Medical image processing not only provides methods for diagnosis and pre-operative treatment planing, but also bridges the gap into the intervention room, and supports navigation and treatment guidance during the procedure. By providing detailed navigational aid and treatment control to the physicians, the outcome of the procedure is improved.

Need for Efficient Medical Image Processing Solutions

With every improvement in scanning technology, more and more image data is generated, e.g. an improvement in twice the spatial resolution per image dimension will result in four times the data for 2D imaging and eight times for 3D imaging. All this data has to be efficiently processed by a specialized medical image processing pipeline, from acquisition, reconstruction, image enhancement, image processing, to visualization. The same holds true for real-time imaging modalities, where the data has to be processed and visualized often within a very small time frame to guarantee real-time, lag-free presentation to the physicians. The general trend to minimally-invasive, image guided procedures furthermore postulates the request for efficient, fast medical image acquisition and processing solutions to facilitate image-based registration and tracking for navigation and real-time visualization in the intervention room.

1.1.2 High Performance Computing on GPUs

General purpose Central Processing Units (CPUs) have for a long time been able to process data fast enough for most applications. But with the increasing data volumes generated by imaging modalities, simulations and scientific computations current CPUs have performance-wise been falling behind modern Graphics Processing Unit (GPU) at least one or two orders of magnitude for many parallel computing problems. Over the recent years, GPUs have evolved from fixed function, dedicated graphic co-processors into highly parallel, fully programmable, general purpose processors.

Modern GPUs feature hundreds of so called programmable shaders or stream processors, and an extremely fast bus system with huge data transfer rates. Although clocked at lower frequencies than their CPU-brethren GPUs can outperform CPUs many times if all stream processors are active and the latency of memory access can be hidden by numerical computations executed by a large number of scheduled threads on the hardware. CPUs on the other hand clearly outperform GPUs for sequential or only low-level parallel computing problems.

However, many of the problems encountered in science and medical image processing can be solved by data-parallel algorithms, inspiring researchers to use GPUs as co-processors to accelerate numerical computations. Initial research led to the development and foundation of the general purpose computation on GPU (General Purpose GPU (GPGPU)) concept [65, 125, 148]. Numerous recent works have already proposed (GP)GPU technology to address and accelerate various problems in medical imaging in the areas of medical image data visualization [36], registration [186], segmentation [60] and general image processing [100].

1.2 Objectives

The objectives of this work are to investigate the application of GPUs and GPGPU to accelerate and improve existing and realize novel algorithms and methods for automatic mono- and multi-modal fusion, real-time visualization and medical augmented reality. In the following listings the individual prime objectives and sub-objectives are briefly highlighted.

- Develop a GPU-accelerated visualization framework for high quality visualization. Support different medical imaging modalities, CT, MRI, SPECT, US and Microscopy data of 2D, 3D and 4D dimensionality.
- Integrate state of the art, GPU-accelerated visualization techniques into medical Augmented Reality (AR) framework for advanced in-situ AR visualization. Improve visual perception by combination of advanced rendering and image processing techniques. Evaluate impact of visualization effects on navigation task in experiments with physicians.
- Accelerate multi-modal US-CT fusion by executing the computationally most expensive tasks on GPUs.

  (1) Accelerate patient-specific ultrasound simulation from CT for medical training and multi-modal fusion.

  (2) Develop GPU-accelerated registration framework for mono- and multi-modal fusion of medical image data. Investigate acceleration potential of pair-wise rigid, simultaneous rigid and deformable registration approaches.

1.3 Contributions

In the course of this thesis several algorithms and methods have been contributed to the medical image analysis and medical augmented reality community in form of papers published in international conferences and journals.

- **State of the Art GPU-accelerated Volume Rendering**. The renderer developed and implemented within the scope of this work has contributed to several publications, e.g. real-time volume rendering of 4D CT data and aortic valve models [80], real-time visualization of simulated ultrasound [110, 111], 2D-3D registration of X-ray and CT-data [31, 32, 240], medical augmented reality visualization [10, 19, 109] and fused volume rendering of SPECT and CT data [227, 228].

  Chapter 4 discusses in detail the basics for GPU-accelerated direct volume rendering, provides an overview of related work in volume visualization community, and how to set up a flexible and high performance GPU ray-caster. The chapter concludes with a series of visualization case studies for different medical applications.

- **High Quality Medical AR Visualization** In this work GPU-accelerated volume rendering has been integrated into an existing medical AR framework. The rendering pipeline is adapted and optimized to meet the constraints of real-time and lag-free visualization on a stereo video-see through HMD. Furthermore, it addresses one fundamental augmented reality problem: interaction and occlusion of real and virtual objects. We present a solution producing correct renderings by combining video frame image processing with depth buffer analysis on the GPU. The system performance and clinical usability for surgical navigation tasks is evaluated within experiments with several experienced surgeons. The system has been presented in workshops at international conferences [12, 109] and conferences [10, 19]. Chapter 5 discusses in detail the AR system, the challenges faced, lessons learned and the reworked rendering pipeline.

- **GPU-accelerated US simulation from CT** supporting different simulation models for either multi-modal fusion or visual convincing images for medical training scenarios. Chapter 6 discusses in detail the basics and methods for simulating US from CT and how to implement a ray-based simulation in general on a GPU. Parts of this work have been published in [19, 110, 111] focusing on patient-based simulation and real-time visualization and in [112] for accelerating simultaneous multi-modal registration by patient-specific US simulation from CT. This work has also been used in first studies for successful deformable US/CT registration of real patient data [225].

- **GPU-based US-US and multi-modal US-CT Registration**

  Previous work [218] on simultaneous US-US registration is ported to the GPU, enabling for the first time to simultaneously register multiple real clinical datasets with standard resolutions in a reasonable amount of processing time compared to previous CPU implementations. Furthermore, an extension of the mono-modal US-US registration approach to multi-modal US-CT registration is presented. Results of this work have been presented in [112]. Besides, we have investigated simultaneous registration of 3D Power-Doppler data.

  We also present a framework for fully automatic dense deformable registration of CT to 3D US data, implemented entirely on the GPU. First results with real patient data sets are presented in this work and [225]

  Chapter 7 focuses on GPU-accelerated image registration and discusses the newly developed methods for pairwise and simultaneous rigid, and deformable intensity-based US-US and US-CT registration.

  Furthermore, the GPU registration framework has been used in several publications and research projects addressing various registration problems. 2D-3D X-ray with CT(A) registration: Contrast removal during DRR generation to address image dissimilarities [31], 2D-3D and 3D-3D image registration using Markov Random Fields (MRFs) and discrete optimization [239, 240], and 2D-3D registration for virtual fluoroscopy using a camera mounted mobile C-arm (CAMC)[32].

CHAPTER 2

Medical Imaging - Methodology and Background

Todays clinical routine can hardly be imagined anymore without the pervasive presence of medical imaging technology. Images and data generated by various different imaging modalities are routinely used for diagnosis, treatment control and long-term monitoring. Internally all modalities are powered by sophisticated methods for medical image analysis, image processing and visualization. The field of medical imaging related technology, modalities and algorithms, is vast and it is impossible to cover every aspect of the many different general and problem specific solutions developed in the recent years within the scope of this thesis. Nevertheless we want to provide the reader with the necessary and relevant background information about medical imaging modalities and fundamental concepts of medical image computing, intra-operative navigation and visualization related to and covered by this work.

2.1 Imaging Modalities

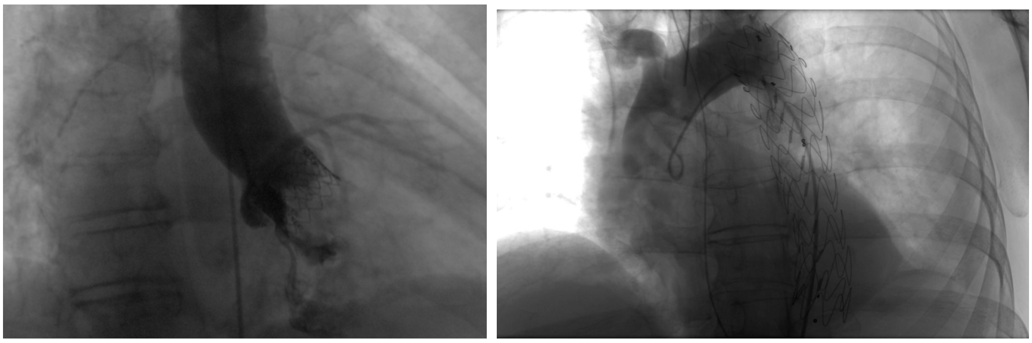
Numerous imaging modalities have been developed and introduced into clinical routine over the last decades. They can be classified mainly into anatomical and functional imaging modalities. Anatomical imaging modalities, through usage of various underlying physical principles of biological tissues, provide image data purely containing anatomical information. Anatomical image data is frequently used for diagnosis, e.g. tumor localization, lesion quantification, and for image-guided interventions, e.g. implant placement control. Functional imaging modalities on the other hand directly provide information about biological processes in tissue. The information is often measured or imaged indirectly, by introducing a contrast agent or tracer that amplifies or enables first of all the imaging process.

In the following paragraphs we will present an overview on the modalities, X-ray, CT, MRI and US, we mainly worked on within the scope of this thesis. The overview is concluded by a summary on further imaging modalities and recent trends and developments.

2.1.1 X-ray Imaging

X-ray imaging in medicine is based on the physical properties of X-ray attenuation in biological tissues. The basic building blocks of X-ray imaging devices are a X-ray source (Emitter), intensifier (not needed anymore in current digital flat panel X-ray devices), and X-ray detector. The source contains an X-ray tube, that generates and emits X-ray photons of a specific energy. The photons are then sent through the to be scanned object and travel towards the detector. During object traversal, the X-ray radiation is attenuated based on the specific properties of the tissues along the beam. The measured energy on the detector surface corresponds to the emitted energy minus attenuation, producing the characteristic 2D black/white X-ray images. The geometric and image properties of X-ray devices are often described by optical camera models frequently used in the computer vision community [41, 66, 236].

However, with the major difference that X-ray images are projection images in contrast to the surface images produced by optical cameras. For an X-ray camera model the rays travel from the camera through the object and hit the image plane at the detector, instead of being reflected on object surfaces towards the camera. An additional difference is the physical separation of source and detector compared to optical cameras and that the imaged object is located between them.



(a) (b)

Figure 2.1: Examples of 2D X-ray images used for guiding minimally invasive cardiac procedures. (a) X-ray image depicting aortic valve implant in aortic root. (b) X-ray image for catheter guidance during endovascular thoracic aortic stenting procedure.

X-ray images are frequently used in clinical application domains where high difference in attenuation provides good contrast of the structures of interest. This applies primarily to the skeleton and bony structures in the human body, due to the high difference in attenuation of bone and soft tissue.

When combined with a radiopaque contrast dye introduced into the blood flow, X-ray images are used to visualize vascular structures. Often a digital subtracted angiography (DSA), visualizing primarily vascular structures, is generated by subtraction of contrasted and non-contrasted images. X-ray devices, either mobile or stationary, are frequently used for monitoring interventions, e.g. examinations with an intercardiac catheter in cardiology, aortic stent placement or in orthopedic surgery. The good contrast and visibility of medical devices, e.g. wires, catheters, in the X-ray images and the possibility for real-time fluoroscopic imaging are two of the main reasons for their application in the intervention room. Latest generation X-ray devices offer the capability to reconstruct 3D volumes by rotating the C-arm about 190°around the object.

(a) (b) (c)

Figure 2.2: Examples of currently clinically available X-ray devices. (a) Mobile C-arm used in intervention rooms. (b) Robotically controlled C-arm (Siemens Dyna CT) with 3D volume reconstruction capabilities. (c) Latest generation robotically controlled C-arm (Zeego) with two rotation centers for extended field of view 3D Cone-Beam reconstruction. All images are courtesy of Siemens Healthcare.

However, besides the aforementioned strengths, X-ray imaging comes with the drawback of ionizing radiation applied to the patient and to an even higher extent to the physician. Although protected by a lead vest, the radiation accumulated over multiple interventions and time poses a non-ignorable health risk to the clinical staff. Furthermore, possible contrast dye complications for the patient, e.g. kidney failure, often limit the amount of imaging time in the intervention room.

2.1.2 CT - Computed Tomography

Computed Tomography imaging shares the underlying physical imaging principles with X-ray imaging. However, instead of generating only single or a series of 2D projection images over time for the same perspective, computed tomography devices reconstruct 3D imaging data from a set of multiple views by rotating the source and detector at least 180°around the object. Scans of larger regions can be performed by moving the object axially at constant speeds through the rotating X-ray source and detector configuration (gantry). Currently available stationary CT scanners have a X-ray source and a two-dimensional detector element array, also known as multi-slice CT, on the opposite side mounted inside the scan gantry. During examination the gantry is rotating very fast, up to multiple full rotations per second, while the patient bed is moved at constant speed through it. The combination of gantry rotation and linear patient bed movement results in a spiral scan path, the resulting raw scan data is then reconstructed into a rectilinear 3D data volume using tomographic reconstruction algorithms [42].

Over the recent years the slice number, accelerating the imaging processing, has been increasing constantly, currently up to 128 slice configurations. The most recent development is Dual Intensity CT scanner technology. A second X-ray source and detector configuration is mounted in a 90°angle to the first inside the gantry. Dual Source CT enables faster imaging with reduced radiation and enables interesting research possibilities to improve tissue classification, image analysis, and visualization from the multivariate data.


(a) (b) (c)

Figure 2.3: (a) Stationary CT scanner system (Siemens Somaton Definition, image courtesy of Siemens Healthcare) (b) 2D slice extracted from CT slice stack of liver CT scan. (c) 3D direct volume rendering of abdominal CTA data set, depicting kidneys, abdominal vasculature, bones and skin surface.

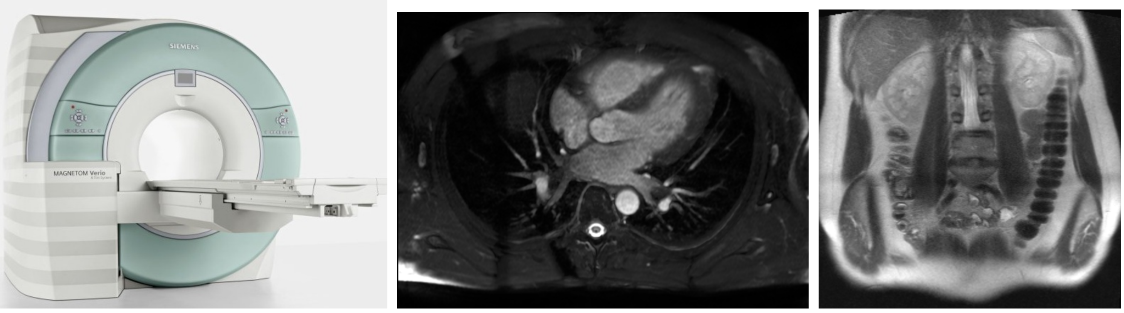
As in X-ray imaging, contrast dye can be added into the bloodstream during acquisition, the examination is then denoted Computed Tomography Angiography (CTA). CTA data is mainly used to visualize vascular structures, e.g. internal bleedings, vessel volumetry, in the Volume of Interest (VOI) and frequently used for diagnosis and treatment planning in cardiology, cardio-thoracic, vascular and abdominal surgery.

The fast imaging times of modern multi-slice CT scanners even allow acquisition of 4D dimensional data for study of dynamic processes and movement of anatomy of interest. Frequently gating techniques, for compensating breathing and heart-beat motion, are used to acquire high-resolution artifact free volumes over time.

One important aspect of CT imaging is the use of the Hounsfield scale to describe the radiodensity in Hounsfield Unit (HU) for every sample location in the scan volume. For many applications this quantitative scale facilitates the use of the same or very similar settings for image processing algorithms to process the data generated by CT scanners. For example the design of a classification table for either visualization or segmentation can be directly done based on the relationship of specific tissues and their HU values or windows, e.g. air has a value of -1000HU, fat -120HU, bone 400HU and greater, instead for every dataset.

2.1.3 MRI - Magnetic Resonance Imaging

Magnetic resonance imaging is both an anatomical and a functional imaging modality. It is based on measuring the radio-frequency signals emitted by hydrogen atoms, inside the human body. Protons of hydrogen atoms are first aligned along the field lines of a very strong magnetic field generated within the gantry of the scanner. The protons are then manipulated by an additional pulsating magnetic field. During excitation decay the protons emit radio-frequency signals that are detected by the scanner. Due to the tissue-specific concentration of protons the signals can be related to individual tissues in the human body.



(a) (b) (c)

Figure 2.4: (a) Stationary MRI scanner system (Siemens Verio, image courtesy of Siemens Healthcare) (b) 2D cardiac MRI scan (c) 2D abdominal MRI scan.

Similar to CT and X-ray imaging, solutions have been developed to acquire contrasted images of vascular structures using MRI scanners, often denoted Magnetic Resonance Imaging Angiography (MRA). The methods use either a contrast agent injected into the bloodstream, or exploit blood flow properties to visualize vascular structures.

A major benefit of MRI compared to CT imaging is the absence of ionizing radiation, and the much higher soft tissue contrast. Various scanning protocols allow to acquire both anatomical and functional image data with the same scanner hardware. However, the relative long acquisition times currently obstruct the use of MRI for 4D imaging. Because of its advantages MRI is frequently used for soft tissue imaging in diagnosis and treatment planning and follow-up in many medical disciplines, e.g. neurology, cardiology, and oncology. The interventional use of MRI [196] however is complicated by the strong magnetic field generated by the scanner. The whole intervention room, every instrument, and device [104] has to be fully MRI compatible. This results in significant economic investments for the hospitals. Therefore, interventional MRI is currently available only in selected hospitals, mainly research centers.

2.1.4 US - Ultrasound Imaging

Multiple parts of this worked are centered around US imaging and focus on simulation (see chapter 6) and registration of US image data with other imaging modalities (see chapter 7). Thus, we want to provide a more detailed overview about US imaging and its physical principles compared to the other imaging modalities discussed in this chapter.

Ultrasound is one of the most popular and widely used imaging modalities in today's clinical practice. For numerous reasons its use in diagnosis, therapy and image-guided interventions is desirable: (1) real-time imaging with high temporal resolution, (2) it is risk-free (radiation-free and non-hazardous), (3) US systems are relatively inexpensive compared to other imaging technologies, e.g. CT or MRI, (4) US systems are mobile, even portable system are available, (5) specialized US probes are available for almost every clinical application and (6) US probes can be used to target small tissue interfaces in endoscopic, laparoscopic, and intravascular applications.

Diagnostic US is used ubiquitously in medicine today, from general practitioners to specialists in almost every medical field. Echocardiography for visualization of cardiac anatomy and function, or obstetric US to monitor the developing fetus during pregnancy, just to name two examples. By utilizing the energy transported by ultrasonic waves, US can be directly used to apply therapy. Huber *et al.* [79] use a focused US beam for the treatment of breast cancer. The energy of the focused US beam induces temperature elevations at the focal point, which causes instantaneously cellular death and vascular obliteration in normal and tumor tissue. Another therapeutic application of US is the fragmentation of kidney and gall bladder stones [35]. Recently, interventional US imaging guided minimally invasive interventions, e.g. US imaging controlled placement and positioning of catheters, needles, and surgical instruments, are gaining more and more importance and interest in the medical community. US imaging is the only imaging modality that allows continuous, real-time, non-invasive image acquisition without imposing a health risk to patient or physician. Transesophegeal Echocardioagraphy (TEE) is used routinely for monitoring minimally invasive procedures in cardiology and cardiothoracic surgery. Recent works investigate real-time cardiac 3D US image guidance to facilitate off-pump cardiac surgical procedures, e.g. minimally invasive aortic or mitral implantation, [122, 199].

However, a major obstacle for realization of any US imaged guided intervention is the interaction of the ultrasonic sound waves with the used instruments and the resulting artifacts and reduced quality of the US images. For an excellent overview of instrument caused artifacts see [76].

2.1.4.1 US Image Acquisition

In a typical US examination the physician will place the US transducer probe onto the skin of the patient over the anatomy of interest. A special coupling gel is used between transducer and skin to avoid any air in-between, to improve transmission of ultrasound waves from transducer into the patient's body. During acquisition the transducer alternates between send and receive modes. In send mode it emits short ultrasound pulses with a frequency between 1 – 15 Mhz. It then switches into receive mode and waits for echoes of the sound pulse reflected towards the transducer. While the sound pulses traverse the patient's body they are subject to interaction effects, e.g. reflection, scattering and attenuation, with the encountered tissues. Echoes towards the US transducer are generated by either reflections at major tissue interfaces, or by tissue-specific scatterers, and make up the majority of the US image information. The echoes and their temporal delay are recorded by the transducer, stored and make up the so called scan line data. In B-Mode imaging the scan line data is, after some signal post-processing, e.g. amplification, noise removal, and often scan-conversion, displayed as a 2D Cartesian brightness image on the screen. Each 2D US image representing a cut plane through the patient's anatomy orientated downwards from the US transducer surface.

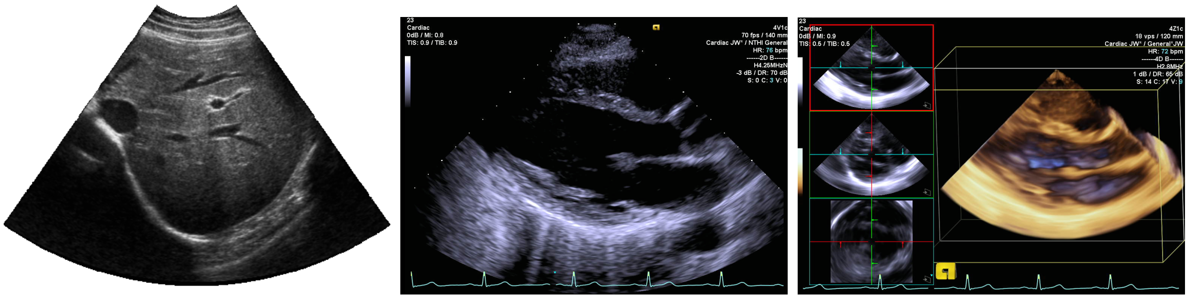

(a) (b) (c)

Figure 2.5: Examplary US images (a) 2D US images of liver, depicting major vessels. (b) 2D cardiac US image, (c) 3D volume visualization and MPR of 3D cardiac US data. Images courtesy of Dr. Wolfgang Wein, SCR.

View and tissue dependent US imaging effects make US image acquisition and interpretation challenging in general. While traversing the body tissues the sound wave is attenuated by various effects, mainly view-dependent reflection, tissue-specific and view-independent scattering and tissue specific absorption. Strong reflections at interfaces with large differences in acoustic impedance, e.g. air soft tissue interface in the lung, or bone and soft tissue interfaces result in shadowing of the region behind the interface. Localized high absorption is another source for shadows in US images, as the US pulse is completely absorbed locally and regions further away cannot be imaged. Speckle, cause by scattering, might be superficially considered mainly as noise. However, speckle is view-independent and in many cases can reveal valuable diagnostic information, e.g. inflamed tissues or cysts show a different speckle pattern than regular healthy tissue. There are many more US imaging effects and artifacts, some are discussed in chapter 6, for a detailed introduction and presentation we refer to [233].

2.1.4.2 US Transducers

For the various different application areas of US in medicine, specially designed US transducer arrays have been developed, namely phased, linear, curvilinear and matrix arrays. The differences lie mainly in the geometry of the US field of view formed by the different transducer types.

Internally US transducers use an array or matrix of piezo-electric elements, for generating the ultrasound pulses and transforming the received echoes into electrical signals. The piezo-electric elements have the properties of (a) changing size, either by compaction or compression when a electrical current is applied, and (b) generating a voltage when being subjugated to mechanical stress, e.g. deformation by external force. The first effect is used for generating the sound pulses in send mode with the desired frequency and amplitude by making the elements vibrate by applying alternating voltages. The second effect is used when the US transducer is in receive mode to convert echoed, incoming sound waves into electrical signals. The incoming sound wave induces mechanical stress to the elements which generates a voltage, which is proportional to its intensity.

Beamforming is used to steer and focus the US sound beam during acquisition by controlling a group of neighboring array elements to emit sound pulses and wait for echoes. Phased arrays (see figure 2.6(a)) produce a fan shaped US field of view, by angulated beam steering. During send and receive all array elements are active. The relatively small footprint of phased transducer arrays makes them ideally suited for scanning applications with limited and narrow scanning windows, e.g. between neighboring ribs in trans-thoracic echocardiography, or US imaging of the kidneys.

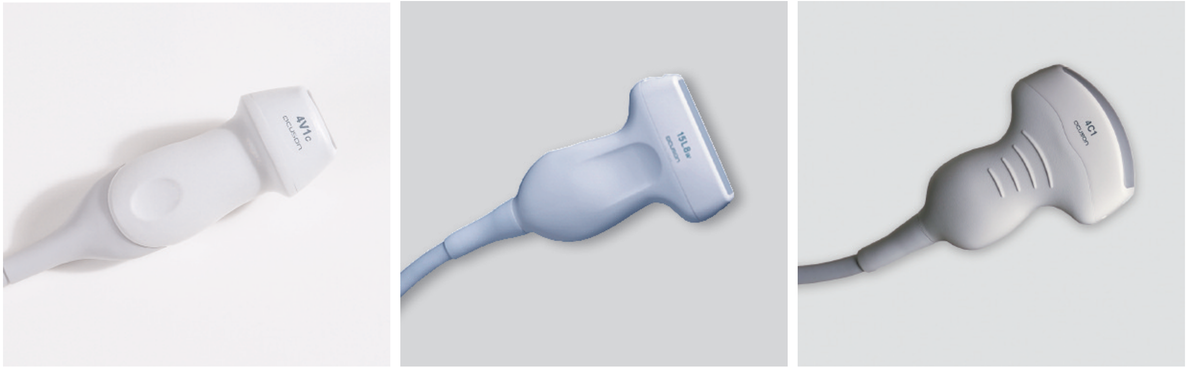
Linear arrays (see figure 2.6(b)) have a flat transducer surface and use parallel beam steering, by sequentially triggering a group of active elements from the left to right side of the array to generate a beam for each scan line. Linear arrays are mostly used for generating rectangular, high resolution, small field of view images of structures close to the transducer surface. Therefore, relatively high scanning frequencies are used limiting the depth penetration to a few centimeters. Curvilinear arrays (see figure 2.6(c)) are similar to linear arrays, with the difference of having a curved transducer surfaces and in general featuring much more transducer elements that are arranged in a convex shaped array. They produce large and wide field of view images, similar to phased transducers. However, due to the larger number of array elements, the images are in general of higher quality. Curvilinear arrays are frequently used for scanning applications where the large transducer footprint is not an issue, e.g. US imaging of abdominal anatomy.

For selected applications, US imaging from outside the human body is not sufficient or applicable. In many cases it is simply not possible to scan the region of interest, due to occlusion or simply insufficient depth penetration through tissue layers. To overcome this limitation, US probes were developed which can be inserted directly into the patients body through either natural orifices, small incisions, or percutaneously to allow unobstructed US imaging from inside the body close to the Region of Interest (ROI). Especially in echocardiography, transesophegal (see figure 2.6(d)) or endovascular US probes, are used to produce artifact free US images of the beating heart.

2.1.4.3 3D Ultrasound

Traditional US is a two-dimensional real-time imaging modality. For many tasks, such as image understanding and exact measurements of anatomical structures and lesions, the real-time acquisition of three-dimensional data of the patients anatomy is desirable. 3D US imaging [43, 140] is a relatively new advancement of US imaging and is slowly but constantly gaining more and more importance in today's clinical applications. There exist mainly two approaches for acquiring 3D US image data: (a) Freehand 3D US imaging and (b) hardware 3D US probes.

Freehand 3D US works by attaching a pose (position and orientation) sensor, close to the transducer surface of a conventional 2D US image probe. When scanning the anatomy of interest, the pose of the sensor is measured by a tracking system, and recorded jointly with the 2D US images. In order to generate a 3D volume representation from the set



(a) Phased (b) Linear (c) Curvilinear

(d) TEE (e) 3D Wobbler (f) 3D Matrix

Figure 2.6: (a-f) Different US transducer models. (a) Phased Array Probe, (b) Linear Array Probe, (c) Curvilinear Array Probe, (d) TEE probe, (e) 3D US Wobbler probe, and (f) 3D US probe with 2D transducer matrix. All images courtesy of Siemens Healthcare.

of tracked 2D US images a compounding step is necessary. For each 2D US image pixel, its 3D position is computed by transforming it to the 3D reference coordinate system, in general the tracking system coordinate frame. The position $\boldsymbol{P}_{TrackSys} = [x, y, z, 1.0]$ in tracking system coordinates (TrackSys) of a pixel $\boldsymbol{P}^i_{US2D} = [x_{US2D}, y_{US2D}, 0.0, 1.0]^i$ in 2D US image coordinates ($US2D$) for the i-th US image frame is given by the following equation:

$$\boldsymbol{P}_{TrackSys} = {}^{TrackSys}\boldsymbol{H}^i_{Sensor} * {}^{Sensor}\boldsymbol{H}_{US2D} * \boldsymbol{P}^i_{US2D} \tag{2.1}$$

where ${}^{TrackSys}\boldsymbol{H}^i_{Sensor}$ is the transformation from tracking sensor coordinate system (Sensor) to the tracking sytem coordinate frame (TrackSys) for the i-th ultrasound frame and ${}^{Sensor}\boldsymbol{H}_{US2D}$ the transformation from 2D US image coordinate system to tracking sensor coordinate system. sytem. For every 3D voxel position in the compounding volume, intensity values from one or multiple 2D US images are combined to yield the final voxel intensity value. For a review of compounding methods we refer to [162]. Prior to acquisition and compounding the parameters of ${}^{Sensor}\boldsymbol{H}_{US2D}$ need to be estimated by a one time calibration step, which has to be performed every time the pose sensor is re-attached to the transducer. For a survey of freehand US calibration methods see [135].

The benefit of 3D Freehand US is clearly that one can relatively easy extend any conventional 2D US system to a freehand 3D US system by mounting a pose sensor onto the US probe and performing the calibration. Except the tracking system, no additional hardware is required. A limitation of 3D Freehand systems is the low temporal resolution, which requires breath holds to acquire 3D volumes without breathing motion artifacts, and in general does not allow for 4D imaging.

Hardware 3D US probes Today, almost all US system vendors offer 3D US probes, which allow direct acquisition of 3D and 4D US data. Two different 3D probes technologies exist. The first one uses conventional 1D US imaging array extended by a mechanical steering mechanism inside the probe housing to either move the 1D array in elevational direction or rotate the 1D array about a construction determined rotation axis. Figure 2.6(e) depicts a 3D US Wobbler probe. The probe housing is very large and bulky, restricting the usage to applications where the large transducer footprint is no issue. Image quality is fairly comparable to the traditional 2D US probes, mainly to the shared array technology and element number. However, image acquisition speed varies largely with the selected level of image quality and features, e.g. Doppler imaging. Frequently, breathing and heart beat gating are required for 4D acquisitions as 4D acquisition speed is too low.

Recently developed Matrix transducer array 3D US probes (see figure 2.6(f)) allow for direct real-time 3D and 4D US imaging by electronically steering US beams in the lateral and elevational direction. Compared to mechanically steered 3D US probes, they allow for much faster imaging, up to several volumes per second. However, the image quality is currently limited by the number of elements in the transducer matrix. The possible number of elements is mainly limited by the transducer cooling systems to avoid overheating. However, despite the technical difficulties several 2D matrix US systems have been developed in the recent years. Even TEE probes, and endovascular catheters have been equipped with 2D matrix technology and provide excellent real-time 4D US imaging of the beating heart.

2.1.5 Further Imaging Modalities

Besides the described imaging modalities in the preceding paragraphs, there exist many more modalities. Functional, nuclear radiation based, imaging modalities, e.g. SPECT and PET, are very important tools for the diagnosis and localization of cancer tumors, metastasis but also inflamed tissue regions [229]. Both modalities reconstruct a 3D data volume using tomographic reconstruction. In contrast to CT and MRI the scanners are not made of pairs of source and detector elements, but are only equipped with detector elements. Both modalities detect gamma rays emitted from the patients body itself. Therefore, a radioactive tracer is injected into the patients bloodstream which accumulates in functional active tissue of interest, e.g. tumor cells, inflamed soft tissue or bones. In most cases the radioisotope is piggybacked onto a sugar molecule like Fluorodeoxyglucose.

SPECT reconstructs 3D data from a series of 2D projection images of different angles, acquired with one or multiple gamma cameras mounted inside the gantry. PET imaging provides a higher spatial resolution, as in contrast to SPECT it does not record directly gamma rays produced by the radioisotope but a pair of gamma rays, gamma photons, produced by the annihilation event of a positron emitted by the tracer and an electron from the tissue. Inside the gantry a ring of detector elements detects the gamma photons pair hitting the ring surface on opposite sides. Only photon pair events occurring in a small temporal window are recorded, others are discarded. The resulting raw data is then reconstructed into 3D image space using tomographic reconstruction. As both PET and SPECT do only allow to acquire functional imaging data, the images are often combined with data from anatomical imaging modalities, e.g. CT or MRI to provide contextual anatomical information and cross-check findings from these modalities. Recent efforts focus on the construction of integrated scanners, able to acquire multi-modal data with a single scanner device and examination, e.g. SPECT-CT [178] or PET-CT [18, 39].

With the rapid adaption of minimally-invasive procedures for diagnosis and therapy in todays clinical practice, live imaging feedback of the situs through small entry ports into the body has become a must. For many interventions, e.g. gastroscopy, bronchoscopy, choleosectomy video-imaging is possible by special devices [91, 211]. A variety of instruments has been developed to that end, that can be inserted into the body through small incisions or natural orifices. The main components are in general a light source, an optic system to transmit images from the tip to the other end, and a video camera or eyepiece at the proximal end. Often a working channel for inserting instruments, e.g. needle, scissors, is available. The instruments are available in various sizes and both rigid (e.g. bronchoscope, laporoscope) [136] or flexible (e.g. bronchoscope, gastroscope, colonoscope) configurations. Latest development focus on integrating other imaging modalities into the tip, e.g. US probes, optical imaging probes, creating hybrid systems enabling video-based navigation and enhanced inspection/diagnosis by the additional imaging modality [51].

The evolution of existing and development of new imaging modalities is an ongoing process and will continue for the coming years. Recent trends and developments are for instance molecular imaging [21, 182], optical imaging, and thermal imaging, In general there is also a trend in the modification or combination of existing modalities for interventional applications.

2.2 Medical Image Computing

In general data acquired by the different imaging modalities is processed by multiple different algorithms. In the following paragraphs we briefly described the major domains of medical image computing. Although this thesis does not address algorithms from all domains, the presented methods rely and would not be possible without the other domains. Within the scope of this thesis we focused mainly on algorithms for simulation, registration, real-time visualization, intra-operative navigation and visualization.

2.2.1 Raw Data Acquisition

Raw data is acquired and stored during the examination by a scanning device, e.g. CT, MRI or US. In most cases the raw data is not directly acquired in a rectilinear 2D or 3D coordinate space, generally used for representing image data in later stages of the pipeline, but in a scanner specific scan coordinate space. Additionally raw data often contains noise and measurement outliers which have to removed prior processing by later stages. Transforming the raw data from the scan coordinate space to the image coordinate space is handled by the *reconstruction* stage. Data filtering, noise removal and enhancement is taken care of by the *Filtering/Enhancement* stage.

2.2.2 Filtering/Enhancement

Image filtering/enhancement is frequently applied before data is passed to the main image processing algorithms or visualization. Often multiple filters are applied during data traverses the image processing pipeline. E.g. high-frequency noise has to be removed by a smoothing operator, e.g. Gaussian blur, or by a non-linear filter, e.g. median filter. Many filter or enhancement operations are applied not only to improve the quality of the data for visual perception by a human observer but more importantly also for later on applied algorithms, e.g. angiographic image enhancement by vesselness filters [47] prior segmentation, or speckle reduction [2, 3, 33] in US imaging. In general filtered data with less noise but preserved edges is e.g. much easier and more stable to process by a segmentation algorithm, and result in smoother cost functions for intensity-based image registration.

2.2.3 Reconstruction

Reconstruction is the term often used for transforming data from the native scan coordinate space into the image coordinate space, either a 2D or 3D rectilinear grid storing sampled intensity values at discrete locations. In a more general sense reconstruction can be understood as generating an n-dimensional representation of the data from a set of k-dimensional data. This works both ways, from lower-dimensional to higher-dimensional data, e.g. reconstructing a 3D CT volume from a series of 2D projection images [42], or compounding a 3D US volume from a set of oriented 2D US slices [162], as well as from higher-dimensional to lower-dimensional data, e.g. generating 2D Multi Planar Reconstructions (MPRs) from a 3D dataset. High-quality reconstruction is in most cases time consuming even on high-end hardware. Thus, many recent works have addressed this issue by proposing GPU hardware accelerated solutions [90, 179] or parallelization on multi CPU machines. In the scope of this work we have developed a method for GPU-accelerated 3D US compounding [89, 90]. The compounding algorithm is used both for generating 3D US volumes for 3D volume and 2D MPRs visualization as well as compounding simulated US volumes for mono-modal and multi-modal registration (see chapter 6 and chapter 7).

2.2.4 Segmentation

Segmentation algorithms are often applied to separate the anatomy of interest, e.g. vasculature, tumor tissue or whole organs from the rest of the data. The resulting partitioning in a set of two or multiple labels, each one for a specific anatomy of interest is of high value for diagnosis, therapy planing and treatment control. One of the key challenges is to compute the correct classification label for every pixel/voxel automatically. E.g. partial volume effect can lead to ambiguous situations, making a pixel/voxel a wrong candidate for multiple or zero labels. Several segmentation algorithms have been developed, fully automatic, interactive, or manual ones, as well as binary and multi-label ones [20, 54, 55]. In this work we have not worked on any segmentation algorithm directly, but have used the results of different algorithms for various applications. In [31], we use a segmentation of the contrasted aorta to generate non-contrasted Digitally Reconstructed Radiographs (DRRs) from CTA data to improve 2D/3D registration with non-contrasted intra-operative X-ray images.

2.2.5 Simulation

Simulation has been an important tool in the last 30 years in the medical physics domain. Today it is also gaining more and more importance for real-time applications in the medical image computing domain. The application areas can be classified into three main groups.

- *Registration:* For specific registration applications direct computation of a similarity value is not straightforward possible, either because of non-matching dimensionality, e.g. 2D-3D registration problems [92, 108], or considerable differences in the image characteristics of two different modalities that can not be overcome even by sophisticated, difference tolerant similarity measures, e.g. US to CT registration [112, 222]. In this situations the imaging process is simulated by generating virtual image data of one modality from existing data, acquired with the same or a different modality, using a physically based model, e.g. DRR generation from CT or US image simulation from CT [222] or pre-recorded 3D US data. Within the scope of this thesis we have worked on accelerating DRR generation on GPUs for various registration applications [31, 240] and patient-specific simulation of 2D, 3D, and 4D US images from CT data [110, 111] for patient-specific training applications [19] and multi-modal image registration [112, 225].

- *Medical Training Simulators:* A problem in education of young physicians is often that the training process is time-consuming, requires the supervision of an expert, and that rare pathologies are mostly encountered only in textbooks or lectures but not real life. For optimal therapy and diagnosis the effective use and understanding of imaging modalities and therapy techniques is indispensable. To overcome the bottleneck of limited expert time, and access to rare cases, medical training simulators have been proposed. Thereby, medical students can effectively train image acquisition or new therapy techniques in a virtual sandbox. Multiple academic and commercial solutions have been proposed in the recent years. Wright *et al.* [231] use virtual reality to teach students radiographic positioning. Weidenbach *et al.* present an augmented reality simulator for 2D echocardiography training [221]. Magee *et al.* [129] and Zhu *et al.* [237] present systems for simulating needle placement using virtual US images. Soler *et al.* [192] present patient based surgical simulation for both US image guided and laparoscopic interventions.

- *Modeling:* The last and also vast area in medical image computing is simulation and modeling of physiological or physical processes for studying and understanding complex phenomena or simulate yet non-existing imaging hardware. Computational Fluid Dynamic (CFD) simulations are frequently used to simulate blood flow inside vessels, with special interest on diseased regions. In [73] the hemodynamics of cerebral vasculature and aneurysms are modeled based on patient-specific models extracted from CTA data. Blood flow analysis is also a major interest for therapy in cardiology and cardio-thoracic surgery. Simulations help to understand the complex interaction between blood flow dynamics and cardiac valve function, or help to predict the risk of aortic aneurysm rupture. Another field of simulation application is the design of novel imaging hardware. Numerical simulations of US wave propagation [83] are frequently used in initial US transducer design stages.

2.2.6 Registration

Bringing together information from multiple images/volumes into one common reference frame is the task of registration algorithms. Registration of multiple datasets is frequently employed for guidance in e.g. radiation therapy [92] or to aid navigation in minimally invasive procedures by fusing information from pre- and intra-operative imaging data. Fusion of multiple co-registered datasets from the same or different modalities furthermore improves diagnosis by providing complementary information from e.g. anatomical and functional imaging modalities, or global anatomical reference from larger field of views. Direct treatment control in the intervention room, e.g. resection control, or progression monitoring e.g. monitoring of tumor growth over time, as well as statistical anatomical atlas construction are just three more application examples.

A vast number of registration algorithms has been developed in the past years. The approaches can be classified into mono-modal, multi-modal, point or feature based , intensity based, rigid, deformable, pair-wise and simultaneous registration algorithms For detailed treatment of the subject we refer to chapter 7, reviews and surveys in literature [63, 67, 72, 130, 137, 232, 242].

In general a registration problem has multiple of the described properties and often a specialized algorithm is developed to solve a specific problem. In this work we have mainly focused on mono- and multi-modal fusion of US and CT, using different approaches (rigid, deformable, pair-wise, simultaneous). For a detailed report and discussion of the methods and results we refer to chapter 7. Furthermore, I was involved in research on 2D-3D registration of X-ray to CT data, as well as bronchoscopy video images to CT. For details we refer to the appendix chapter A of this work.

2.2.7 Visualization

Visualization is one of the key components of almost every medical imaging application. The challenges of any visualization system are direct, lag-free, high-quality visualization of medical image data. This is especially important for all real-time imaging modalities, e.g. US imaging, X-ray or video imaging, and for systems used in image-based guided and navigated medical procedures. Furthermore, visualization is used pervasively in medical imaging: interacting with medical image data, image manipulation and processing algorithms require visual feedback and input from the user, e.g. manual annotation, lesion labeling, segmentation seed placement, registration result overlay.

Today, medical image data is either visualized directly based on the pixel/voxel intensity values or indirectly as traditional polygonal geometry after an intermediate segmentation and mesh generation step. Frequently hybrid solutions, combining direct and indirect visualization approaches, e.g. mesh rendering of segmented organ from CT scan embedded in real-time 4D US direct volume rendering, are used in clinical practice. 3D and 4D data can be either visualized in 3D by direct volume rendering techniques or in 2D by re-slicing the 3D data with either axis-aligned or any-plane MPRs. 2D image data is often directly displayed on the screen or if the position and orientation of the image plane in 3D is known by rendering a plane texture mapped with the image.

Intelligent algorithms for automatically providing the appropriate visualization of the data are gaining more and more importance, e.g. automatically revealing insight onto inner structures or only visualizing specific tissues or anatomy of interest, e.g. by means of advanced tissue classification or automatic fuzzy segmentation computed on-the-fly during rendering.

Traditionally data is visualized on a single screen, however especially for minimally-invasive procedures and interventions radically new visualization devices and techniques are required, e.g. Medical AR visualization on stereo video or optical see-through Head Mounted Displays (HMDs) [172, 207] or projection of the rendered images directly onto the situs [124, 203].

The number of specially developed algorithms and solutions for medical image visualization is huge, and beyond the scope of this thesis. In this work we focus mainly on GPU-accelerated direct volume rendering, for (a) direct visualization of 3D and 4D data from scanners and produced by simulations, (b) generating DRRs for 2D-3D registration, and (c) integration into a medical AR framework for direct, pre-processing free, high-quality and real-time visualization on a stereo video see-through HMD.

For detailed presentation of medical visualization concepts we refer to chapter 4 and chapter 5 in this work. Chapter 4 provides an overview on the basic concepts of Direct Volume Rendering (DVR), GPU-accelerated DVR and advanced rendering techniques. chapter 5 presents the challenges of integrating and tuning the renderer for medical AR, where lag-free, real-time rendering in high-quality is required. Both chapters provide the interested reader with links to the state of art in the relevant literature.

2.3 Intra-operative Imaging, Navigation and Visualization

One of the developments with major momentum in modern clinical practice is the transition from traditional open surgery to minimally-invasive procedures and interventions. The obvious benefits for the patients are less trauma, reduced risk, increased eligibility of high-risk patients, smaller wounds and faster healing, and an earlier release back home from the hospital. Politics in the developed countries are favoring and forcing this movement because of the high potential for effective cost reduction in the public health care systems. However, without a direct view onto the operating situs as in traditional surgery, the physicians have to rely on real-time imaging and navigation solutions to effectively and accurately apply therapy. Interventional imaging, navigation and visualization brings together many different techniques from the preceding paragraphs in this chapter. Only by combining information from pre-operative and intra-operative imaging together with special instruments, tracking systems, and real-time algorithms for registration, segmentation and visualization minimally-invasive procedures have become possible.

In the following paragraphs we will briefly describe the techniques for tracking, navigation and intra-operative visualization, present in most systems for image-guided surgery and interventions.

2.3.1 Tracking

Tracking or spatial localization, of the position and orientation of medical instruments, imaging devices and the patient during a medical procedure is necessary to bring all information into one common reference frame and establish a virtual representation of spatio-temporally, co-registered data. For tracking of tools and devices several systems are available that can be classified into six groups: mechanical, acoustic, electromagnetic, optical, image based, and hybrid tracking systems combining two or more of the before mentioned techniques. In the following paragraphs, the for this work most relevant tracking solutions are briefly described.

- *Optical Tracking (OT):* The idea of optical tracking is to have one or multiple cameras observing a tracking target. The multiple views of the target allow the position and orientation to be determined with respect to the camera geometry. The target usually consists of a set of optical markers arranged rigidly. The geometry of the markers is predefined and has to be non-symmetrical to avoid any ambiguities in the orientation of the tracking target. At least three markers are needed for the estimation of the position and orientation of the rigid body in space. More markers on the rigid body, introduce redundant information, facilitate better visibility of the target, and improve measurement accuracy. Visibility of the tracking target and the accuracy of its estimated position and orientation in space are highly dependent on the arrangement of the optical markers on the tracking target, the arrangement of the cameras, and distance of the cameras to the markers.

Optical markers are either active markers, infrared Light emitting Diodes (LEDs), or passive markers (infrared light reflectors) in the shape of spheres or circles. A drawback of the active markers are the additional wires connected to each led. Optical tracking system allow the tracking of multiple tracking targets. For passive and active marker configurations different strategies exist. For passive markers, the configuration of the markers on each tracking target must be unique to avoid any ambiguities in the identification of the tracking target.

- *Electro-Magnetic Tracking (EMT):*

  The principle of electro-magnetic tracking systems is to compute the position of a sensor by measuring the induced electrical currents when the sensor (receiver) is moved inside a magnetic field generated by either an alternating current (AC) or direct current (DC) field generator (transmitter). The first paper on a electro-magnetic tracking system was published by Raab *et al.* [158] in 1979. Both, AC and DC systems are affected by ferromagnetic objects close to the transmitter or receiver, as well as electromagnetic fields generated by other electrical devices, e.g. electrical motors or power sources. Thus the use of electromagnetic tracking systems is challenging inside the operating room, where numerous ferromagnetic objects are moved through and close to the magnetic field. AC and DC systems are both affected by the presence of ferromagnetic materials, e.g. iron or steel, as they change the homogeneity of the magnetic field generated by the transmitter. Despite the problem of distortion in the magnetic fields due to ferromagnetic objects or electromagnetic fields, electromagnetic tracking systems have a unique, major advantage above all other tracking technologies. No direct unobstructed line of sight between the receiver and transmitter is required for the tracking system to work properly. Therefore, electro-magnetic tracking systems are the only type of tracking system that allow to track the position of flexible medical instruments that are inserted into the patient's body, such as endoscopes, or endovascular catheters.

The decision which tracking technology is used for a medical application depends heavily on application specific requirements, e.g. update rate, accuracy and precision, and practical usability of the tracking system within the clinical environment.

Today, optical tracking systems are frequently used in neurosurgery, orthopedic surgery and for tracking rigid endoscopic instruments in minimally invasive procedures. Several commercial solutions exist for these applications, e.g. BrainLAB's VectorVision®[1] or Medtronic's StealthStation®[2] neurosurgery planning and navigation system. Electro-magnetic tracking system are applied whenever the position and orientation of tools inside the patient's body have to be tracked. Electro-magnetic tracking is used for instance in BioSenseWebster's [3] Carto XP ®EP navigation system. Image-based tracking is an alternative to electro-magnetic tracking for applications where other tracking solutions cannot be applied. However, only for a selected number of clinical applications where real-time

---

[1] http://www.brainlab.com

[2] http://www.medtronic.com

[3] http://www.biosensewebster.com

imaging, e.g. X-ray or US, is possible and the position and orientation of instruments can be estimated robustly from one or multiple images in real-time.

2.3.2 Navigation and Visualization

The transition from traditional open surgery to minimally invasive procedures requires novel solutions for consistent and integrated presentation and visualization of the virtual and real data in the intervention room. Without a direct line of sight onto the operating situs the treating physician solely relies on information displayed on one or multiple monitors. The increased mental demand from mapping displayed pre- or intra-operatively acquired image data onto the situs and combining it with tracking and further information sources, e.g. breathing, heart rate monitor, has to be addressed by these solutions.

Traditional single monitor based visualization of medical data, as used in the radiology reading room, has the serious drawback of having the treating physician to split his attention between the situs and one or multiple information monitors continuously during the procedure. Ideally, a single display would serve him with all necessary information during the procedure.

Recent developments in display technology have made mass market production of very large, high-quality flat-screen monitors made feasible. Instead of multiple, small information displays one large display positioned directly above the situs in the field of view of the physician can be used, eliminating the need to look around to gather all information. Most recent displays offer the option to visualize data in 3D by varying the user's viewpoint, e.g. by slightly rotating the head, offering an improved depth perception without the need to wear special glasses or restriction to specific view points. However, in order to produce the stereo effect the data has to be rendered for a set of viewpoints, which can be a performance bottleneck for direct medical volume visualziation. In [166], an optimized direct volume rendering method for visualization of medical data on multi-view lenticular displays is presented addressing the specific requirements when rendering for multiple views at once.

Medical AR is an alternative to monitor based virtual reality visualization with great potential for application in minimally invasive procedures and image-guided navigation. Several AR-based solutions have been proposed for medical procedures using different augmentation techniques and devices. HMD-based medical AR is presented by Sauer *et al.* [173, 215] and Birkfellner *et al.* [16]. Fused real and rendered images of the patient, tracked instrument from the viewpoint of the user are visualized on the HMD's screen, providing real-time, integrated in-situ visualization of the real operating situs and virtual data. Projector-based medical AR is proposed by Tardif *et al.* [203]. The virtual data is projected directly onto the patient after a mapping between the projector and a camera image, from the surgeon's point of view, has been established. A combination of HMD and projector based AR is proposed by Low *et al.* [124]. Further works, present AR visualization for microscopes in the operating room [98, 156], endoscopes and laparoscopes [45, 49].

In [122], a monitor based augmented reality visualization of real-time cardiac image and virtual data is presented for supporting minimally-invasive, US imaging guided cardiac surgery. In [144], Novotny *et al.* present a system for stereoscopic display of real-time 3D US images for guiding surgical robotics. They report betters results for the stereo system compared to standard visualization for a study of five users performing surgical navigation tasks with a robot.

Navab *et al.* present CAMC (Camera Augmented Mobile C-arm), an AR-based system for improving navigation in surgery [139, 206]. A standard mobile C-arm is equipped with an additional optical camera. By a semi-transparent mirror construction and one time calibration the X-ray and camera images are co-registered. Interventionally fused camera and X-ray images are displayed to aid surgical navigation tasks. Extensions of the system have been presented to generate large panoramic X-ray images [220] and support intra-operative position of the C-arm using artificial fluoroscopy images [32].

Intra-operative imaging, navigation and visualization solutions are created by combining medical imaging modalities and medical computing techniques into one integrated solution. This facilitates the creation of a virtual model of the patient and the operating situs in the computer for interventional navigation and real-time visualization. However, when designing an intra-operative navigation solution one has to assess the overall system errors. Tracking systems, calibration (e.g. tool calibration, camera or visualization system calibration) and registration components each introduce errors into the system that have to be taken into account in order to assess the total system errors. The different error sources have to be modeled and properly propagated along the transformation chains to determine the overall error [7, 189].

Summary

This concludes the overview about medical imaging modalities and the medical image computing domain. In today's clinical practice many different combinations of images from different modalities, handled by specialized image processing pipelines are used everyday to improve the outcome of diagnosis and therapy for the patient. Ongoing research in both domains, imaging modalities and image computing, is continuously striving to improve existing technology and introduce new methods and tools to advance the state of art of medical treatment possibilities.

CHAPTER 3

High Performance Computing on GPUs

GPU performance, feature set, programmability and architecture has made an incredible development in the recent years. The rapid evolution from fixed function geometry processing accelerators to today's fully programmable co-processors for computer graphics and scientific computations was in the beginning primarily driven by the ever increasing demands for graphics performance and visual realism in modern computer games.

Today, commodity GPUs outperform current high end CPUs by at least one order of magnitude in terms of computational power and data throughput. These recent developments have made GPUs a very interesting platform for high performance and scientific computing. In this chapter, I will give a brief overview on the evolution of GPUs from basic video accelerators to the modern unified architectures for high performance computer graphics and computing. This is followed by a discussion of the major differences of today's CPU and GPU architectures and concluded by a brief excursion into GPU programming concepts and an outlook on future trends in GPU development.

3.1 Brief History of GPU Evolution

Many of the hardware and software designs found in today's GPU architectures and programming Application Programmer Interfaces (APIs) root back to its earliest ancestors, the large and expensive graphic workstations of the 1980s. From the 1980s to the late 1990s graphics hardware evolved in several generations from large workstations, to smaller workstation, to today's PC graphics accelerator card form factor. Every generation largely reduced the costs and size of the hardware, while at the same time increasing the performance compared to the previous generation.

The PC graphic accelerators of the late 1990s implemented fixed-function graphic pipelines and did not yet offer any programmable stages as today's GPUs do. Computer graphic applications were written in graphic APIs, such as Open Graphics Library (OpenGL) or Microsoft's Direct3D, that allowed global specification of geometry data, transformations, texturing setup, fragment shading and compositing parameters. In 1999, Nvidia introduced the Geforce 256 chip and concurrently coined the term GPU to describe the new architecture. The Geforce 256 was the first GPU to implement the complete OpenGL pipeline in hardware and introduced programmable register combiners, a first step towards programmable GPU hardware units.

Programmable GPUs Following the introduction of the first GPU a rapid evolution of GPU hardware and software followed mainly oriented along the releases of new versions of Microsoft's DirectX's, Direct3D and Shader Model specifications. In 2001, with the release of DirectX 8, NVIDIA's Geforce 3 and ATI's Radeon 8000 series GPUs, implementing Shader Model 1.x enabled the first programmable vertex and pixel shaders. The next generation of GPUs, e.g. NVIDIA's Geforce FX and ATI's Radeon 9000 series, implemented Shader Model 2.x, which introduced floating point texture formats and removed limitations for the number of texture fetches and texture indirections.

Concurrently with the rapid development of GPU feature sets and processing power, several GPU specific so called *high level shading languages* were developed to facilitate the creation of so called *shaders* for the programmable rendering pipeline stages. Microsoft's High Level Shading Language (HLSL) and NVIDIA's C for Graphics (Cg) [159] originated from a joint project of the two companies but have since then evolved separately. OpenGL Shading Language (GLSL) is OpenGL's built in high level shading language [163]. A drawback of HLSL is that it is only available on Microsoft's windows platform, compared to Cg and GLSL. Both are available for many different operating systems and architectures. Cg is the only shading language that can be used in conjunction with either OpenGL or DirectX.

In 2004, the GPU program length limitation, the major remaining limitation of Shader Model 2.x, was removed with the introduction of Shader Model 3.0 and DirectX 9 compatible GPUs, e.g. NVIDIA Geforce 6 and 7 GPUs and ATI's Radeon x1000 GPUs. Figure 3.1) depicts a diagram of the NV40/Geforce 6 series architecture. Special hardware units are responsible for vertex and fragment processing. Also note that difference in number of processing units for fragment and vertex stage. As fragment operations are in general more expensive to compute and many games became fragment-processing power bound, GPU vendors increased the number of fragment processing units more and more.

Shader model 3.0 introduced many more improvements, e.g. vertex stage texture fetch, improved floating point texture format support, efficient transform and feedback mechanisms for geometry instancing on the GPU, and maybe most importantly improved branching and dynamic flow control. The removed program length limit and improved dynamic branching support allowed the implementation of more complex operations in one GPU program in a single render pass, compared to previous multi-pass implementations. One such example related to the scope of this work, is GPU ray-casting that really took of with the introduction of Shader Model 3.0 hardware. Early GPU ray-casting implementations employed multiple render passes to generate the final image [106], later Shader Model 3.0 based implementations only use a single render pass to compute the volume rendering integral inside a loop in the fragment shader [195].

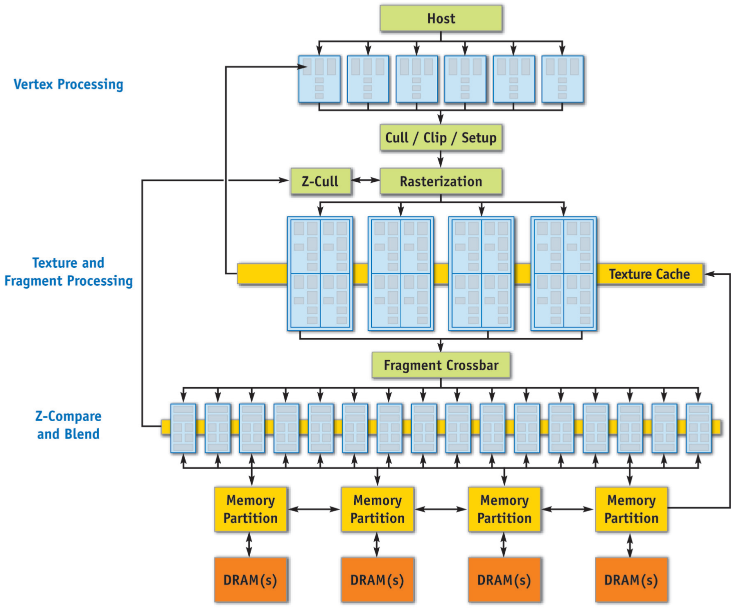


Figure 3.1: Diagram of the NV40/Geforce 6 series GPU architecture from 2004. Vertex and Fragment stages are separated and executed on special hardware shader units. Figure from [96].

GPGPU The improved programmability of GPUs and the potential high performance for data parallel computing applications also motivated several researchers to explore the potential of GPUs for accelerating general purpose computations, e.g. accelerating linear algebra [107].

The term GPGPU is used to describe this first efforts to utilize the GPU for accelerating general computations, see [65] and [148] for surveys. Due to the lack of parallel compute GPU APIs at that time, the computations had to be performed by implementing the algorithm using computer graphic APIs, e.g. OpenGL or Direct3D. To compute for instance the element-wise sum $\boldsymbol{C}$ of two $n \times m$ matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, one has to store the matrices in textures on the GPU and draw a quadrilateral, spanning an area of $n \times m$ pixels. The rasterizer then generates $n \times m$ fragment shader instances, one for every pixel in the covered viewport area. Each fragment shader instance computes the sum for one element of the result matrix $\boldsymbol{C}$ by fetching a value from the textures storing $\boldsymbol{A}$ and $\boldsymbol{B}$ and writes the result to the framebuffer.

Due to the use of computer graphic APIs, the computations are restricted by several limitations imposed by the programming model and APIs. Most importantly it is not possible to write the result of a computation to a specific location in GPU memory directly. The location of the result is pre-determined, so called scatter writes are not possible. Only by writing the result and its new location to a temporary render target and reading from it in a following render pass with displacement mapping in the vertex stage one can indirectly implement scatter writes on the GPU. Another, important limitation, especially for image processing algorithms, is the lack of a communicate scheme, e.g. shared memory, for a group of shader instances. Furthermore, although GPU programming languages allow simple user-defined types, e.g. structs in CG, the format for data input/output is limited to the implemented internal texture formats, namely one up to four element float vectors. Because of these limitations this first generation of GPGPU was limited mainly to academic research and GPU enthusiasts. However, several projects addressed this and provided abstraction interfaces for GPGPU-based computation backends, e.g. Brook [23], Sh [133], Glift [115].

Unified GPU Architectures The next big leap in the evolution of GPU architectures followed in 2005/2006 with the introduction of unified GPU architectures. The Xenos GPU used inside Microsoft's XBox 360 gaming console introduced the first unified architecture, followed shortly by NVIDIA's unified G80/Geforce 8 series (see figure 3.2), and ATI's Radeon HD 2000 series architectures.

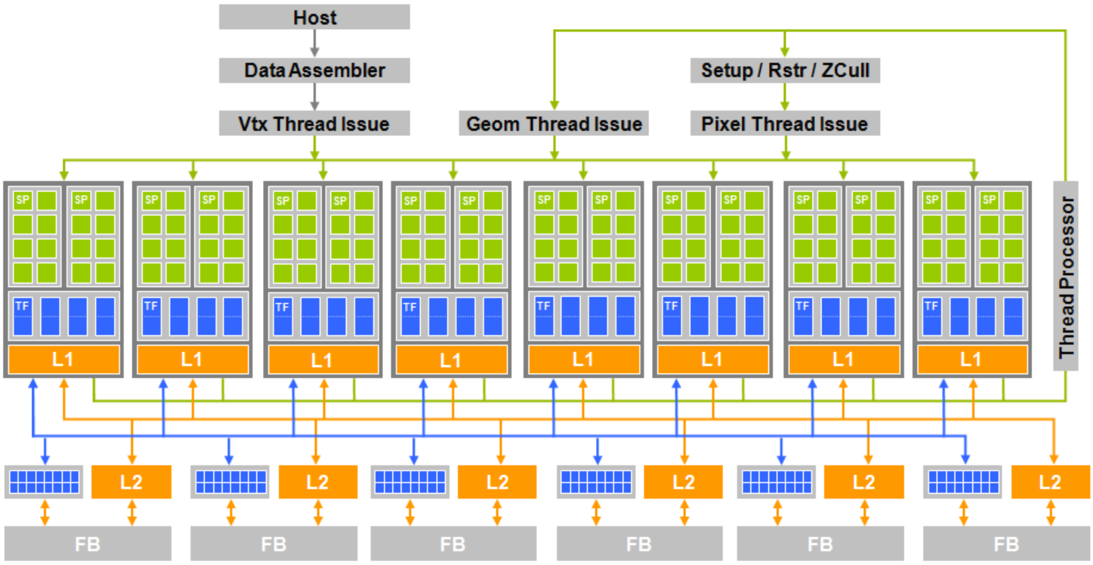


Figure 3.2: Unified G80, Geforce 8 series GPU architecture. Execution of vertex, geometry and fragment stage shaders is dispatched by schedulers to the same unified thread processors instead of dedicated ones as in previous GPU architectures.

The change from dedicated to unified hardware units was motivated by two main developments. For more and more applications it became increasingly complex to achieve high performance in all situations. For example in computer gaming, some parts of a game required high fragment processing power, while others required much more vertex processing resources. Thus, it became increasingly difficult to provide stable performance for any situation with the existing fixed number of per pipeline stage specialized hardware units.

Additionally the new version of Microsoft's DirectX 10 and Shader Model 4.0, introduced a new programmable geometry shader stage to the rendering pipeline situated between vertex and fragment stage. Sticking to the old scheme would have required designing yet another special hardware unit, furthermore increasing the problems of load balancing applications. As in shader model 4.0 the vertex, geometry and fragment shader stage implement the same feature and functionality set, a unified shader architecture was the obvious best solution. Furthermore, it solved many of the load balancing problems as shaders could now be scheduled on demand on the GPU hardware units.

Besides geometry shaders, Shader Model 4.0 introduced many more features, e.g. native integer and integer texture support, hardware accelerated interpolation of floating point format textures, bitwise operators, and largely increased number of registers that are also of importance for today's compute centric GPU APIs. The second generation of unified GPU architectures, NVIDIA's G200/Geforce 200 series and AMD/ATI's Radeon HD 4000 series, again improved the performance, and introduced most notably native support for double precision floating point numbers. The currently available high end single cards offer about 1 Teraflops (single precision peak arithmetic throughput) within a 200W power envelope.

GPU Computing The unified GPU architectures also enabled the development of a new class of GPU, parallel compute focused, programming APIs. The underlying scheme is to abstract the GPU hardware by a massively parallel Single Program Multiple Data processor. In 2006 NVIDIA introduced CUDA, an extension of the C programming language to allow the implementation of data parallel so called CUDA kernels on NVIDIA's G80 and all following architectures [119].

In 2009 the Open Computing Language (OpenCL) specification was published, after initial developments by Apple and following proposal to the Khronos Group. Similar to the OpenGL specification it specifies an open API that can be implemented by vendors on their specific hardware, and enhanced by hardware specific extensions. Today, OpenCL runtimes are available for multi (x86) and many core (GPU) architectures, freely available from NVIDIA, AMD/ATI or integrated into Apple's MacOS X operating system. Thanks to its open nature and platform independence, we will likely see OpenCL implementations on further hardware in the not so distant future, e.g. the Cell Broadband engine, Intel's Larrabee, or DSP chips. OpenCL's computation model itself and the API are closely related to NVIDIA's CUDA. For more details on the CUDA and the OpenCL programming model see section 3.2.2.

This concludes the brief history of GPU evolution up until today. At the end of this chapter we cover current and future trends. For more information on advanced rendering and computing concepts using GPUs we refer the interested reader to the GPU Gems book series [44, 141, 155] and the recent book on programming massively parallel architectures by Kirk *et al.* [99].

3.2 Today's GPU Hardware and Software

Today's unified GPU architectures offer programmers two main approaches to accelerate their applications by executing it on the many parallel stream processors. Both approaches abstract the GPU hardware, memory and compute resources, with an application specific model. Computer graphic APIs, e.g. OpenGL, Direct3D, use rendering pipelines to describe the control and data flows. Compute centric APIs employ the notion of multi-dimensional grids of blocks of threads, to facilitate an efficient transfer of data parallel algorithms to the GPU hardware. The following two paragraphs will briefly describe the GPU from a computer graphic and a high performance computing point of view.

3.2.1 From a Computer Graphics' point of view

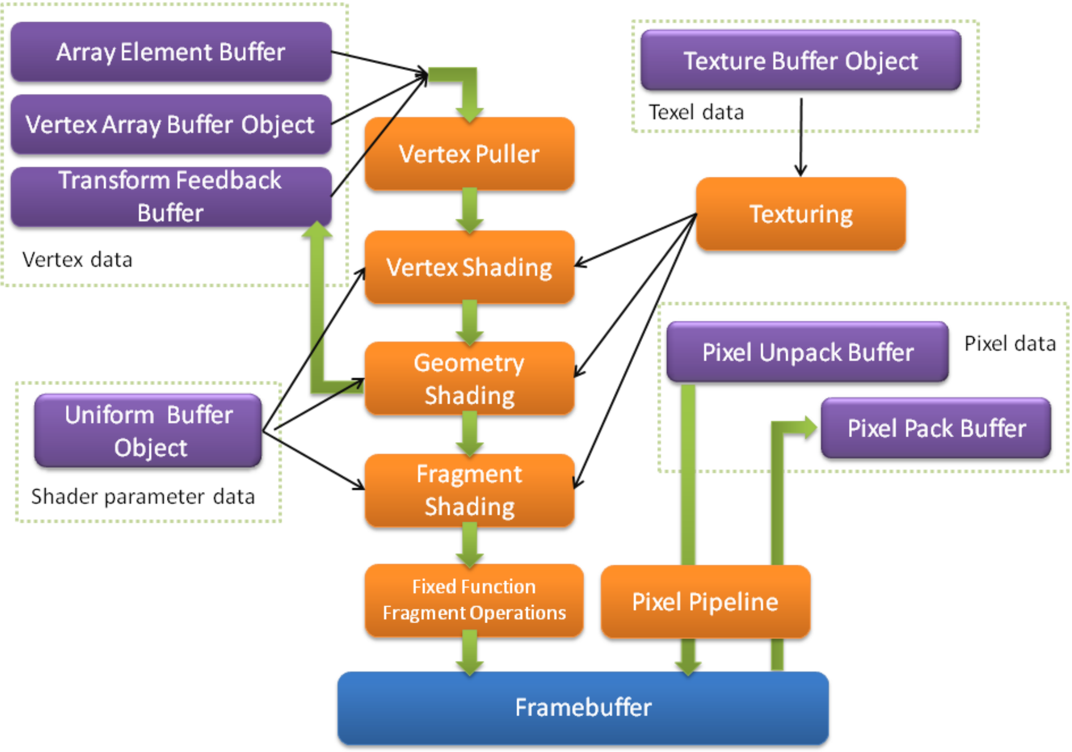


Figure 3.3: Diagram of data flow and stages in modern computer graphics pipelines. Geometry data is fetched by the vertex puller and passed into the vertex shading stage.

Pipeline Stages

- *Vertex Puller:* The vertex puller stages feeds the per-vertex data of geometric primitives, e.g. positions, normals, colors or texture coordinates, into the rendering pipeline and passes it on to the vertex shader stage.
- *Vertex Shading:* The first programmable stage in the rendering pipeline is the *Vertex Shading* stage. In general it performs transform and basic per-vertex lighting transformations, e.g. transforming vertex coordinates and normals from model to camera coordinate system, or project to the image plane. Nowadays, vertex shaders are not limited to traditional transform and lighting anymore. On modern unified

shader architectures the vertex shaders can perform any operation on the vertex data and read from up to 16 textures, also known as Vertex Texture Fetch. With this feature set, the vertex shading stage is also extremely useful for non-graphics related algorithms, e.g. histogram computation on the GPU using vertex texture fetch [177].

- *Geometry Shading:* The geometry shading stage was first introduced with Shader Model 4.0 and DirectX 10 by Microsoft and implemented in hardware by Geforce 8800, Radeon 2900 and later GPUs. This fully programmable pipeline stage operates on geometry primitive level, e.g. triangles, lines, and was introduced to allow programmers to change geometry primitive type and instance numbers. The main application is to stream one geometry primitive into this stage and let the geometry shader stream out multiple, altered version of the primitive, e.g. increase polygon model tessellation, create multiple instances from one geometry primitive. Currently, the stream out is limited to 1024 floats per geometry shader instance, if only vertex data is streamed out this allows up to 256 vertices (each with four floats encoding position in homogeneous coordinates). The geometry shader has full access to all information of every vertex of the primitive, e.g. position, normals, and additionally can read from multiple textures to fetch additional values for calculations. Furthermore, the results of the geometry shading stage can be streamed out to so called "Transform and Feedback Buffers" on the GPU and be reused in subsequent rendering passes.

- *Rasterizer:* The Rasterizer stages takes care of transforming the continuous geometry information into discretized per-fragment information. For every triangle the rasterizer computes the screen coverage and fills the fragments between the triangle edges. All fragment values, e.g. depth value, texture coordinate, screen position, normal, are interpolated using the chosen interpolation method, e.g. flat, linear, and accessible for the subsequent fragment shading stage. Fragments failing the Depth Test are discarded and not evaluated anymore. Note, that explicit writes to the depth value in the subsequent fragment stage effectively disables the early Depth Test in the Rasterizer.

- *Fragment Shading:* The fragment shading stage is the last of the three programmable graphic pipeline stages. In this stage a fragment shader is executed for every fragment in the framebuffer region. It has access to all interpolated information from the previous stages, and random read access to multiple textures. Originally this stage took care of computing the final color and lighting values, e.g. lookup and combine values from textures or varying variables, on a per-fragment basis. Today, this stage can compute any kind of graphic or numerical result and write it to the framebuffer or an off-screen render target.

- *Fixed Function Fragment Operations:* In this stage the values from the fragment shading stages are combined with values already present in the framebuffer. An example is alpha-blending when rendering semi-transparent objects, or depth or stencil tests resolving fragment visibility. As the name implies the last stage of the rendering pipeline is yet not programmable, however future Shader Model revisions will likely incorporate a programmable blending/compositing stage.

Memory Structures In the recent years computer graphic APIs have been changing towards buffer-centric memory models. This changes has been facilitated by the ever increasing on-board memory of modern GPUs and the performance limiting aspects of host-to-GPU-transfer bandwidth and API call overhead. Nowadays, data is packed into large buffers and transferred in one piece from host to GPU, to maximize performance and reduce number of transfers. When changes to the buffer data is required, the buffers can be mapped into host address space and modified. The data buffers are categorized as described in the following listing.

- *Vertex Data:* This data category comprises vertex data buffers, e.g. position, normal or texture coordinate, index Buffers, and transform and feedback buffers. These buffers are used primarily for storing geometry data and passing it efficiently to the rendering pipeline.
- *Texture Data:* Textures store data, in general 1D, 2D, or 3D image data, for texturing. Textures provide interpolated access to the texel values, using hardware accelerated nearest, linear and mipmap interpolation. To improve the latency of memory access the data is rearranged by the GPU after being transferred from the host.
- *Shader Parameter Data:* Modern GPU programs feature a large number of uniform variables controlling the rendering outcome, e.g. local illumination and shading variables. To improve the performance of setting these variables, recently uniform buffers have been introduced. Instead of calling a function for the transfer of every single uniform, a block of uniform variables can be directly transfered by one call using the new uniform buffer objects.
- *Pixel Data:* Pixel buffer objects store pixel information either for draw (when transferring data to the framebuffer or uploading to textures) or read operations (reading from textures or render-able buffers). Data transfers using pixel buffer objects are in general faster then using buffers residing in host memory, as the GPU driver can in general skip at least one memcpy.

3.2.2 From a HPC co-processor point of view

Today, NVIDIA's CUDA and the open, platform independent OpenCL are the major APIs and environments for implementing high performance, parallel compute application on GPUs. OpenCL is very similar to CUDA, mainly different names for compute primitives and memory types, therefore we focus only on CUDA in the following chapter.

The Cuda models abstract the GPU hardware as *a scalable array of multi-threaded Streaming Multiprocessors (SM)*, see figure 3.4. In the G80 architecture, each SM is made up of cluster of eight single precision MAD pipelines (SP), two transcendental units

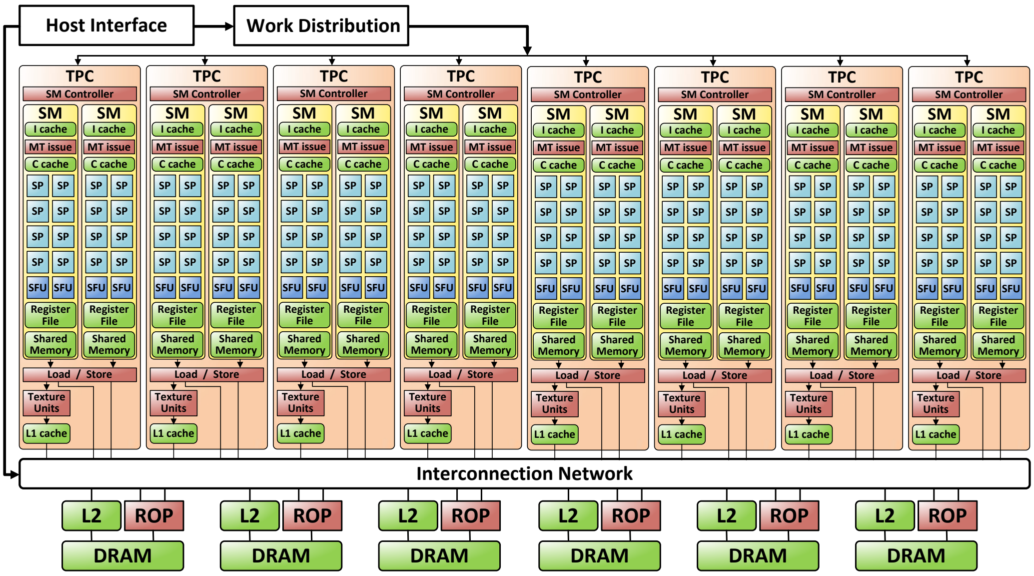


Figure 3.4: Diagram of the compute centric view onto NVIDIA's G80 architecture used int CUDA. Image courtesy of Philipp Stefan.

(SFU), one multi-threaded instruction unit (MT) and 16Kbytes of local storage (shared memory). The more recent G200 architecture adds one double precision MAD pipeline (GPU) per SM. At runtime the hardware enables the execution of thousands of logical light-weight, scalar threads. It can manage up to 768 (G80) concurrent scalar threads with zero scheduling overhead. Each scalar thread has its own instruction counter and can follow its own code path, thanks to dynamic flow control. Barrier-based synchronization allows for fine-grained parallelism.

CUDA offers a more generic parallel programming model, and exposes the GPU compute and memory resources in a hierarchical way to the programmer. Logical compute resources are organized in multidimensional compute grids, each grid block containing a number logical threads. These logical threads are at runtime dispatched by the thread scheduler for execution on the actual GPU stream processors (SP). The grid/block/thread scheme allows the direct identification of every grid block and thread element based on unique numeric IDs inside CUDA kernels.

In CUDA several types of GPU memory, global, texture, constant, shared, and registers are made available to the programmer. The major differences between the different types of memory is the location they reside on the GPU, access from the compute grid, and memory access bandwidth and latency. Global, texture and constant memory resides in GPU DRAM and is available from every grid block and tread. Registers and shared memory are on-chip memory. Shared memory can only be addressed by threads within the same grid block, registers are allocated and addressable only per individual thread. Being implemented as on-chip memory, shared memory and registers are closest to the actual compute units and offer the highest performance. CUDA supports random access reads and writes (scatter) to global memory. However for optimal performance sequential memory address schemes are advised.

By exposing much more details of the actual hardware CUDA allows the programmer more freedom to fine tune its code to the GPU architecture compared to the GPGPU approach using computer graphic APIs.

3.2.3 Differences to modern CPUs

The major differences in modern CPU and GPU architectures originate from the different major application domains the architectures were developed for. CPU designs focus on accelerating the execution of sequential program code. Over the last years, CPUs have been enhanced by various improvements to accelerate execution, e.g. larger caches, out-of order execution, speculative execution and parallel execution of instructions. However, neither cache size nor enhanced control logic contribute considerably to compute peak performance. Peak performance is mainly CPU clock rate and memory bandwidth limited. However, both clockrate and memory bandwidth have not been increasing tremendously in the last years. Recent CPU designs introduced multi-core and hyper-threading technology to improve program execution.

GPU designs have originated from solving the high throughput and parallel computation challenges posed by 3D computer graphic problems. Most operations performed in rendering pipelines are inherently data parallel, e.g. transforming vertices, shading fragments, writes to the framebuffer. Therefore, GPU designs have always been focused on maximizing the execution throughput of large number of threads running concurrently on the GPU hardware. In contrast to CPUs, today's GPUs many-core architectures are based on much simpler core processing unit building blocks. However, GPUs feature a large number of these units and devote only few chip area to control logic or cache structures compared to CPUs. The majority of the chip area is dedicated to the processing units. Furthermore, the memory bandwidth of GPUs is approximately 10 times that of CPUs of a similar vintage. Besides differences in hardware this also is partially rooted in the simpler memory models implemented by GPUs, due to less required flexibility and no legacy support compared to memory models implemented on CPUs.

3.3 Parallel Programming Concepts on GPUs

In the following paragraphs we briefly want to discuss the major differences in writing code for traditional CPUs and GPUs. We will discuss this for the example of reducing a set of N numbers stored in an array to a single value by summing up all array elements. Listing 3.1 depicts such an accumulate function written in standard C++.

```
float accumulate(float * array, int N)
{
  float sum = 0.0f;
  for(for int i = 0; i < N; ++i)
    sum+= array[i];
  return sum;
}
```

Listing 3.1: Sequential C code accumulating the first N values from an array.

Reduction operations (e.g. using binary operations +, -, min, max) are one of the fundamental classes of algorithms encountered for many applications and are very well suited as examples for demonstrating the differences of sequential and parallel implementations. When transforming the code from sequential, single threaded execution to multi or many thread execution one has to keep in mind and address the specific environment. For optimal performance one would like to equally distribute the amount of computations among the worker threads and minimize the amount of synchronization between threads.

Considering the simple case of going from a single-threaded execution of the code given in listing 3.1 to one using two worker threads, the computations could be shared by having the first thread computing the sum for the range $[0, \ldots, \lfloor N/2 \rfloor\,[$ and the second thread computing the sum for the range $[\lfloor N/2 \rfloor, \ldots N[$. To avoid synchronization issues when updating sum, both threads would ideally store the subrange accumulation result in thread local variables sum_{T1} and sum_{T2}. Once both threads have finished accumulating the values over the assigned array ranges, sum is computed by $sum := sum_{T1} + sum_{T2}$ by one of the two worker threads only.

However, this approach implicitly requires the programmer to partition the work among the available threads, allocate and manage thread local storage variables, and make sure the threads are synchronized before sum is computed. Furthermore, it requires much more code than just the one for the accumulate function, namely setting up, managing and terminating the threads. Ideally, programmers would like to have an environment that allows them to transparently use parallel computing resources, focusing on the actual code, without having to worry about setting up the environment themselves.

OpenMP is an open industry standard API [29] for accelerating C, C++ and Fortran code in shared memory multi-processing environments. OpenMP allows programmers to improve performance of their code by adding simple directives for OpenMP enabled compilers to benefit from available parallel computing resources. At program runtime the OpenMP runtime takes care of allocating a pool of worker threads, distributing and assigning the work to individual threads, as well as cleaning up on program termination.

The major benefit of OpenMP is that existing code can be modified relatively easy to benefit from parallel execution. Listing 3.2 depicts this for the accumulate example. Just by adding the pragma directive in line 4 before the for loop the reduction operation is executed by multiple threads at runtime.

```
1   float accumulate(float * array, int N)
2   {
3     float sum = 0.0f;
4     #pragma omp parallel for reduction(+: sum)
5     for(for int i = 0; i < N; ++i)
6       sum+= array[i];
7     return sum;
8   }
```

Listing 3.2: Parallel reduce using OpenMP

By adding `reduction(+:` sum) at the end of the pragma directive the programmer lets the OpenMP runtime know that every thread works on a local variable which finally is reduced using the + - operator into a global shared variable.

The major benefit of OpenMP is that the same code can be used for serial and parallel applications. Compilers not supporting OpenMP simply treat the OpenMP commands as comments. Furthermore, one can enable parallelism only for selected parts of the code. For the performance of the parallel code parts of a program one can expect a N times speedup when executing the program on a N processor system. However, OpenMP is only available for CPUs and not for GPUs. To benefit from the massive parallel architectures of modern GPUs one has use GPU specific APIs.

Parallel Reduction on the GPU

Currently there exist two approaches for implementing parallel reduction GPUs. The first one is to use computer graphic APIs, e.g. OpenGL or Direct3D, and employ a combination of ping-pong render-to-texture and shaders for the reduction on the GPU. The second approach is to use OpenCL or CUDA to implement parallel reduction using a more C like programming environment. In the following paragraphs we will discuss both approaches, for OpenGL and Cuda implementation examples, as the computer graphic based approach is still of importance today for many applications. The additional overhead of setting up, managing a CUDA or OpenCL context, and transferring or sharing memory between OpenGL/Direct3D and CUDA/OpenCL is not to be neglected. For many graphic applications it is easier to implement some lightweight computations in OpenGL or Direct3D directly and only outsource heavy processing tasks to parallel compute dedicated APIs, such as CUDA or OpenCL, on the GPU.

OpenGL-based Parallel Reduction

For parallel reduction of N values to a single one in OpenGL a combination of GPU (GLSL) code and host (OpenGL, C/C++) code to drive the reduction is used. The underlying scheme is to use a ping-pong rendering technique, using two *read-write* textures that are used alternately as *read-from* and *write-to* texture per rendering pass. On initialization both textures are resized to $k \times l$, and the N values are transfered to the *read-from* texture, with $k \times l \geq N$. The reduction value is computed by a series of rendering passes. In every rendering pass a $(k/s^i) \times (l/s^i)$ pixel spanning quadrilateral, is rendered over the $k \times l$ viewport, where $i \in [1, \ldots, \mathrm{N}]$ is the number of the current render pass. For every generated fragment a GLSL shader computes the reduction result for a $s \times s$ region of values from the current *read-from* texture and writes it to a texel in the current *write-to* texture. Before the next rendering pass *read-from* and *write-to* texture IDs are swapped. The rendering of quadrilaterals continues as long as the current quad resolution in pixels per dimension is a multiple of s. Otherwise, there exists no exact pixel to $s \times s$ region mapping between the current and in the previous pass rendered quadrilaterals.

If the 2D reduction did not complete entirely on the GPU, the remaining values are read back from the GPU to the CPU and reduced on the host side. Alternatively, reductions either only along the first or second texture coordinate axis can be implemented to reduce the remaining values. However, this often results in a much smaller number of active fragments per rendering pass, or hardware threads on the GPU, and increased

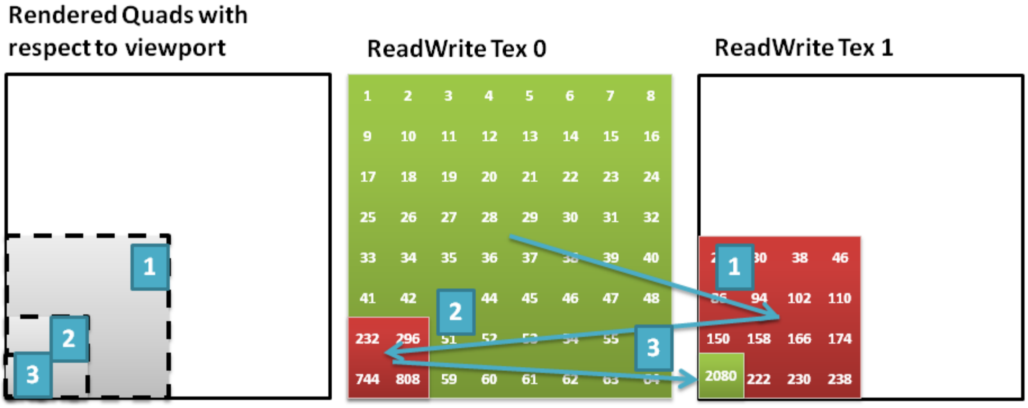


Figure 3.5: Reduction scheme using ping-pong render to texture in OpenGL, demonstrated for computing the sum of the first 64 natural numbers stored in a 8x8 texture. In every rendering pass a quadrilateral spanning a quarter of the previous viewport area is rendered. A GLSL program computes for every fragment in the current *write-to* texture the sum of 4 values, fetched from a 2x2 region in the current *read-from* texture. After three iterations the final result 2080 is obtained in this example.

number of rendering passes compared to reduction along both axes. Therefore, frequently $s = 2$ is chosen and k and l are replaced with their next greater or equal multiples of two. Thereby, each fragment will reduce 2×2 values per pass and the final value is obtained in $log_2(max(k, l))$ rendering passes entirely on the GPU. Figure 3.5 depicts the rendering scheme for $l = 2$ applied for the example of computing the sum of the first 64 natural numbers stored in a 8×8 texture.

Parallel Reduction in C for CUDA

In contrast to the OpenGL version depicted in the previous paragraph, C for CUDA does not require to wrap computations by rendering operations and provides the programmer with a more familiar, C-like programming environment. Data is directly available in arrays instead of textures and the programmer has access to the individual threads on the GPU hardware and shared memory to improve performance compared to the OpenGL version.

Compared to the OpenGL/GLSL reduction the CUDA kernel benefits from the use of shared memory per grid block to accelerate the computation on the GPU. The kernel in listing 3.3 can be broken down into three main parts. In the first phase values from global memory are read into shared memory and the first level of reduction is already performed (lines 9 – 12). Each thread reads in two values from global memory, with sequential addressing to improve performance. In the second phase the reduce operation is performed entirely in shared memory (lines 16 – 21), followed by the third phase, write result for the current block to global memory (line 23). Thread synchronization is used during to guarantee all threads have finished the reduction for the current level before proceeding to the next one. This separation of computing phases into read input values from global memory, compute result from local storage, and write result to global

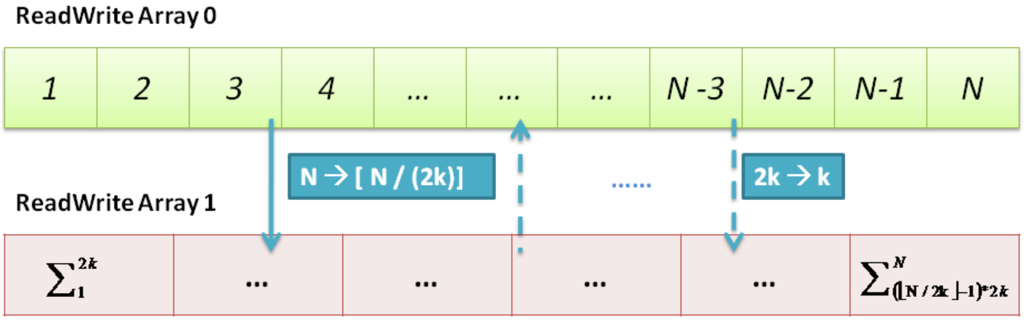


Figure 3.6: Parallel reduction using CUDA. Instead of having to use ping-pong render to texture as for OpenGL, CUDA directly allows addressing arrays in memory. In each iteration $2k$ elements of the current *read-from* array are reduced to a single one in the current *write-to* array by k threads per block of the compute grid.

memory facilitates an improved parallelization and GPU resource usage when running a huge number of thread blocks on the GPU.

Figure 3.6 depicts the scheme for parallel reduction of an array with N elements used in cuda. In every iteration, a computation grid of $|b = \lceil N/(2k) \rceil$ blocks, each with k threads is setup to reduce the data by a factor of $2k$ and write the result to another array in GPU memory by the kernel given in listing 3.3. The kernel source code is taken from the CUDA SDK parallel reduction example. In total $i = log_{2k}(N)$ iterations are needed to compute the final result.

```
template <class T>
__global__ void reduce(T *g_idata, T *g_odata, unsigned int N)
{
    SharedMemory<T> smem;
    T *sdata = smem.getPointer();
    unsigned int tid = threadIdx.x;
    unsigned int i = blockIdx.x*(blockDim.x*2) + threadIdx.x;
    // read from global to shared mem
    sdata[tid] = (i < N) ? g_idata[i] : 0;
    if (i + blockDim.x < N)
        sdata[tid] += g_idata[i+blockDim.x];

    __syncthreads();

    // do reduction in shared mem
    for(unsigned int s=blockDim.x/2; s>0; s>>=1)
    {
        if (tid < s)
            sdata[tid] += sdata[tid + s];
        __syncthreads();
    }
    // write result for this block to global mem
    if (tid == 0) g_odata[blockIdx.x] = sdata[0];
}
```

Listing 3.3: C for CUDA reduction kernel source code. From CUDA SDK.

3.4 Future Trends in (GP)GPU Development

The development of GPUs will continue in the coming years. Currently NVIDIA and ATI/AMD have announced or already started to ship their most recent architectures, namely the NVIDIA Fermi architecture and ATI/AMD Radeon HD 5000 series. Compared to the previous generation the number of stream processors has again doubled, pushing the peak performance of the high end, single chip cards beyond 2 Teraflops. Both GPU architectures implement the most recent Shader Model version 5.0, which introduces Microsoft's DirectCompute shading stage to the Direct3D rendering pipeline for general purpose computation on DirectX 10 and DirectX 11 for Windows Vista and 7 systems, sharing many of the concepts with the already existing CUDA and OpenCL compute languages.

NVIDIA's Fermi introduces a set of unique features and architectural improvements, high performance computing users have been requesting and waiting for. It is the first GPU to support more than 4 Gbytes of memory by providing a CPU-like address space, 64-bit virtual and at least 40-bit physical address space. Furthermore, it will introduce a unified device memory space, and support for pointers to GPU memory. Further improvements include improved device memory access speed, larger cache sizes and increased shared memory, more efficient flow-control, higher double precision performance, support for real function calls in computation kernels, exception handling, simultaneous execution of multiple different kernels, and support for kernel execution interruption. Thereby, Fermi based GPUs will improve the performance of many existing compute kernels, and extend and improve the use of GPUs into many more application domains due to the improved execution model and programmability.

Intel's Larrabee (GP)GPU chip architecture [180] is another interesting concept for high-performance computing on GPUs. Larrabee's unique, hybrid architecture combines many features of current CPUs and GPUs on one chip. The biggest benefit is definitely the native support for existing x86 code, thanks to its x86 based architecture. Parallel x86 code, e.g. using OpenMP or PThreads, will directly benefit from execution on Larrabee's many-core architecture. It remains to be seen how well existing computer graphics and GPGPU applications will perform on Larrabee. Furthermore, Intel is working on C for Throughput Computing (Ct), yet another language for parallel high performance computing. Larrabee was initially scheduled to be released as a product in 2010, but due to project delays and performance issues the release was canceled. Instead, access to development samples and SDKs was given to selected strategic academic and industrial partners.

The current trend in GPU evolution will continue, increasing the number of processors and performance with every generation. Additionally, more advanced features will be added to the still relatively simple core GPU processors, as demonstrated by the changes introduced with NVIDIA's Fermi architecture. Today, high performance computing based features are developed concurrently with computer graphics features and integrated into one single architecture. It is only a question of time, when computing and computer graphics features will be competing for the limited space on the chip. When this happens, it might result in the end of the unified GPU architecture and split into two architecture lines, one for high performance computing only and one for computer graphics.

However, there is another ongoing development, which will eventually have a great impact on high performance computing in general. There is a strong trend to mobile devices, e.g. smart phones, tablet PCs, netbooks, and thin clients, with very low computational power compared to current existing desktop and laptop systems. Concurrently, hardware and software solutions are developed for efficiently streaming high definition image, video or gaming content onto these devices, or facilitate remote applications, by sharing computing resources over the Internet. More and more computations will be moved to data and compute centers, where server farms equipped with large numbers of CPUs and GPUs e.g. run multiple instances of a computer game that are streamed over the Internet to thin-clients, run a heavy-compute intense numerical simulation, or act as application server for a series of medical workstation. Scalability of their architecture design from low power devices to high performance compute systems is one of the major challenges the chip manufactures are facing in the future.

CHAPTER 4

GPU-accelerated Volume Visualization

Volume Visualization is an important key technology for understanding, visual exploration and interpretation of 3D data in both the academic and industrial domain. In contrast to traditional, geometry based computer graphics, volume visualization deals with visualizing three-dimensional scalar data. Originally volume visualization methods were mostly motivated by scientific visualization problems, mainly the visualization of 3D scalar data acquired by scanners, e.g. 3D medical imaging modalities, or generated by simulations or computations in science. But, in recent years volume visualization techniques have also been adopted by the movie and computer gaming industry, for generating realistic special effects, e.g. fluid, fire or star simulations.

Today, one of the largest application domains of DVR is visualization of 3D and 4D data acquired by various modern imaging technologies, e.g. CT, PET, MRI and US. Medical image data visualization is a challenging field itself, as techniques developed for one modality often cannot be straightforward transferred and applied to other modalities. Furthermore, the data properties and thus visualization requirements are largely varying for different modalities, e.g. 4D US visualization requires real-time streaming and display of many small 3D volumes and compensation techniques for low US quality. On the other hand, visualization of large, high-resolution CT scans require sophisticated memory management techniques to fit the volume data into the limited on-board GPU memory. A common requirement for visualization of any medical data is that the visualization algorithm aids in visualizing the anatomical region or focus of interest and removal of non-interesting areas. This can be accomplished by various techniques, e.g. clipping, classification, importance driven, or focus and context rendering. The field of medical image data visualization is huge and beyond the scope of this chapter. Thus, in this chapter we focus on discussing the basic and background theory of DVR, a detailed presentation of state-of-the-art DVR using GPU ray casting, and conclude with an overview on selected advanced rendering techniques for 4D, stereoscopic volume rendering, focus and context visualization, and fused rendering of multiple, multi-modal volumes.

4.1 Volume Rendering Background

DVR is the generic term for direct visualization of volume data without prior extraction of geometry, e.g. by segmentation and surface reconstruction, from the volume. Figure 4.1 depicts three exemplary visualizations of the same CT dataset by direct volume rendering techniques.

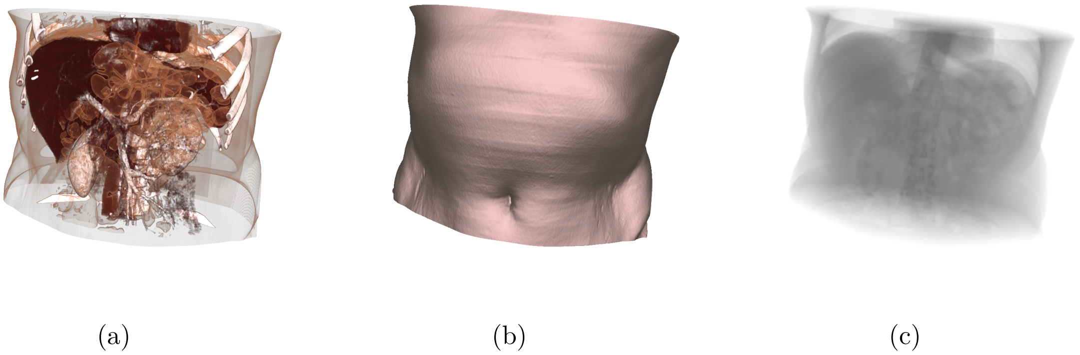


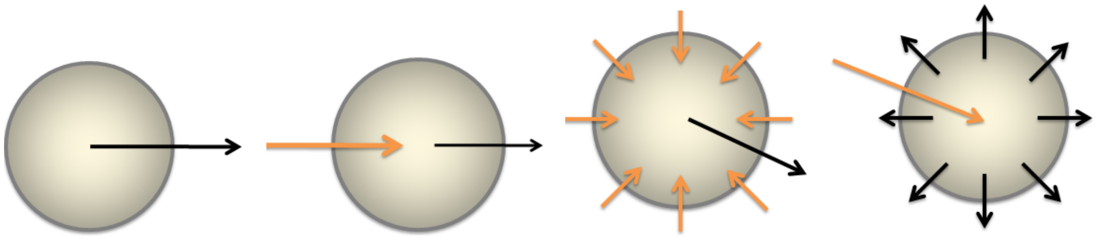
Figure 4.1: (a-c) Example images to illustrate various different final images that can be generated by different rendering techniques. The same dataset is rendered from the same perspective with three different rendering models, (a) Emission-absorption Volume Rendering with Local Illumination, (b) Shaded non-polygonal isosurface rendering, and (c) simulated X-ray from CT also known as Digitally Reconstructed Radiograph (DRR).

From the images in figure 4.1 it becomes clear that the information extracted from the volume data is heavily dependent on the light transport model employed when generating the images. Figure 4.1(a) uses an emission-absorption model, figure 4.1(b) uses non-polygonal isosurface rendering, and figure 4.1(c) uses a pure emission based light transport model. The emission-absorption model is one of the most widely used light transport model in volume rendering and will be explained in detail in the following section.

Non-polygonal isosurfaces are generated by first-hit search in camera to volume direction and are frequently shaded using traditional surface, local illumination based shading models. They are an alternative to indirect rendering of isosurfaces extracted by segmentation and surface reconstruction, when one is only interested in visualizing the isosurface and not storing the actual geometry.

4.1.1 Light Transfer Model

To generate images from the volume data one first needs a model how light rays propagating through the volume are interacting with the volume, or medium. In general there are four different possible interactions that can occur (see figure 4.2).



(a) Emission (b) Absorption (c) In-Scattering (d) Out-Scattering

Figure 4.2: The four different possible interactions that can occur between light modeled by rays in participating media: (a) Pure emission: Light is emitted by the medium itself, (b) pure absorption: incoming light is completely absorbed, (c) in-scattering: multiple incoming light rays are combined, and (d) out-scattering: a single incoming light ray is scattered into multiple rays

The first mode is pure emission (see figure 4.2(a)), one can also imagine the volume data emitting light, radiative energy, actively by itself. The second kind of interaction that can occur is absorption of incident light (see figure 4.2(b)). Thus the radiative energy is attenuated during traversal along the propagation direction. The third and fourth type of interaction involve scattering (see figure 4.2(c) and figure 4.2(d)). Either in or out scattering. Both potentially change the major direction of light propagation. In-scattering occurs when one or more incoming light rays a re bundled into a new ray with a new direction and combined radiant energy. This may also involve a change of the wavelength. We refer to out-scattering as the process when one incoming light ray is scattered into multiple rays, each with a new propagation direction and radiate energy.

4.1.2 Volume Rendering Integral

The basic idea of all DVR algorithms is to trace light rays through the volume into the virtual camera and compute the final radiative energy when the ray hits the image plane, by sampling the volume along the propagation direction and integrating the radiative energy based on the sampled value. To that end various light transport models of varying computational complexity have been developed, incorporating one or more of the possible light/media interactions described in the previous paragraph.

The emission-absorption model is one of the most frequently employed light transport models for DVR. The volume can both emit light by itself and absorb incoming light along the light ray direction. However, scattering and indirect illumination effects are neglected. In comparison to light transport models integrating scattering effects, the emission-absorption model is of less computational complexity as only light interaction in ray direction are considered.

For an emission absorption model the volume rendering integral along a ray propagating through the volume is defined by

$$I(s) = I(s_0)e^{-\tau(s_0,s)} + \int_{s_0}^{s} q(\tilde{s})e^{-\tau(\tilde{s},s)}d\tilde{s} \tag{4.1}$$

where $I(s)$ is the received radiant energy at point s, $I(s_0)$ is the initial radiant energy at point s_0, $e^{-\tau(s_0,s)}$ is the absorption along the ray segment $s-s_0$, $q(\tilde{s})$ the radiant energy emitted at point $\tilde{s}$ along the viewing ray and $e^{-\tau(\tilde{s},s)}$ the absorption between $\tilde{s}$ and s.

For the actual computation a discretized, numerical approximation of the analytical volume rendering integral is used. The integral is computed iteratively either in front-to-back or back-to-front direction along the viewing ray. The current integral value is computed by composition of the value at the current sample location and the so far computed integral.

$$I(D) = \sum_{i=0}^{n} c_i \prod_{j=i+1}^{n} T_j, \text{ with } c_0 = I(s_0) \tag{4.2}$$

The summations and multiplications in equation 4.1.2 can be reformulated into a set of sequentially executed compositing operations. For compositing in front-to-back direction (from camera to volume) along the ray the compositing equations are given by:

$$C_{dst} = C_{dst} + (1 - \alpha_{dst}) \cdot C_{src} \tag{4.3}$$

$$\alpha_{dst} = \alpha_{dst} + (1 - \alpha_{dst}) \cdot \alpha_{src} \tag{4.4}$$

where C_{dst} denotes the currently accumulated color value and C_{src} the currently sampled emissive color component. Likewise $alpha_{dst}$ denotes the accumulated opacity and α_{src} the opacity value for the current volume sample. C_{dst} and α_{dst} are initialized with 0 and equation 4.1.2 is repeatedly evaluated when marching from the eye along a viewing ray through the volume. An interesting property of the front-to-back compositing scheme is that compositing can be stopped when $alpha_{dst}$ becomes one as further samples will not contribute anymore to the color and opacity components on the left sides of equation 4.1.2.

Besides the presented compositing schemes, Maximum Intensity Projection (MIP) is another widely used scheme to compute the final color value in the image plane. It simply computes the maximum color value of all color samples along the sample ray.

$$C_{dst} = max(C_{dst}, C_{src}) \tag{4.5}$$

In contrast to emission-absorption related light transport models or non-polygonal isosurface rendering, the computation C_{dst} for MIP is independent of the volume traversal order. It can be computed in any order. This applies likewise to average or summation based compositing schemes, e.g. DRR.

4.1.3 Rendering Pipeline

Despite the large number of different existing methods for direct volume rendering, a core pipeline exist that is common to them (see figure 4.3). This pipeline consists of six stages, namely: volume traversal, interpolation, gradient, classification, shading and illumination, and compositing (all stages are described in the following paragraphs). The four stages, interpolation, gradient, classification, and shading and illumination are all executed only locally for every sample location. They are neither dependent on the order of volume traversal nor on the compositing scheme. Thus theses stages can easily be reused and shared among different volume rendering algorithms. Also not all of these four stages are used in every rendering method, any stage can be skipped if the following stages are independent on the output produced by the specific stage. In contrast compositing is totally dependent on the employed volume traversal scheme and direction, e.g. front-to-back or back-to-front.

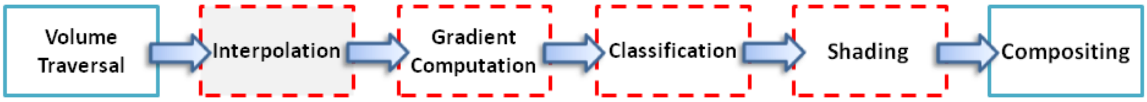


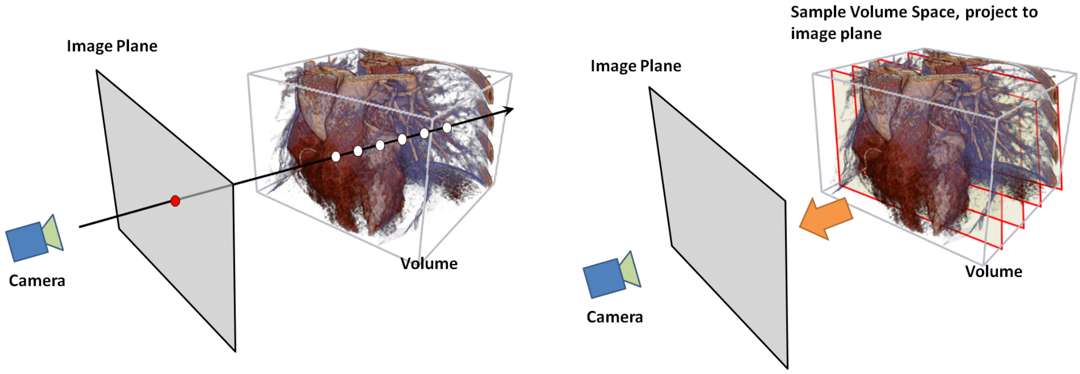
Figure 4.3: Generalized direct volume rendering pipeline. Stages that are executed per sample location and might be shared between different algorithms are highlighted with a red, dashed boundary.

- *Volume Traversal:* This stage defines the specific volume sampling approach. The only condition for the sampling order is to ensure that the samples can be composited correctly in either front-to-back or back-to-front direction. The different volume traversal methods can be classified into image-order and object-order approaches.

  The main and most widely known image-order approach is volume ray casting, originally proposed by Levoy *et al.* [118]. For every pixel in the image a ray is casted back into the object space. If the ray intersects the objects, the final pixel value is computed by marching and sampling the data along the ray-segment with equidistant steps. The main advantage of the algorithm is the high-quality of the rendered images. It mainly scales with the image resolution, and is mostly independent of the volume resolution. Simply resizing the image to a lower resolution often results in major performance boosts. The major disadvantage of the basic, unmodified ray casting algorithm is the rather large memory consumption, as the whole volume has to be stored in a regular grid, and the low performance because of the inability to skip non-visible parts of the volume. However, several optimizations exist that address these limitations, e.g. bricked volume storage, empty space skipping, adaptive sampling, and early ray termination. Most of these optimization will be discussed in detail in section 4.3.2. Because of its straightforward and easy implementation on modern GPUs and the great flexibility and adaptability of the core algorithm, ray casting has become the de facto standard for real-time, high-quality direct volume rendering.

  Object-order approaches work the other way around, starting traversal in object space and projecting the samples onto the image. A large number of algorithms has been developed, e.g. 2D and 3D Texture Slicing, Shear-Warp, Splatting (for more details we refer to [36]). Currently texture slicing methods are the most used object-order rendering techniques for GPU accelerated direct volume rendering. The implementation is straightforward and only requires hardware accelerated 2D or 3D texturing and blending support. The underlying method is to render a number of slices, intersecting the volume, in front-to-back or back-to-front order, interpolate samples from 2D or 3D textures, and blend the fragments into the framebuffer. For high quality images, floating point blending support is required.

In contrast to image-order approaches, object order approaches can exploit knowledge about the volume data, e.g. visible and non-visible part in order to directly reduce the number of samples and consumed volume memory. The restriction to store the data in a regular grid can be removed, as the traversal order is known in object space. However, on the other side all object-order traversal approaches have one main drawback: the footprint of the projected sample in the image. It is very hard to not over- or under-sample in the image plane. Zooming in or out, will for most methods quickly reveal specific projection artifacts.



(a) (b)

Figure 4.4: Major volume traversal schemes. (a) Image order approach. Traversal starts from a pixel in the image plane through the volume. (b) Object order approach. Traversal starts in object space and samples are projected onto image plane.

- *Interpolation* of the scalar value at the sampling position is an essential part of any volume rendering method. In general the sampling position is continuous and the sample value has to be determined through interpolation of values from the closest discrete grid positions. Nearest neighbor interpolation is the fastest interpolation, however also the one with the least quality. Linear interpolation provides a good balance between speed and quality for visualization in most cases. It is used in general, also because modern GPUs feature very efficient hardware accelerated linear interpolated texture access implemented in hardware. Alternatively, higher-order interpolation filters can be used by implementing the filter directly in the DVR fragment shader. In [191] a recursive third-order interpolation filter is presented. The filter weights and offsets are pre-computed and stored in a lookup table, requiring only a small number of additional texture lookups and calculations. On recent GPU hardware the filter weights and offset can be computed on the fly without a major performance drawback.

- *Gradient Estimation* - Depending on the volume rendering algorithm the gradient at the sample position has to be determined, e.g. as input for multi-dimensional classification or shading and local illumination. One can either use interpolated gradients from pre-computed gradient volumes or compute the gradient on the fly. The later is usually of better quality, but requires additional samples in the neighborhood of the sample position. At least three additional samples for forward and backward differences or six for centralized differences. A good compromise is to conditionally evaluate the gradient only if the contribution to the volume rendering integral at the current position exceeds a threshold.

- *Classification* denotes the determination of the appearance of the volume rendering. Usually lookup tables are used to assign a specific color (emission) and opacity (absorption) value for every scalar volume sample. In general one-dimensional (1D) lookup tables are used, assigning RGBA values to every scalar value. Kniss *et al.* [101, 102] propose multi-dimensional classification, using both the scalar and gradient magnitude values to determine the color and opacity for a sample. A major challenge in classification for DVR is the actual definition of the transfer function. Many works implement graphical user interfaces [97, 168] for interactively defining the transfer function by manipulating geometric primitives, e.g. ramps, ellipses, triangles, with respect to the histogram or joint intensity/gradient magnitude histogram of the dataset.

- *Shading* is frequently used for adding an illumination component to the emissive component of the current sample. The key benefit is that local illumination models do not increase the complexity of computing the volume rendering integral, only slightly reducing real-time performance mostly due to gradient calculation. In general local illumination models, e.g. Phong, Blinn-Phong or Cook-Torrance, are used to model single scattering similar to local surface rendering. Local illumination models assume that external light reaches unchanged every location in the volume, thus ignore global light/media interaction effects, e.g. scattering, attenuation. The volume gradient at the current sample location is used as the normal vector for the local illumination models. This is a valid assumption, as the volume gradient corresponds to the normal vector of the isosurface through the sample location in volume space.

  Local illumination is very well suited to improve the perception of volume-rendering by shading semitransparent volume surfaces. However, for shading large homogeneous regions they are inappropriate. The gradient in these regions is often undefined or very noisy resulting in a reduced image quality. Global illumination models that incorporate global phenomena, e.g. scattering [160], shadowing [59], ambient occlusion [123, 169] can be used instead. However, all these models increase the complexity of the volume rendering integral substantially, in most cases disabling real-time performance even on most recent hardware.

- *Compositing* is the last stage in the pipeline and computes the final pixel value from the individual (shaded) source samples. Different compositing schemes can be used to achieve different effects. Most used schemes are emission-absorption direct volume rendering DVR , first hit or also known as non-polygonal isosurface rendering, MIP, average intensity, or digitally reconstructed radiographs DRRs.

4.2 GPU-accelerated Volume Rendering

In the recent years, GPU-accelerated volume rendering has become feasible thanks to the development of programmable GPUs and high-level shading languages. For many applications GPU accelerated DVR has become the de facto standard. The earliest methods implemented 2D and 3D texture slicing algorithms, recently GPU-accelerated ray casting has become feasible and the standard algorithm on modern GPUs. In the following paragraphs we will discuss GPU-accelerated ray casting in detail. We start with the basic algorithm, explain integration of opaque scene geometry and how to implement clipping either procedurally, based on convex clipping geometries or on clip volumes. Then we continue with a detailed description of quality and performance modifications to the basic algorithm, and conclude with sections on advanced rendering techniques, e.g. focus and context visualization, real-time 4D rendering and fused rendering of multi-modal data.

4.2.1 GPU Ray Casting

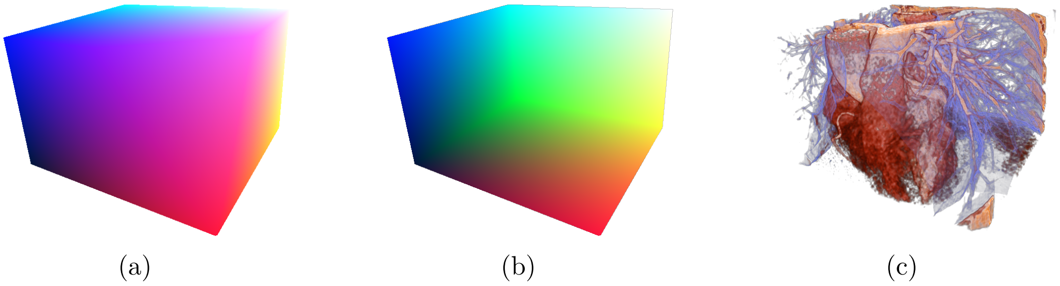

(a) (b) (c)

Figure 4.5: Color-coded volume texture coordinates by rendering (a) front and (b) back faces of volume bounding box. (c) Volume Rendering result.

Krüger *et al.* [106] presented one of the first GPU-accelerated ray casters. The basic idea of their algorithm is as simple as ingenious. Instead of computing the ray volume bounding box intersection per pixel in a shader or on the CPU, they let the GPU compute it by rendering the front and back faces of the volume proxy geometry (they used the volume bounding box) color coded to two RGBA textures (see figure 4.5). For every pixel one can directly read the entry and exit position in volume texture coordinates, each RGB triplet corresponds to a XYZ volume texture coordinate triplet. Due to at that time still restricted number of instructions in a GPU program, a multi-render pass technique was used to fully traverse the volume along the rays. In each render pass, a fragment shader reads for every active pixel the current start position along the ray and current integral value from textures. A fixed number of samples is performed and the updated sample position and integral value are written to separate textures. If ray traversal ended, a very small depth value is written to the depth buffer, essentially prohibiting further calculations for this pixels by the depth test of the render pipeline.

Single-Pass Ray Casting

With the advancement of GPU hardware, more complex and much longer programs became possible, enabling so called *single pass* ray casting algorithms [175, 195]. The whole traversal along the ray can be implemented inside a single loop of a single fragment shader. However, most GPU-accelerated ray casting volume renderers employ a combination of multiple render passes to compute the ray entry and exit position followed by a single ray casting pass. Although everything, including ray entry and exit computation and volume rendering, can be done in one single render pass, the multi-pass solution offers more flexibility to add features and leave room for both performance and quality improvements.

Optimizing Ray Setup - Integration of Opaque Scene Geometry and Clipping

For many applications the combination of both volume rendering and conventionally rendered opaque scene geometry is desired. This is frequently the case for medical applications, where volume data is combined with conventionally rendered meshes, e.g. results of segmentation algorithms, or virtual models of medical tools and instruments. Multiple cases are possible: (a) Occlusion of volume in view direction by opaque geometry: No volume rendering should be performed as the result does not contribute to the final pixel value. Ray start position has to be invalidated. (b) Geometry inside the volume: Volume rendering is performed until geometry surface is hit by ray. The ray exit position has to be updated with the geometry/ray intersection. In a final step blending of volume rendering image and geometry image has to be performed to yield the final image.

Another desirable feature, especially for virtual fly-through applications in the medical domain, is the correct intersection of the volume with the near clipping plane of the virtual camera. If not handled carefully an incorrect rendering is produced, as the near clip plane will simply clip the volume proxy geometry and the rays will either be not started at all or at a wrong start position.

The last feature that is often requested is support for closed convex clipping geometries. Clipping is one of the most basic, yet most intuitive methods to provide insight into a dataset. The volume is either rendered behind the clipping geometry, only inside the clipping geometry, or parts of the volume are skipped with respect to the geometry.Clipping can be implemented in different ways. One way is similar to the interaction with solid geometry objects. Another way is clipping based on voxelized geometries or by clipping volumes, e.g. obtained from a segmentation of the data. The last method is to procedurally clip against a clipping plane/object that can be described by a simple formula (e.g. plane, sphere, box) in the ray casting shader.

Related works employ a series of additional render passes to update the ray start and exit positions according to near plane intersection, opaque geometry intersection and clipping geometries. Scharsach *et al.* [175] for instance, first render the RGB values of the back-projected near plane into the ray start RGBA render target, followed by an only alpha-write enabled pass of the closest back faces. Every generated fragment writes 1.0 to the alpha channel. If not a valid front face fragment is generated later on by the nearest front faces pass, the near plane position is used as ray start position. In the third and final pass the nearest front faces are rendered with RGBA writes enabled. The alpha channel is used as a flag to denote if a ray shall be started or not by the following ray casting pass. To correctly combine the opaque scene geometry and volume rendering, one needs to know the position of the scene geometry with respect to the volume. To that end the geometry is rendered after the respective front and back faces passes. For the modification of the ray start position the depth test is set to *less*, and for every fragment of the geometry passing the test, meaning it is closer to the camera than the volume, the ray start position is invalidated. The same needs to be done for the ray exit positions. After rendering the furthest away back faces, the depth test is reversed from *greater* to *less* and the geometry is rendered inside the volume with color coded positions. For restricting the rendering to the inside of a closed convex geometry the updates are performed in similar way exploiting the GPU depth test.

The drawback of this method is the repeated rendering of the scene geometry to update the ray start and end position. In contrast our modified method works with rendering scene and clipping geometry only once. To that end the scene geometry is rendered once before volume rendering into an off-screen render target. Both color and depth buffers of this pass are stored in separate textures. The same way the closest clipping geometry front and farthest back faces are rendered to two separate depth textures. Compared to [175], the rendering of the front and back faces of the volume proxy geometry is modified. We do not directly render the coordinates to RGBA textures, but only render the depth values for the closest front and back faces into two separate textures. To support correct clipping the near clip plane the front face are rendered in two passes. The first pass renders the closest back faces of the proxy geometry, for every fragment the depth value in the render target is set to the near plane depth value. In a second pass we render the closest front faces to the render target. Fragments that were not clipped will update the texture with the correct depth values for the closest ray/proxy geometry intersection per pixel. For the clipped fragments the near plane value from the first pass is valid. In the following render pass the depth values from all intermediate textures (ray start and exit, scene geometry and clipping geometry) are read by a fragment shader, sorted from smallest to largest value and by type (scene geometry, clipping geometry, proxy geometry start and end), and the final ray start and exit positions are computed by back-projecting the start and end depth values into volume space. The benefits of our approach are that the number of render passes is reduced. Furthermore, the method lends itself for lazy updates of the respective textures. Only with respect to the last frame changed geometry needs to be re-rendered to re-compute the ray start and exit positions. Although at first glance, the reduced number of render passes might not look like much, but for more complex proxy geometries, e.g. octtrees for empty space skipping, each render pass is reducing the overall frame rate. Imagine rendering a simple instrument moving through the volume and having to re-render a complex proxy geometry every frame! Additionally with a simple extension, namely comparing the newly generated ray start and exit values with the last ones, skip ray casting for every non changed ray.

4.3 Optimizing Speed and Quality

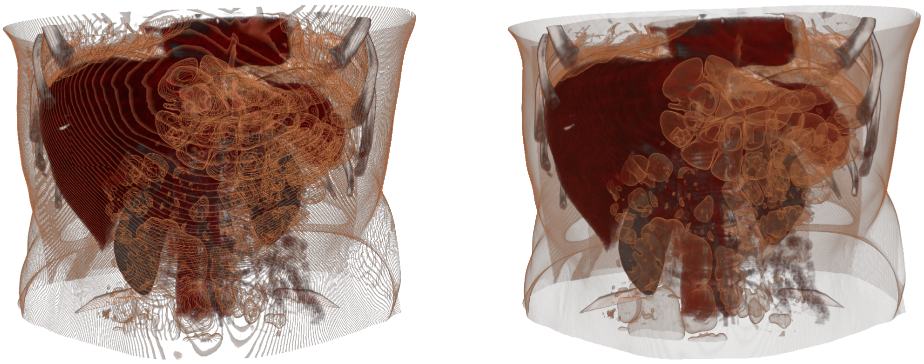
The methods described in the proceeding sections have covered the theoretical foundations of volume rendering and described the principal methods for GPU-accelerated ray casting based direct volume rendering. The basic ray casting algorithm still features some shortcomings, and can be further improved and optimized both in performance and in terms of rendered image quality. In the following paragraphs we will discuss in details some methods to improve rendering quality and performance. Unfortunately, in real world applications we often encounter that increased image quality results in decreased rendering performance, mostly due to higher computational demand. Thus, frequently combinations of multiple methods for improving quality and performance are used. However, over-optimization e.g. by trying to incorporate too many improvements simultaneously can reduce performance in the end. Therefore, the employed methods should be carefully selected and combined for the specific application and problem.

4.3.1 Quality

- *Iso Surface Refinement:* Isosurface hit-point refinement is an important quality factor for rendering non-polygonal isosurfaces. After a first intersection estimate of the isosurface along the sampling ray has been determined it is refined iteratively either by bisection search or by linear interpolation [36, 176]. The resulting isosurface is much smoother and thus the image of higher quality. Additionally a lower sampling rate for the first isosurface intersection search can be used, resulting in high quality isosurfaces with improved performance at the same time.

- *Classification:* High frequency components, mainly steep opacity slopes, in the transfer function can cause aliasing artifacts during ray integration (see figure 4.6(a)). Unfortunately, such transfer functions are quite frequently encountered in medical image visualization, e.g. semi-transparent visualization of bone or skin surfaces.

  To cope with the problem of aliasing artifacts, the sampling rate has to be increased to avoid an under-sampling of the volume data along the ray. This however directly decreases performance. As an enhancement of one dimensional transfer functions several solutions have been proposed, all based on the idea of pre-computing an



(a) (b)

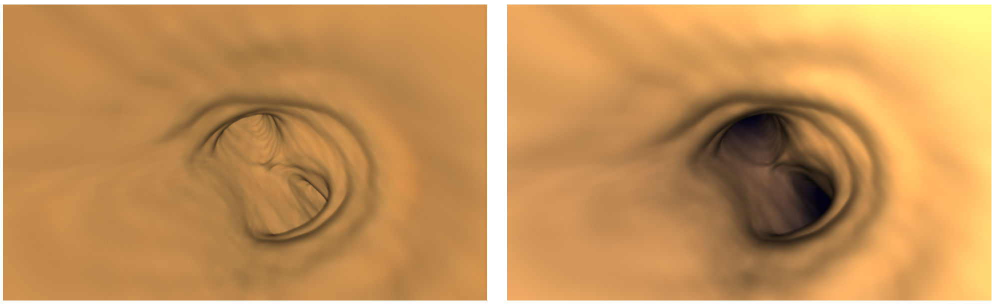
Figure 4.6: Comparison of the difference for (a) standard 1D post-classification and (b) pre-integration for the same rendering parameters (samples per voxel, lookup table) for two CT datasets. Image (a) shows visible ring-artifacts for the transparent skin isosurface from undersampling along the rays, while (b) pre-integration produces an artifact free image.

integral table for ray segments along the sampling ray. The most correct, straightforward, but also brute force approach pre-computes the volume rendering integral for all possible combinations (s_f, s_b), where s_f and s_b denote the scalar values at the front and end of the ray segments. However, the approach has a complexity of $O(n^3)$, with n being the number of LUT table elements. Thus, it is impractical to use for interactive classification.

To solve the complexity problem of pre-integration several methods have been proposed, all approximating the segment integrals with varying degrees of correctness, and various speedups compared to the straightforward approach. In [37], Engel *et al.* present their scheme for generating the pre-integration tables. However the methods has shortcomings with respect to self-occlusion in case of saturated opacity. In [234], post-color attenuated transfer functions are proposed. A separated computation of the color and opacity component is proposed, greatly accelerating the computation, however with reduced correctness due to ignoring absorption inside the ray segment. In [126], Lum *et al.* present a $O(n^2)$ method for pre-integration which is one of the fastest and best approximations of real pre-integration.

All of these methods provide higher quality images compared to post-classification. However, they stil pose a trade-off between quality and interactive classification. Ultimately, real pre-integration (see figure 4.6(a)) provides highest image quality and most correct approximation of the VR integral for any transfer function and data combination. Within the scope of this thesis we have addressed the shortcoming of brute-force pre-integration, by implementing it on the GPU in a fragment shader. This not only greatly accelerates the generation of the pre-integrated lookup table compared to a CPU implementation, but also removes the need for transfers from host to GPU.

- *Floating Point Renderbuffers:* Another important factor for generating high quality images is to use render targets and textures with the correct precision. Inside shaders modern GPUs work with 32bit floating point precision. To avoid clamping of the results when rendering to texture one consequently should employ floating point render targets to store the sub render pass results. Thereby, no precision is lost between the separate stages of the rendering pipeline.
- *Higher-order interpolation:* Higher-order interpolation filters can be a remedy to render noisy datasets with high quality. Furthermore, improved interpolation is beneficial in non-polygonal iso-surface rendering producing smoother surfaces and surface normal vectors. However, these improvements come with the price of additional texture access, often resulting in reduced performance. Thus, these interpolation methods should be carefully selected based on the application and rendering algorithm. In this work we use tri-cubic interpolation only for non-polygonal isosurface renderings. To ensure that performance is not reduced significantly, we combine it with isosurface hit-point refinement and only evaluate the last refinement steps with tri-cubic interpolation.
- *Shading:* Shading is an important visual cue, often aiding in improving the visual perception of the rendering. We use local illumination based shading models for local surface based shading during rendering. A variety of shading models, e.g. Phong, Blinn-Phong, Gooch, Cook-Torrance Shading, were implemented within the scope of this thesis, suited both for DVR and non-polygonal isosurface rendering. To reduce the shading caused artifacts within homogeneous tissue regions in DVR, we conditionally evaluate the shading term only if the local gradient magnitude exceeds a user defined threshold. For an overview of basic, surface-based, shading models in DVR we refer to [36].



(a) (b)

Figure 4.7: Virtual bronchoscopy images of the same datasets with same parameters (a) without depth cueing, (b) with distance to eye based depth cueing enabled.

- *Depth Cueing:* To further improve the visual perception of the rendered images, depth cueing can be implemented. Based on the distance of either the camera or a point in volume space the emissive color is modulated with a pre-defined depth cueing color. In general the color is chosen based on a human perception-physiological basis, e.g. dark blue is associated with further away. In this work depth cuing is used primarily to simulate distance-based attenuation for fly-through applications, e.g. virtual angioscopy and virtual bronchoscopy. For an illustration we refer to figures 4.7.

4.3.2 Performance

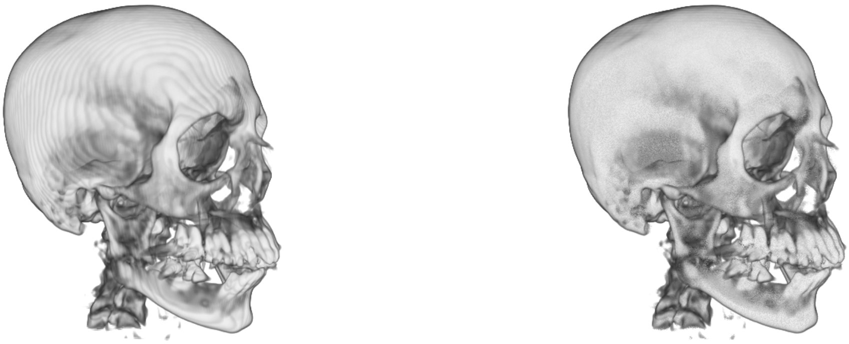
Even on most recent GPUs high quality real-time rendering of volumetric data is a challenging task, due to the computational expenses of the algorithms. In this section we provide a short overview of techniques we implemented and used in our work to improve rendering performance.

- *Variable Render Resolution:* Ray casting is an algorithm that scales very well with the render target resolution. Computational expensive rendering techniques, e.g. shaded volume rendering, can greatly improve the general impression and depth perception, however at the price of reduced performance. To improve performance, it is often advisable to render with a smaller resolution than the final display resolution into an off-screen render target. The rendered image is then in the last step stretched to the final framebuffer size using hardware accelerated bilinear interpolation, which is merely noticeable by the user.

- *Sampling Rate:* The number of samples along a sampling ray is the most basic parameter affecting image quality and rendering performance. In general the volume dataset should be traversed at an appropriate sampling rate, depending on the selected rendering mode and volume parameters. Even at low sampling good image quality can be achieved for both volume rendering and non-polygonal isosurface renderings. Hit-point refinement of isosurfaces and higher quality transfer functions (see section 4.3.1) can help to remove most of the under-sampling artifacts during rendering while still retaining high frame rates.

- *Ray Start Offsets:* Reducing the number of samples will increase rendering frame rates, though at the cost of reduced quality and if reduced too much in clearly visible sampling artifacts. These so-called ring artifacts (see figure 4.8(a)) disturb perception of the rendered image. To make the problem even worse, the human visual system is extremely well trained for detecting such regular patterns in images. In contrast, irregular noisy patterns are often not perceived as much disturbing. By adding an offset to the first sample position along the ray, in general the unit sample distance is multiplied with a random number, the ring artifacts are removed by the added noise (see figure 4.8(b)). This optimization is very well suited for dynamically changing renderings, e.g. during interaction, when moving the volume, or changing the camera, this effect is hardly recognizable by human eyes.

  Although, very well suited for improving image quality, ray-start offsets should be used with care, especially when used for visualization in diagnosis. For diagnosis,



(a) Ring artifacts (b) Enabled ray start offsets

Figure 4.8: (a) Visible ring artifacts arising from under-sampling, (b) same rendering parameters with enabled random ray start offsets.

the optimization should be disabled and images should be rendered with highest quality in order not to introduce noise that could bias the outcome of the diagnosis. This especially applies to naturally noisy image modalities, e.g. US, where additional noise could be perceived as part of the data and could confuse the radiologist or physician.

- *Adaptive Sampling:* Changing the sampling rate during volume traversal is an effective method to improve performance. Homogeneous and empty regions can be skipped by using a larger step size, while inhomogeneous regions with high level of local details can still be sampled with the required sampling frequency. To effectively employ adaptive sampling first one usually needs to extract some information about the local volume properties and store it in a 3D lookup table, that is sampled during ray traversal. However for specific applications, the adaptive sampling rate can be determined procedurally based on volume properties, such as current sample location and direction. This applies especially to data stored in non-Cartesian grids, with varying sample density when remapped to a Cartesian grid representation for volume rendering, e.g. 3D US scan line data stored in a 3D acoustic grid.

- *Early Ray Termination:* Early ray termination is a basic, yet very effective optimization technique to terminate the evaluation of the volume rendering integral as soon as possible. The only condition is to compute the volume rendering integral in front to back direction along the ray. As soon as the accumulated opacity is greater than or equal to one, or a pre-defined threshold all following computations can be skipped as they will not contribute to the visible result. Early ray termination can easily be realized with a single conditional statement in the shader computing the volume rendering integral.

- *Empty Space Skipping:* During rendering typically only a small number of voxels

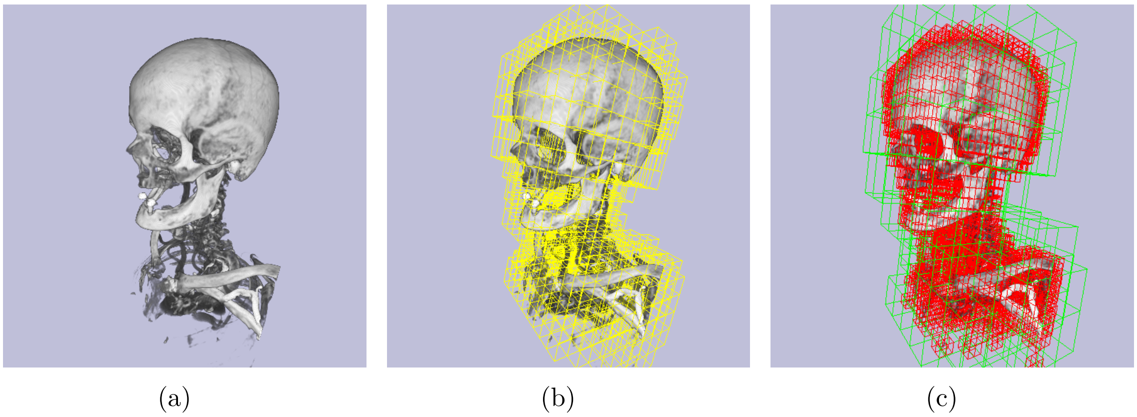

(a) (b) (c)

Figure 4.9: Examples for empty space skipping. (a) DVR of CT dataset and bone trasfer function (b) Octtree based empty space skipping geometry. Active octtree cell boundaries rendered in yellow with DVR. (c) Two-level Brick/Block based empty space skipping. Active brick boundaries rendered in yellow, active blocks rendered in green.

is effectively visible. Most parts of the volume are not visible, either due to zero opacity in the transfer function or they do not lie on the selected isosurface. Of course it is very inefficient to evaluate all stages of the volume rendering pipeline for these parts of the volume. In order to skip all non-contributing voxels empty space skipping was introduced. Empty space skipping approaches replace the simple volume proxy geometry (volume bounding box) by a more complex one. In general the volume is subdivided into multiple cells of the same size. For all voxels enclosed by the cell, and an additional interpolation safety margin of one voxel in every direction, the minimum and maximum values are computed and stored. During rendering the cells are culled with respect to the selected transfer function or iso value. Only the visible/active cells are rendered in order to compute the ray start and exit positions. For highest performance the cell geometry data is stored in an OpenGL Vertex Buffer Object on the GPU. Separate index buffers, for transfer function and iso value visibility, are used to render the active cells. In this work both both an octtree based, and an two-level brick/block scheme for empty space skipping, have been implemented.

Figure 4.9 depicts an example for empty space skipping for a 512x512x460 CT scan. For this example an octtree (see figure 4.11(e)) with 4096 cells each enclosing 32^3 voxels and a brick block proxy geometry (see figure 4.11(f)) were used. Each brick was enclosing 64^3 voxels, each block 16^3 voxels, resulting in 512 bricks and 29696 blocks. For the depicted transfer function 29% octtree cells were active compared to 45% of the bricks and only 15% of the blocks in the brick-block scheme. Using the occtree proxy geometry 45 frames per second (fps) were measured compared to 65 fps using the brick-block proxy geometry. Both approaches used OpenGL vertex buffer objects to accelerate the drawing of the proxy geometry. The better performance of the brick-block scheme is mainly due to a higher resolution proxy geometry compared to the octtree approach, resulting in less volumes samples during ray casting.

- *Deferred Rendering:* Deferring computations as long as possible and employing lazy evaluation throughout the rendering pipeline is an important factor for improving the performance. Costly computations, e.g. gradient calculation or shading term computation, should be conditionally evaluated and deferred as long as possible. It often makes sense to split a complex render pass into two more simple ones to benefit from deferring computations, e.g. separating isosurface computation and shading into two passes.
- *Resource Sharing:* Although modern GPUs are equipped with ever increasing onboard memory, excessive use of the GPU resource will degrade performance. Temporary textures, render targets, and buffers should be reused in multiple stages of the rendering pipeline instead of replicating them. This guideline does applies mainly to multi-pass volume rendering algorithms (see section 4.4.3), where even for small image resolutions the memory limit can be reached quite easily.
- *Pre-Computed Volumetric Gradients:* On-the-fly gradient computation, either using central differences or better a Sobel Filter, will give the highest quality estimate for the gradient, however largely affecting overall rendering performance. An alternative is to use pre-computed gradients. The gradient is estimated only during the initialization phase before the volume data is uploaded to the GPU. The downside of this approach is an increased memory consumption on the GPU of at least four times the size of the original data. Additionally, precomputed gradients are stored with the precision of the volume dataset, thus are of lower quality than those computed on-the-fly. However, the latest generation of commodity graphics cards offers support for hardware accelerated interpolation of floating point format 3D textures. CT data is usually stored with 12bits and requantisized to 16bit before download to the GPU. Using a 16bit floating point format for the 3D texture storing the volumetric gradient and original intensity data is a good trade-off between speed and quality, yet keeping the amount of used memory to a minimum.

In this section we have covered basic GPU-accelerated ray casting, as well as several methods to optimize rendering performance and image quality. The discussed methods serve as a foundation to implement a high quality, real-time volume render using GPU ray casting. However, there exists a large amount of related work to explore for the interested reader. Therefore, we refer to the corresponding journals and conference proceedings in the computer graphics, volume graphics and visualization community. As start for research on related work we refer to the book [36] and the recent and excellent tutorial [61] on GPU-accelerated volume rendering

4.4 Advanced Rendering Techniques

Now that we have covered the basic and advanced concepts for direct volume rendering we want to conclude this chapter with the following sections on advanced rendering techniques for medical image data.

4.4.1 Rendering time-varying Data

Most phenomena in nature are dynamic over time and can not be adequately studied by inspecting only a single isolated snapshot in time. This applies as well to the study of the output of time-resolved numerical simulations in science, as to 4D data, provided by modern medical imaging technologies (e.g. 4D CT and US), for diagnosis and therapy. 4D CTA is for instance used to analyze the aortic valve (see figure 4.10) and to generate patient-specific valve models [80].

4D ultrasound is today used primarily in cardiac applications for diagnosis and for monitoring cardiac interventions. Current research focuses on facilitating 4D US image guided procedures [122, 143].

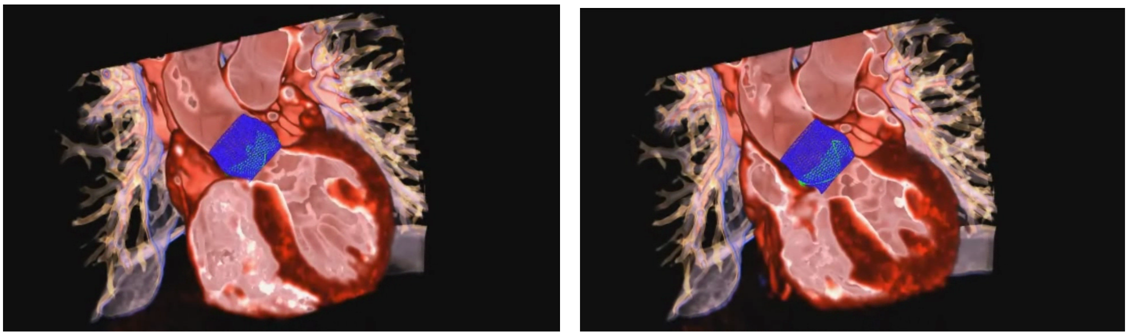

(a) (b)

Figure 4.10: Exemplary rendering of two 3D volume frames from a ten frame 4D CTA dataset of the human heart. Inside DVR a mesh representation of a automatically extracted aortic valve model [80] is rendered. Contrast inside vessels and heart chambers is removed by a combination of 3D volume filtering and pre-integration during ray-casing.

The underlying volume rendering techniques used for 4D rendering are the same as used for visualization of 3D data. The key challenge for realizing a real-time 4D renderer is the efficient data transfer to the GPU. Additionally, the problem is largely application and data size dependent. For instance the upper memory consumption for a currently typical single 3D ultrasound volume can be estimated with 16 Mbytes (256^3 voxels stored with 8bits per voxel). In general the memory usage per volume is even less.

However, during live US imaging the data has to be transferred directly after acquisition, without a lag, onto the display of the US system. Typically every second 20 to 30 volumes have to be transferred and visualized by the rendering system. Assuming 16 Mbytes per volume and 30 volumes per second this accumulates to 480 Mbytes/sec that have to be transferred over the bus system of the host computer and transferred to the GPU for rendering. Typical current commodity GPUs can transfer between 600 Mbytes/sec to 1000 Mbytes/sec.

To improve transfer performance, recent computer graphic APIs provide buffer primitives to accelerate the transfer from host to GPU. A buffer in GPU memory, e.g. OpenGL pixel buffer object (PBO), can be allocated and mapped to host address space where the new texture data is then written direct to GPU memory. The transfer from PBO to texture is in most cases more efficient than host memory to texture transfers, as at least one memcpy inside the driver is skipped.

Live 4D imaging is only one application, more often recorded 4D data has to be visualized for diagnosis and study purposes. As long as the 4D data fits into GPU memory buffers, there is in general no problem. All volume frames can be copied into PBOs and during visualization the frames are copied on the GPU into a 3D texture that is used for volume rendering. However, the problem quite often occurs that the data is too big to fit into GPU memory, e.g. 4D CT data, and the transfers from host to GPU can stall the rendering pipeline.

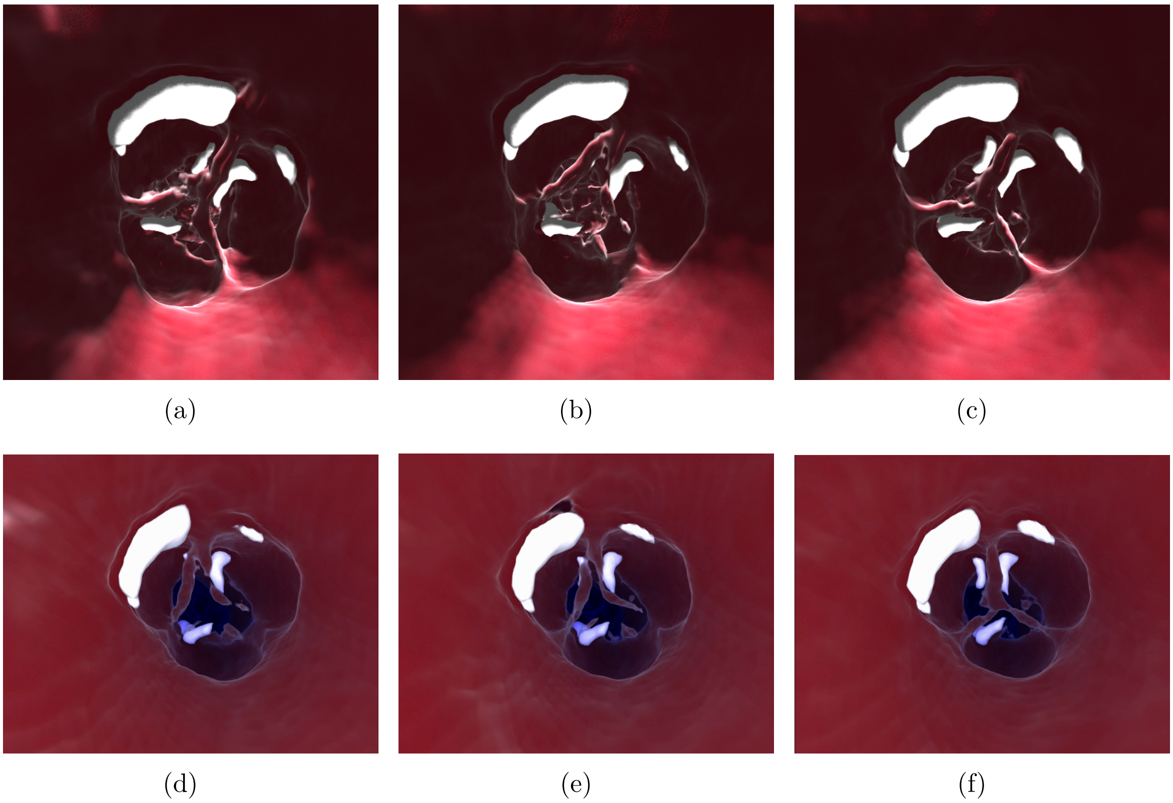

(a) (b) (c) (d) (e) (f)

Figure 4.11: Exemplary rendering of six 3D volume frames from a ten frame 4D CTA dataset of the human heart. The camera is aligned with respect to the aortic root and facing towards the aortic valve. Images in first row depict results with per volume applied Gaussian smoothing to reduce contrast artifacts. Second row depicts additional distance to camera based depth cueing to improve depth perception.

In the scope of this work we have accelerated 4D rendering by using OpenGL pixel-buffer objects to store the 4D data if possible inside GPU memory. This approach works efficiently for real-time 4D rendering of 4D US data, mid resolution or down-sampled 4D CT/CTA data. Figure 4.10 and figure 4.11 show exemplary images generated from a ten frame 4D cardiac CTA dataset. The individual volumes were sufficiently small to fit the entire sequence into PBOs on the GPU for rendering. In this work we mainly addressed the joint visualization of heart valve model geometry, and high quality DVR. One of the challenges was to automatically remove contrast dye artifacts during rendering inside the aorta and heart chamber, as wells as to improve visual perception by adding clipping, shading and depth cueing effects. To reduce the contrast dye artifacts we employed a 3D Gaussian blur filter implemented on the GPU to allow for interactive adjustment of the filter parameters and real-time visualization.

4.4.2 Stereo Rendering

Traditionally data is visualized on a single screen, e.g. volume inspection in the radiology reading room. Recent developments in human-computer-interfaces and display technology offer the possibility to visualize data in 3D using different approaches, e.g. stereoscopic HMD, anaglyphic rendering, shutter glasses, auto-stereoscopic multi-view displays [30]. A common property of all this technologies is the requirement to render the volume data for two or more views (see figure 4.12) to achieve stereoscopic perception.

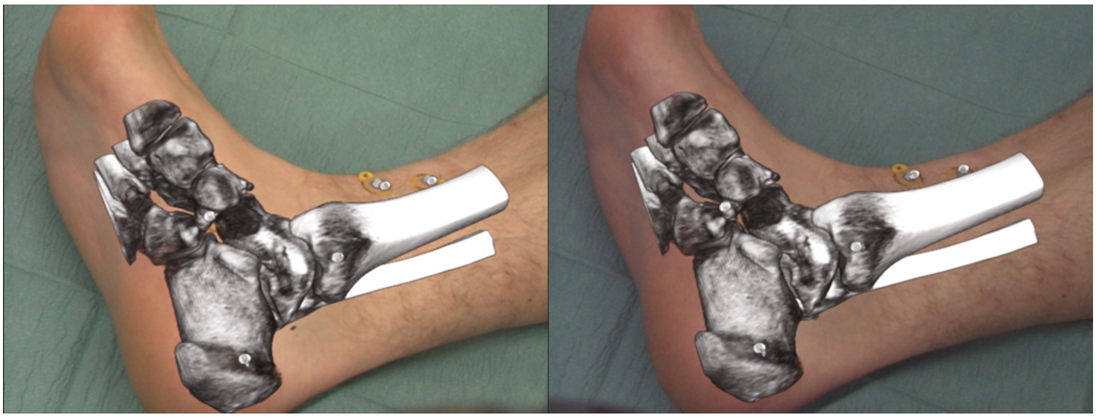


Figure 4.12: Examplary stereo rendering generated for medical AR visualization. For each of the two views a volume rendered image with the respective camera matrices has to be generated in every frame.

In this work we have focused on stereoscopic volume visualization for display on a stereo video see-through HMD (see figure 4.12 and chapter 5 for more details on the medical AR visualization). Multiple approaches have been evaluated in terms of ease of implementation and performance, described in detail in the following listing.

- *Interleaved rendering per eye:* The first approach is the straightforward extension of the single view case. The single view rendering code is nested inside a loop in which the projection and modelview transformation per eye is set and the rendering code is called. The results of the individual per eye passes are stored in off-screen render targets and for final display the two textures, for left and right eye, are mapped to the respective regions in one large viewport and displayed on the screens of the HMD.

- *Interleaved proxy geometry rendering, simultaneous stereo volume rendering:* In this approach we investigated if it is beneficial to activate the costly ray casting rendering passes only once for the whole large viewport instead of rendering the volume twice to two smaller off-screen render targets. Without using a geometry shader (see third approach) the ray proxy geometry still has to be rendered using two passes per eye. It is impossible to project the same geometry twice into different viewport region using only vertex and fragment shaders in one pass. Once the proxy geometry is rendered the volume rendering is triggered for the whole viewport and the stereo views are rendered simultaneously.

- *Simultaneous stereo proxy geometry and volume rendering:* The last approach deals with the shortcomings of the second and employs a geometry shader to duplicate the proxy geometry and project it using different projection and modelview matrices for each eye. Thus, the complete rendering can be done in one pass for the two views. The duplication of every geometric primitive increases the load on the geometry stages of the rendering pipeline. Especially, for proxy geometries with large numbers of primitives this can have a downside effect on performance.

The performance of the three approaches is very much data and rendering setting dependent. When using a simple cube as proxy geometry the geometry shader approach outperforms the two other approaches. However, when enabling more complex proxy geometry for empty space skipping the geometry shader based approach falls back behind the other approaches due to increased amount of geometry generated in the geometry shader. This, might change in the future with more efficient GPU hardware and improved driver versions. Initially the geometry shader approach was falling behind 50% in performance to the interleaved rendering approach. But, with over time improving graphic drivers the geometry shader-based approach's performance increased, resulting in about 90% of the interleaved performance for complex proxy geometries.

In terms of ease of implementation either the interleaved one or the geometry shader-based approach are preferred as they require only very few local changes to the rendering code.

4.4.3 Focus and Context Rendering

Standard volume rendering techniques often face the problem of information overflow.In many applications the user is only interested in visualization of a specific VOI or also called focus region. However, traditional volume rendering techniques possess no means of distinguishing between important/focus information and the rest of the data, denoted context. Recent works on illustrative, importance based, and focus and context based visualization techniques address that specific problem. Viola *et al.* [214] propose importance based volume visualization, adding an additional importance volume encoding an importance value for every voxel in the real data volume. This can be for instance a multi-label segmentation volume, with a specific importance label for every segmented organ. During rendering the importance volume is sampled and the opacity of the current data volume sample is modulated based on the importance value. This guarantees that unimportant features that occlude the important ones are rendered transparently providing insight into the dataset. Burns [24] and Haidacher [62] extend importance driven volume rendering with contextual cutaway views and combine 2D ultrasound and 3D CT image data based on importance. Instead of an importance volume they use an additional 2D importance transfer function that is indexed by intensity and gradient magnitude. Thereby, important features can be interactively classified during rendering and time-consuming segmentation is avoided.

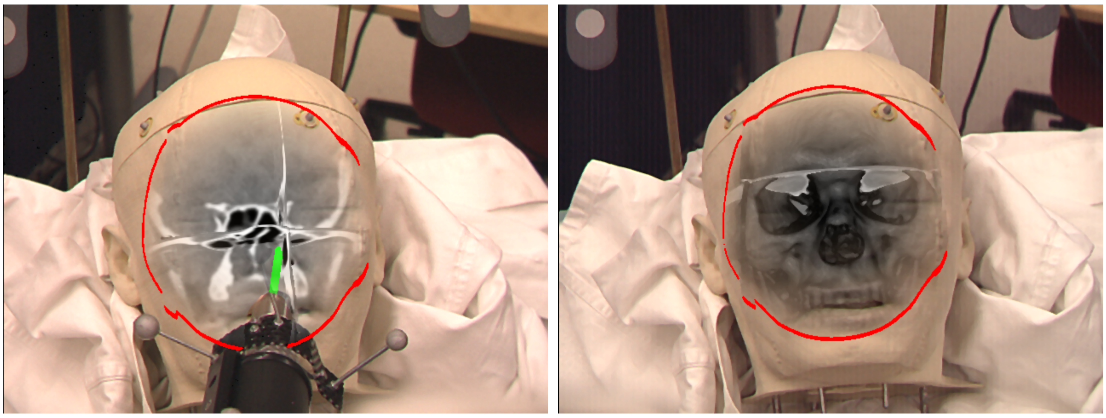
In [105] Krüger *et al.* propose ClearView for segmentation free, real-time, focus and context visualization of three dimensional image data.Their focus and context rendering technique focuses the attention of the viewer on the focus zone, in which the important features are embedded. The transparency of the surrounding context layers is modulated, e.g. based on context layer curvatures properties or their distance to the focus layer. The context and focus are extracted on the fly from volume data, but can also be represented by traditionally rendered geometry. The technique is quite flexible and straightforward to implement.

In this work we extended ClearView for visualization in medical AR and fused volume rendering of multi-modal data. The modifications mainly concern rendering performance, with special emphasis on maintaining real-time performance for dynamic scenes, as in a medical augmented reality environment. The original ClearView method generates the focus and context layers sequentially one by one. For each layer the previous isosurface layer in view direction is used as start for ray casting. However, no visibility information with regard to the focus region position and size is taking into account at this stage.

Thus, for every layer ClearView will extract the maximum amount of information from the volume data. Determination of contribution of the layer pixel values is deferred to the final compositing pass. This allows interactive visualization when changing the compositing mode, shading, focus region position and size, as well as layer properties. Only the deferred compositing pass has to be re-evaluated and no re-computation of the layer information is required. This defers the costly layer information extraction passes as late as possibly, until parameters, e.g. projection modelview matrices, transfer function or iso value, effecting the layer have changed.

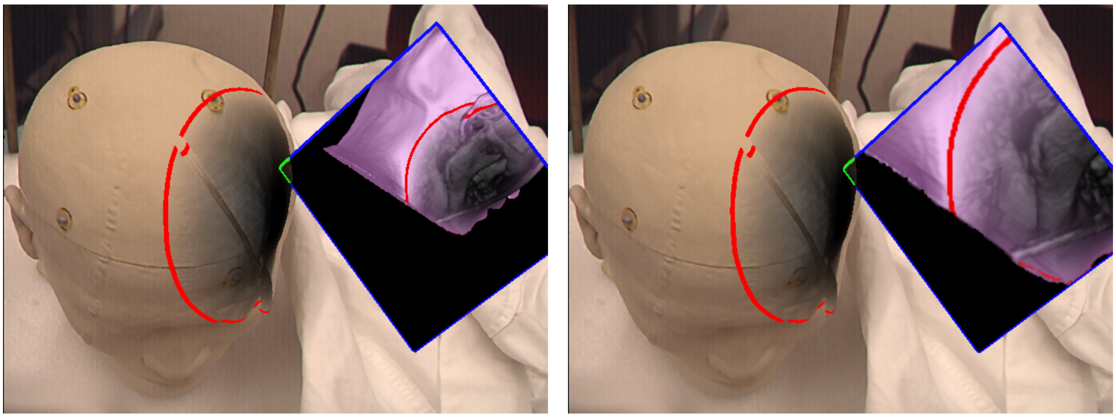
However, for our intended use of ClearView in medical AR this deferred rendering scheme actually decreases performances. We are facing a environment with from frame to frame changing modelview matrices, thus requiring an efficient re-computation of all layers and compositing for every frame. Therefore, we incorporate the focus point position and focus region size already when generating the context and focus layers. For each extracted isosurface layer we compute the intersection with the focus region per layer fragment and mask only the fragments which are valid ray start positions for the next layer in ray-direction. Thereby, no computations are performed for fragments non-visible in the final composited image. Especially computationally more expensive rendering techniques benefit from this modification. For optimal performance our ClearView implementation allows the separate parameterization of each layer, and an arbitrary volume rendering technique per layer for maximum flexibility.

Furthermore, we integrated embedded MPRs rendering into the ClearView focus region and improved depth perception by introducing an additional focus and context rendering enabled virtual mirror. The extension are discussed in detail in the following paragraphs.



(a) Three axial MPR planes rendered inside focus region. Drill tip controls intersection point of MPR planes.

(b) Single MPR plane rendered inside focus region with DVR layer displaying bone structures.



(c) Virtual mirror inside scene

(d) Zoom-in inside virtual mirror

Figure 4.13: MPR and virtual mirror extensions for focus and context rendering.

4.4.3.1 Embedded MPRs

MPR rendering is one of the most well known and most used rendering techniques in today's clinical practice. Clinical users frequently employ MPR rendering to generate 2D images from 3D image data by re-slicing the 3D data, using either slicing planes perpendicular to one of the major anatomical directions or user defined oblique slicing planes. Especially, for performing measurements from the image data MPRs are used predominantly. In [207, 208], Traub *et al.* investigated the use of in-situ visualized axial and instrument axis aligned MPR planes for medical AR. The combination of traditional axial and instrument aligned MPR visualization was shown to be of advantage for image-guided surgical navigation.

Traditionally MPR planes are rendered as opaque planes, with enabled 2D or 3D texture mapping from a 3D data set. However, for the integration with the focus and context visualization this approach has to be adapted. The MPRs have to be clipped with respect to the first most context layer and focus region. Therefore, MPRs are rendered after the first context surface has been extracted from the volume data and a depth test in a shaders is used to clip fragments that are in front of the first context surface in view direction (see figure 4.13(a) and figure 4.13(b) for examples of MPRs rendered inside focus zone). Currently, the MPR context layer intersection is only computed for the front side of the context isosurface. The backside of the context surface is actually never extracted in the current implementation as it is in most cases occluded by inner context or focus layers anyway. However, for the special case of MPR rendering this results in visualization of sharp plane borders towards the end of the focus region. One approach to solve this problem, could be to apply distance-based opacity modulation for the MPR fragments, slowly fading them out with increasing distance to the eye.

4.4.3.2 Virtual Mirror

DVR allows for the inspection of the volume data from any perspective simply by changing the current position and orientation of the virtual camera. However, for several applications in image guided interventions, and especially medical AR, changing the virtual camera position or orientation is not possible or not desired, e.g. physical restrictions in movement of physician or patient. However, restriction to certain perspectives has some severe effects on depth perception and the amount of information visually extractable from the 3D dataset.

In real life, mirrors are are used to overcome perspective limitations for a very large number of applications, e.g. mirrors in dentistry, security inspection of car undersides, or traffic mirrors for road safety. Analog virtual mirrors have been proposed to solve the problem of perspective limitation in computer graphics [48, 95] and medical AR [14]. Because of the widespread use of mirrors in real life, the virtual mirror requires no introduction or training of the users and is used rather intuitively. The virtual mirror offers the user an additional motion perspective and motion parallax for depth cues by changing the position and orientation of the virtual mirror, without having to move the virtual camera. In addition, essential depth hints are provided by the mirror perspective for cases of misleading depth information gained from the occlusion and relative size depth cues.

The advantage of the virtual mirror for medical AR-based navigation was demonstrated by an experiment performed by Bichlmeier *et al.* [13]. Several surgeons were asked to perform a drilling task on a lumbar vertebrae phantom, once using a monitor-based navigation system and once using in-situ based visualization with virtual mirror. Using the virtual mirror a more accurate drilling result is reported. Navab *et al.* [138] also proposed the virtual mirror for augmented laparoscopic surgery.

We integrated the virtual mirror into the focus and context rendering pipeline by introducing a new layer, denoted virtual mirror layer (see figure 4.13(c) and figure 4.13(d)). In this work we use a render to texture technique for rendering the reflected scene inside the virtual mirror. The mirror geometry is restricted to a planar surface of certain size and shape, e.g. rectangle or circle. For rendering the mirrored scene, all modelview matrices of object in the scene, e.g. volume, virtual surgical instrument, have to be multiplied by an additional transformation matrix T_M. This matrix mirrors everything with respect to the mirror plane P_M defined by $P_M := (\vec{n}, \vec{p})$, where $\vec{n} := [x_n, y_n, z_n]^{\mathrm{T}}$ is the unit length plane normal vector and $\vec{p} := [x_p, y_p, z_p]^{\mathrm{T}}$ a point that lies inside the mirror plane and inside the mirror geometry. The transformation matrix is then given as defined in [95]:

$$T_M = \begin{pmatrix} 1 - 2 * x_n^2 & -2 * y_n * x_n & -2 * z_n * x_n & 2 * \vec{p} \cdot \vec{n} * x_n \\ -2 * x_n * y_n & 1 - 2 * y_n^2 & -2 * z_n * y_n & 2 * \vec{p} \cdot \vec{n} * y_n \\ -2 * x_n * z_n & -2 * y_n * z_n & 1 - 2 * z_n^2 & 2 * \vec{p} \cdot \vec{n} * z_n \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.6}$$

For ease of use it is recommended to define $\vec{p}$ and $\vec{n}$ in object space coordinates because this results in a mirror transformation matrix which can simply be multiplied onto the current modelview matrix. However, this requires to transform $\vec{p}$ and $\vec{n}$ first from the mirror coordinate system to each object coordinate system to obtain the correct respective mirror matrices by evaluating equation 4.6. Applying the mirror matrix will result in changing the polygon winding order, which is used in computer graphic APIs to determine if a polygon front or back face is directed towards the camera. Thus, if polygon culling is enabled one has to change the currently active cull face prior to rendering the mirrored geometry.

Virtual Mirror Extensions Adding the virtual mirror into the scene already greatly improves depth perception, understanding and estimation of relative distances. However, for practical use we present the following extensions.

- *Alignment to a surgical instrument:* For most medical navigation tasks, an instrument is involved and the user is especially interested in viewing a special region of the instrument, e.g. the drill tip point of a surgical drilling device. By making the position and orientation of the virtual mirror dependent of the current instrument position and orientation, it is guaranteed that the instrument region of interest is always visible inside the mirror's view.

  Alternatively the mirror can only be attached to the instrument for initial mirror positioning, e.g. drilling dry run, and then be fixed to the user selection position and orientation for the rest of the procedure. Thereby, no additional user interfaces are required to control the placement of the virtual mirror inside the scene.

- *Mirror specific focus point:* Using globally defined focus point coordinates can result in small or no visible focus regions inside the mirror view. To ensure that focus information is also always visible inside the mirrors this extension redefines the focus region center for the mirrors, making it independent of the main perspective focus region center. When rendering focus and context layers for the mirrored scene, the focus region center position is evaluated e.g. by picking the first context layer position for the ray passing through the mirror center point, or set to a point of interest on a surgical instrument.

However, by de-synchronizing the main focus and mirror focus regions the global mirror effect is de-naturalized to a certain degree, as parts, mainly the focus region, of the mirror scene appear to be incorrectly mirrored on a first view. Because the generated images do not match the users expected images, this is rather confusing and diminishes the added improvement of always viewing the focus region in both views.

- *Adaptive mirror texture size:* Adding a virtual mirror into the scene greatly increases the computational complexity for rendering one image, mainly due to the doubled number of volume rendering passes. To reduce the computational load we render the mirror image with a reduced render target resolution to improve performance and use bilinear texture interpolation to stretch the result image when compositing the final scene.

- *Zoom-Effect:* The mirror's footprint in the final 2D image is in most cases rather small and it can be hard to view specific details inside it. However, the user of the system is interested in viewing a specific anatomic region inside the mirror in sufficient detail. To improve it's visibility inside the mirror view we implemented a zooming effect for the virtual mirror view (see figure 4.13(d)).

  When texture mapping the virtual mirror image into the final image we apply a scale and offset to the texture coordinates in the fragment shader which results in an enlarged view of the mirror scene.

4.4.4 Rendering multi-modal Data

Today, more and more medical applications require the simultaneous visualization of multi-modal data, usually two or more data sets from different modalities. 3D SPECT or PET data is often visualized jointly with a CT or MRI data set of patient to provided anatomical contextual information. However, multi-modal visualization is gaining more and more importance for medical applications, and will be of even greater importance in the future. Fused visualization of multi-modal data aids in understanding by combining complementary information from two or more different sources. In this work we have started to investigate multi-modal visualization mainly for fused visualization of SPECT and CT data (see figure 4.14(a) and figure 4.14(b) CT datasets courtesy of SurgicEye GmbH[1]).

In our setup, we had a very controlled environment, knowing that the SPECT data is always inside the CT data and does not move dynamically. In the depicted images (see figure 4.14) we combined focus and context rendering and fused DVR of SPECT and CT. The skin context layer is extracted solely from the CT data and only within the focus region a fused DVR image is computed by sampling both the SPECT and CT data during ray casting. Thereby, the complexity of the fused rendering is greatly reduced, as we do not have to cope with a dynamic scene and moving volumes. Furthermore, the DVR is

---

[1] http://www.surgiceye.com

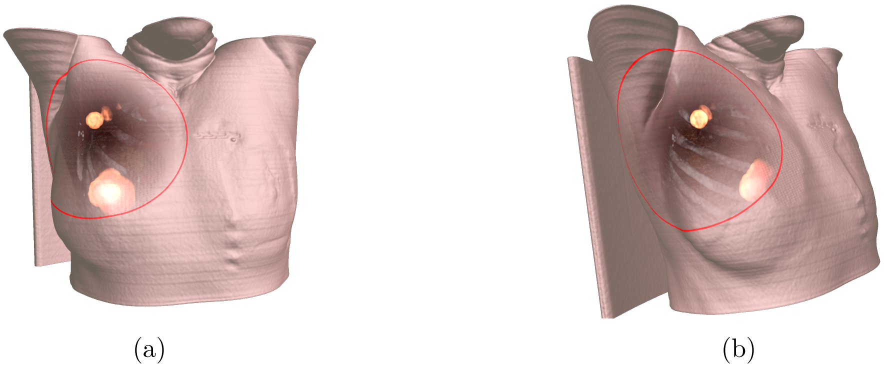

(a) (b)

Figure 4.14: Fused, focus and context SPECT/CT rendering. A nuclear tracer was injected to highlight the peripheral breast lymph nodes for therapy planning in breast cancer. CT datasets courtesy of SurgicEye GmbH

only executed for pixels visible inside the focus region. Thus, the performance is very good due to the low number of rays having to sample both datasets.

However in general this can not be assumed. The challenge is to efficiently visualize any multi-modal volume configuration. In general the volume configurations consist of multiple volumes with different sampling resolution, and arbitrary position and orientation with respect to each other. Simply taking into account all volumes when computing the volume rendering integral results in a highly computational costly algorithm.

We therefore refer the interested reader to related work addressing this special problem in volume graphics in detail.

Lindholm *et al.* [120] propose an efficient, view-order independent method for fused volume rendering. A BSP tree is used for efficient rendering, splitting the rendering into segments of overlapping volumes and combining it with automatic shader generation for all occurring combinations of volume intersections per segment.

In [8], Beyer *et al.* present a highly sophisticated multi-modal, multi-volume renderer for pre-operative planning in neurosurgery. Their renderer allows simultaneous visualization of CT, MRI, fMRI and segmented volume data.

Kainz *et al.* present a Nvidia CUDA based volume renderer that is capable of rendering multiple intersecting volumes and combining it with opaque and transparent geometry rendering [87]. The renderer achieves interactive frame rates for dynamic scenes with volumes moving and intersecting.

4.5 Summary

In this chapter we have presented the basics of direct volume rendering, GPU accelerated ray casting, methods to improve rendering performance and quality. We have given an outlook into advanced rendering techniques for medical applications treated within the scope of the thesis and beyond.

Real-time, direct volume visualization has been an invaluable tool within the scope of this work. The described rendering techniques have been used throughout this work, primarily to visualize medical image data, and additionally to accelerate 2D-3D registration, and have been integrated into several other works and projects within the scope of this work.

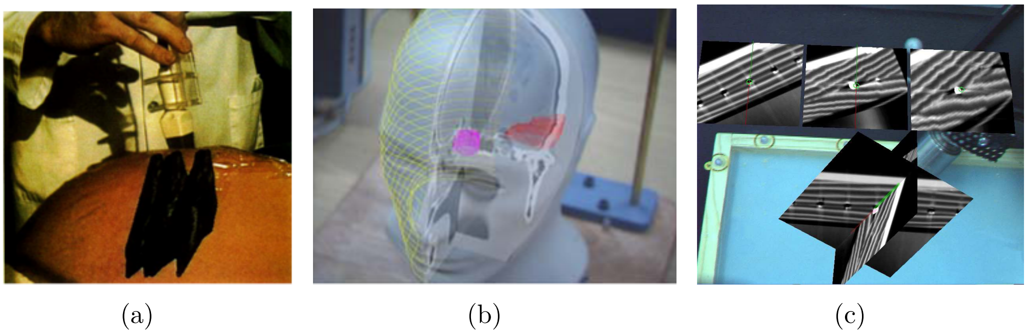
CHAPTER 5

Advanced Medical Augmented Reality Visualization

Real-time visualization is one of the core tasks of any augmented reality system. Besides real-time rendering performance, the visualization has to provide the user with the appropriate visual cues at the right time to support completion of his task. Augmented Reality was introduced as an alternative to monitor based visualization for the presentation of medical image data during surgery. Exemplary systems were proposed within neurosurgery [34, 150], maxillofacial surgery, orthopedic surgery [17, 208], and liver surgery [45]. Existing solutions for medical augmented reality often implement real-time rendering by reducing the quality or using simple rendering techniques. Simple wire-frame surface models or two dimensional slice rendering is applied to achieve a real-time system update rate. Many methods require time consuming preprocessing of the volume data and manual interaction. More sophisticated methods enhance 3D perception within their stereo systems e.g. by transparent surfaces [34], virtual mirrors [13], or viewing windows [6, 11] into the body. Important visual cues, such as shape from shading, depth from occlusion or motion parallax can help the user of a medical augmented reality system by an improved 3D perception of the three dimensional data. In the this chapter we summarize techniques for efficient GPU-accelerated in-situ focus and context visualization of medical image data for medical augmented reality using a stereo video see-through HMD. Furthermore we propose smart extensions to improve visual perception while ensuring the required real-time system update rate.

5.1 Related Work

The field of related work from the fields of computer graphics and medical augmented reality is huge. A number of related augmented reality systems and their medical application was already presented in the introduction of this chapter. Thus, in this section we want to focus on a number of works very closely related to our presented work. Techniques and related work for GPU-accelerated volume rendering and focus and context visualization from the computer graphics community are discussed in chapter 4 of this work. Thus, we refer the reader to refer to the specific sections and links to related works in literature.



(a) (b) (c)

Figure 5.1: (a) Tracked US image augmentation on the belly of a pregnant woman, by Bajura *et al.* [6]. (b) Screenshot from Ramp system of Sauer *et al.* Image from Vogt *et al.* [215]. (c) Hybrid Navigation interface by Traub *et al.* [207].

One of the earliest works on real-time augmentation using a video see-through HMD was presented by Bajura *et al.* [6] in 1992. They describe a system for augmentation of tracked 2D B-Mode US images within the body of a patient. The system setup is demonstrated by augmenting 2D US images of a fetus within the abdomen of the pregnant mother (see figure 5.1(a)). Furthermore they present the idea of rendering a virtual hole, a predecessor of the virtual window technique in later works, to increase the realism of the visualization and attempt to avoid conflicting depth cues.

Sauer *et al.* [171, 173, 215] developed the Reality Augmentation for Medical Procedures (RAMP) system, for real-time in-situ visualization in medical augmented reality (see figure 5.1(b)). The system consists of a stereo video see-trough HMD and an inside-out tracking system. As a large part of the work presented in this chapter is based on the RAMP hard- and software for inside-out tracking and augmentation on the HMD, more details on HMD and tracking are found in section 5.11(f). The RAMP system was evaluated in multiple experiments for supporting different medical navigation tasks. For real-time visualization, mesh models generated from segmentation or MPRs and local VOI volume rendering is used.

In [207, 208], Traub *et al.* combined traditional axial MPR visualization with MPRs augmented within the anatomy of interest in the HMD field of view. From existing monitor based visualization navigation systems surgeons are used to work with axial MPRs when manipulating a tracked surgical instrument. In their work the authors compare the purely monitor based navigation with axial MPR, versus pure AR and a hybrid AR mode (see figure 5.1(c)) in a drilling experiment with three trauma surgeons. Their results indicate that monitor-based visualization is less favorable than in-situ visualization. Mainly due to the additional mental strain to transfer the monitor visualization to the patient anatomy. Although the accuracy was not improved in the experiment by the AR visualization, it resulted in faster task completion due to the more intuitive user interface.

Recently, the great potential of Focus and context visualization techniques for enhancing augmented reality visualization has been discovered. Similar to virtual windows into

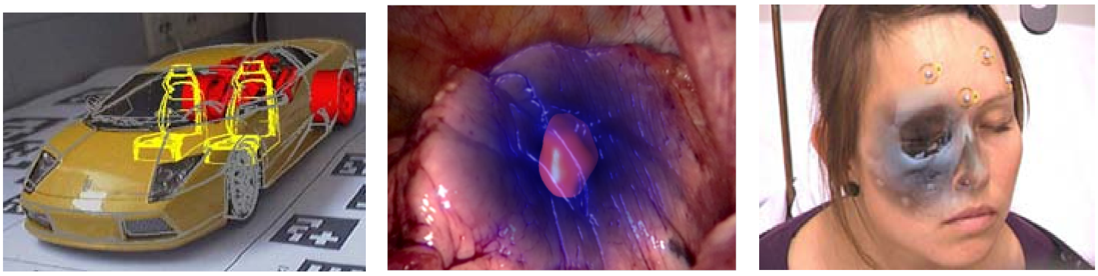

(a) (b) (c)

Figure 5.2: (a) Focus and context visualization for AR presented by Kalkofen *et al.* (b) pq-space NPR augmentation by Lerotic *et al.* (c) Contextual Anatomical Mimesis by Bichlmeier *et al.* [15]

the patient proposed by earlier works [6, 11] the focus zone can be embedded into the real world video image. By moving either the focus zone over the object or an user moving around a fixed focus zone already an impression of depth by the motion parallax is gained. Context layers increase depth perception by layered, partially transparent rendering above the focus region. Kalkofen, Lerotic and Bichlmeier were the among the first to propose focus and context visualization techniques for augmented reality [15, 88, 116]. In [88] Kalkofen *et al.* integrated surface based focus and context visualization into an augmented reality framework (see figure 5.2(a)). Additionally context information, mainly edge information, is superimposed occluding parts of the real objects to intelligently combine real and virtual worlds. In order not to confuse the user with too much virtual focus and context details a selective rendering techniques is employed. User interaction is included with a magic lens metaphor, to restrict focus and context visualization to the lens area. The proposed techniques are mainly demonstrated on a tracked car model, only a short part of the work addresses the potential application in medical augmented reality applications.

In [116], Lerotic *et al.* present a focus and context visualization technique, denoted *pq-space rendering*, for augmented reality visualization in robot assisted minimally invasive surgery. In contrast to [15, 88, 105] their approach is not using any 3D information, e.g. volume data or 3D meshes, but only information extracted from live feed video images. The video image is analyzed to extract a context information layer from the exposed anatomical surface. The resulting context layer contains mainly edge and local high curvature information. This context layer is then rendered with a non-photo-realistic rendering technique, on top of a traditionally rendered image of the anatomy of interest (Focus). Similar to the magic lens metaphor the real video image and the focus and context image can be combined using a circle pattern, creating the impression of a X-ray vision effect (see figure 5.2(b)). In experiments the by the proposed method improved depth perception for surgical navigation tasks was demonstrated.

Bichlmeier *et al.* [15] present a focus and context rendering centered approach for *contextual in-situ visualization* in medical augmented reality. Their approach is heavily related to the work of Krüger *et al.* [105]. The key contribution of their work is the intelligent natural embedding of the focus zone inside the video image (see figure 5.2(c)). Instead of traditionally rendering the skin context surface, the video image transparency is modulated based on the skin context surface and current view properties, e.g. local curvature, dot product of view direction and surface normal vector. However, the methods have not been fully integrated into the medical augmented reality system with the same degree of quality and performance as in the field of computer graphics. Instead of using on-the-fly extracted focus and context layers using GPU-accelerated ray-casting, the presented methods are limited to rendering of surface meshes. Besides these drawbacks, the proposed video transparency modulation technique for the skin context layer is promising as it enables a natural integration of the focus zone into the real video images.

5.2 Contributions

The common limitation of most related works [15, 88, 116] is the restriction to rendering polygonal anatomy mesh models. In general mesh data is extracted by time consuming segmentation from the medical volume data set. Although the relevant anatomy and structures can be depicted, the approach suffers from the lack of real volumetric visualization. Important and standard medical visualization modes are impossible to achieve with these visualization techniques, e.g. MIP, DRR and/or volume rendering of soft tissue. An additional limitation is that the visualization is static and fixed to the time of mesh generation. Information from volumetric data acquired during a medical procedure, e.g. 3D/4D US or DynaCT, can not be directly integrated into the augmented reality visualization. In [103] Kratz *et al.* present their results for the integration of GPU-accelerated volume rendering into a virtual reality environment for medical diagnosis and education. The presented methods and techniques are closely related to our work. However, their work purely concerns virtual reality and not augmented reality. Thus, AR specific important key problems, e.g. fusion of real and virtual data is, interaction of real and virtual objects, avoiding depth cue conflicts, is not addressed.

This chapter demonstrates how integration of state of the art volume visualization techniques can be used to overcome this limitation. The key contributions are (1) Integration of a state-of-the-art GPU-accelerated volume renderer into a medical augmented reality framework for real-time stereo augmentation using a video see-through HMD. (2) Adding GPU-accelerated techniques for handling occlusion problems caused by physician hands and surgical instruments within the field of view of the user. (3) Design of a specialized real-time rendering pipeline, emphasizing rendering effects to address several important depth cues of the human visual system, e.g. motion parallax, occlusion, shading. Exhaustive tuning, optimization and evaluation of the rendering and complete system performance to guarantee real-time system update rates. (4) Evaluation of the system by supporting a surgical drilling task in an experiment with six surgeons on a realistic human phantom.

5.3 Medical Augmented Reality Framework

The medical AR system used throughout this work consist of two optical tracking systems, a video see-through HMD, a high performance PC workstation with high end GPU, and a software framework (CAMPAR [190]) to combine input data from all hardware systems and drive the visualization on the HMD. The individual components are described in detail in the following sections.

5.3.1 Head Mounted Display

A video see-through HMD was used in this work, with two color video cameras and one monochrome camera, sensitive to infrared light, for optical tracking mounted on top of the HMD. It is initially known as RAMP system [171, 174, 215] in the Computer Aided Surgery (CAS) community. The RAMP hard- and software for inside-out optical tracking and augmentation on the HMD was originally developed by Sauer *et al.*, Siemens Corporate Research (SCR), Princeton, NJ, USA. The detailed specifications of the HMD, color and monochrome cameras are depicted in table 5.1.

The use of a video see-through HMD has two major advantages for our work: (1) Hardware based temporal synchronization of video camera images and tracking data allows temporal lag free augmentation of the real world camera images with rendered virtual images. (2) Full control of real world image data, which gives us complete freedom for optimally fusing real and virtual image data for augmentation. An optical see-through HMD system does not allow for the implementation of the techniques described in section 5.4.1 to merge real and virtual images per pixel. Furthermore, an optical see-through system will prohibit temporal synchronization of virtual rendered image data and real world scene, and will result in a perceivable lag between real and virtual scene.

	RAMP system
HMD	Kaiser Proview XL35
HMD resolution	XGA: 1024x768px, 2.34 arcmin/px
HMD FOV (h,v)	40°, 30°
HMD color range	True-Color, 24bit
HMD contrast ratio	40:1
Color cam.	GP-KS 1000, Panasonic
Color cam. resolution	640x480 px (interlaced, analog)
Color cam. update rate	30 Hz
Lens	M-Mount 8mm
Tracking cam.	XC-77RR, Sony
Tracking cam. resolution	570x485px
Tracking cam. update rate	$>$ 30 Hz
Hmd + Camera Weight	$\sim$ 1 kg
Max. system frame rate	30 Hz
Min. System Latency	$\sim$ 100 ms

Table 5.1: HMD, video camera, and tracking system specifications of the used system.

5.3.2 Tracking, Coordinate Systems and Calibration

The optical inside-out tracking of the RAMP system is used for estimation of the HMD position and orientation with respect to an arc shaped optical tracking target (see figure 5.3). In inside-out tracking, the sensor (in our setup a single monochrome camera sensitive to infrared light) is moving and its position and orientation is estimated with respect to a non-moving tracking target. In outside-in tracking approaches the principle is reversed. The sensors (cameras) remain static and the tracking targets are moving within the tracking volume. Compared to outside-in tracking, inside-out tracking provides higher rotational accuracy for estimation of the HMD position and orientation in viewing direction, resulting in less offset in the augmentation [74]. However, the advantages of optical outside-in tracking systems are the much larger possible tracking volume covered by the cameras. The tracking error in outside-in tracking is a function of camera baseline and distance of the tracking target to the camera baseline. Larger camera baselines allow the use of smaller and more convenient tracking target configurations. Especially for tracking of surgical instruments and tools the use of a outside-in tracking is frequently the method of choice because of the mentioned advantages.

In our setup we combine the advantages of both approaches by using an outside-in optical tracking system consisting of four ART ArTrack2 [1] cameras for tracking of all objects inside the tracking volume in addition to the inside-out tracking. The outside-in is capable of precise localization of all tracked instruments, phantom, and the RAMP arc tracking target. The cameras are mounted on a frame above the workspace, covering a large working/tracking volume.

The RAMP arc (see figure 5.3) is the common tracking target for both optical infrared tracking systems. It acts as the reference coordinate system (ARC) for the whole AR scene, to transfer transformations from the outside-in tracking system to the inside-out tracking system. For augmentation the transformations of all tracked objects have to be known with respect to the HMD coordinate system. For a tracked object, e.g. a surgical instrument, tracked by the outside-in tracking system we compute the transformation ${}^{Obj}\boldsymbol{H}_{HMD}$ from HMD coordinate system (HMDc) to object tracking coordinate system (Obj) by

$${}^{Obj}\boldsymbol{H}_{ARTc} * {}^{ARTc}\boldsymbol{H}_{ARC} * {}^{ARC}\boldsymbol{H}_{HMDc} \tag{5.1}$$

Where ${}^{ARC}\boldsymbol{H}_{HMDc}$ is the transformation from HMD coordinate system to the RAMP Arc (ARC) coordinate system given by the inside-out tracking system. ${}^{Obj}\boldsymbol{H}_{ARTc}$ denotes the transformation from the ART coordinate system (ARTc) to the coordinate system (Obj) of the tracked object and ${}^{ARTc}\boldsymbol{H}_{ARC}$ denotes the transformation from ARC to ART coordinates. ${}^{Obj}\boldsymbol{H}_{ARTc}$ is directly given the outside-in tracking system and ${}^{ARTc}\boldsymbol{H}_{ARC}$ is computed by ${}^{ARC}\boldsymbol{H}_{ARTc}{}^{-1}$ given by the outside-in tracking system.

Calibration: Before the view inside the HMD can be augmented with virtual representations of real, tracked objects often the additional step of calibration is necessary to determine the calibration transformation from object tracking coordinates to object

---

[1] http://www.ar-tracking.de

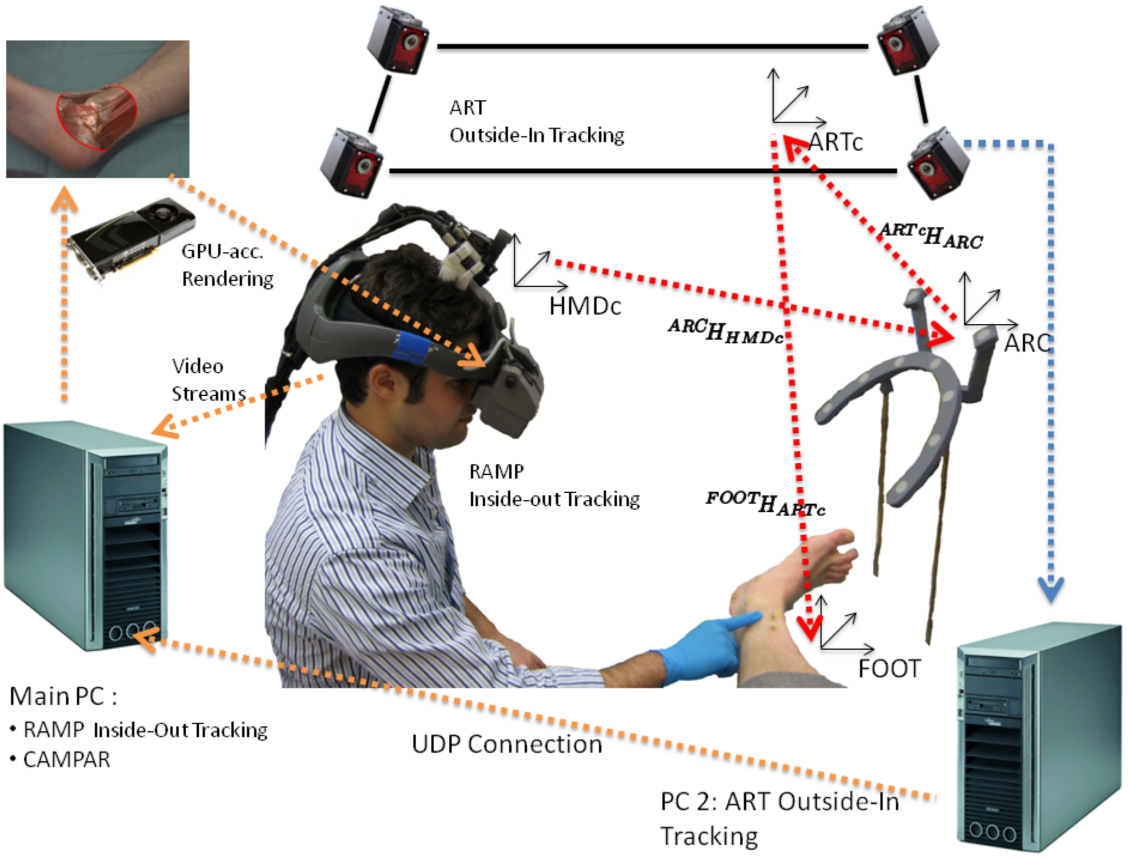


Figure 5.3: Used setup for AR system. User is wearing the HMD with the inside-out tracking system and observes the scene. Outside-in tracking system is mounted over the scene tracking all objects (ARC, Foot) within tracking volume. All transformations are for augmentation are depicted by annotated red arrows.

model coordinates. The calibration has to be performed every time the tracking coordinate system changes, e.g. by changing the tracking marker pattern layout or its position with respect to the real object. Within this work several methods are used to determine the missing transformations.

- *Pointer calibration:* Tracked pointers are often helpful in determining landmark positions in tracking coordinates as a first step for landmark/point based calibration methods. Furthermore, many minimally invasive medical tools are essentially long rigid cylinders and can be treated as pointers during calibration. Pointer calibration, also known as hot-spot or stylus calibration, works by fixating the pointer tip (in general in a small pit) and moving the rest of the pointer about the fixed tip. During pointer movement the pose readings from the attached tracking target are stored. For a robust and accurate calibration the tracking target movement path should cover a hemisphere. The pointer tip position in tracking coordinates is then determined by fitting a sphere to the recorded pose readings. Iterative [50] and closed form [94] solutions can be used.

- *Point-based calibration:* For calibration of CT volume data coordinate frames to tracking coordinate systems we use point-based registration. Prior to acquisition of the CT data markers are attached to the anatomy of interest. We use 4[mm] Beekley [2] CT-spots fiducials. The fiducials are additionally coated with an infrared reflective foil before attaching them to the surface of the subject. Thereby, the fiducials are detectable by the optical tracking system. The marker positions in CT volume coordinate frame are extracted using an automatic segmentation method. The CT and tracking space position of the markers are the used to compute the rigid transformation from CT volume to tracking space using rigid point-based registration (For more details see [207]). Skin are markers are however only possible on rigid anatomy, e.g. in phantom studies. For clinical introduction the concept of dynamic reference base i.e. markers attached to the target (fixated to the bone) anatomy has to be used.

- *Image data based calibration:* If no combined markers or no markers at all were attached to the subject during CT volume acquisition, the data sets have to be registered by other means before augmentation. For our phantom experiments (see Section 5.5.1.1) we therefore attached fiducials to the surface of a phantom (see section 5.3.3) and acquired a high resolution CT scan as the original human CT dataset was acquired without CT fiducials. The phantom CT data was then cropped and split into two datasets, one for the phantom head and for the torso. The centroids of the markers were extracted fully automatically from the data set based on intensity thresholding and moments analysis. Finally the real human CT and phantom CT datasets were registered using intensity based registration. Mutual information was selected as similarity measure and a best neighbor optimizer was used (see [63] for a comprehensive overview on medical image registration). After successful registration, the original human CT was transformed into the coordinate frame of the phantom dataset and used for augmentation.

5.3.3 Phantom

Setting up repeatable and realistic experiments for medical AR is a tedious and complex task. A main factor is the availability of a realistic phantom. Within our group a phantom based on the data of the Visible Korean Human (VKH) Project [149] was manufactured. The phantom will be denoted Visible Korean Human Phantom (VKHP) for the remainder of this chapter. The phantom was created by Bichlmeier *et al.* [12], by first segmenting skin and bone from the original VKH CT data set. In the next step, surface models were created from the segmentation results and prepared as input data for 3D printing. The surface data was then split to create assembly parts small enough to be printed by selective laser sintering. The phantom parts were then assembled and coated with skin colored powder.

The life-size phantom ranges from the head to the lower hips. The head and thorax of the phantom can be separated for independent experiments. Several adjustable operation

---

[2] http://www.beekley.com

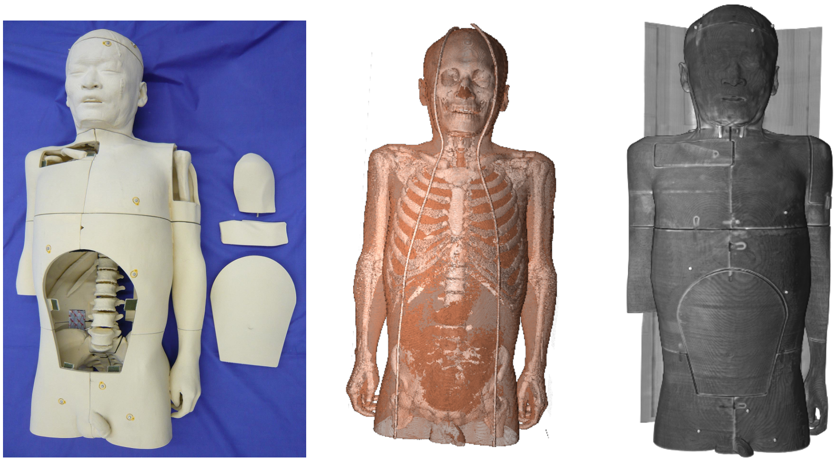

(a) (b) (c)

Figure 5.4: (a) Visible Korean human phantom (VKHP). (b) DVR of visible Korean human CT dataset used as basis for constructing the VKHP. (c) DVR of VKHP CT scan with radiopaque optical tracking fiducials.

sites have been added to the phantom for experiments and training of different surgical procedures in the laboratory. The current VKHP contains only skin and bone surface structures. However, this is sufficient for simulation and training of many orthopedic procedures. In section 5.5.2 the phantom is used for a drilling experiment in the left shoulder area. More details about our model of the VKHP can be obtained from [9, 12].

5.3.4 Software Framework

All dynamic input data, i.e. tracking and video streams, as well as all static data, i.e. volume data, model data and calibration results, are combined within the AR software framework CAMPAR [190]. CAMPAR is a modular, multiple layer, component based framework, with a small core library for the most common functionality in medical AR. It features an easy to extent API for access to various input and output devices. Input devices are grouped in categories, e.g. video input devices or tracking input devices. A set of input device classes wrapping access to the most common tracking and camera devices is included within the core library. Instances of output device classes are in general used for displaying the

	PC Workstation
CPU Memory	Dual Intel Xeon, 3.2Ghz 1.8Gbyte
GPU1 Number Shaders Memory	Nvidia Geforce 8800 Ultra (G80) 128 768 Mbyte
GPU1 Number Shaders Memory	Nvidia Geforce GTX 275 (GT200b) 240 1.792 Gbyte
Framegrabber	3 x Matrox Meteor II (analog, PCI-X)

Table 5.2: Medical augmented reality system PC workstation specification.

augmented reality scene on the HMD, single or multiple monitors, or write data to a file stream. Additional input/output devices can be created by deriving from generic base classes, adding new functionality, and creating a plugin dynamic-link library (DLL) that will be loaded by the driver application. To synchronize input and output, CAMPAR employs a synchronizer that implements a set of temporal synchronization policies. In the default case the synchronizer will wait for the slowest system component prior to updating the displayed AR scene.

For the work presented in this chapter we have developed a CAMPAR output device plugin combining the techniques presented in section 5.4 and the volume rendering techniques presented in chapter 4 of this thesis. Figure 5.3 depicts an overview of all involved components and their connections within our setup. The main PC workstation (for details see table 5.2) runs an instance of CAMPAR. On the same workstation an instance of the RAMP inside-out tracking software is executed that relays its results via UDP connection on the loopback device to the CAMPAR instance. The workstation is equipped with three analog framegrabbers, two for the per eye video camera, and one for the monochrome tracking camera. Over the time of this thesis the workstation was equipped with various GPUs. For the developed methods and experiments presented in this chapter the two GPUs presented in table 5.2 were used. On a second computer an instance of DTrack, ART's software for the outside-in tracking system is executed. The pose values are broad casted via UDP to CAMPAR on the main workstation.

5.4 Methods

The real-time medical AR visualization methods discussed in this chapter are closely related and to a large extent based on the direct volume rendering techniques present in chapter 4 of this work. Thus, I will focus only on special modifications and medical AR specific requirements for real-time visualization in this section.

5.4.1 Focus and Context Rendering for Medical AR

The methods for focus and context rendering presented in this thesis are a combination and extension of the works of Krüger *et al.* [105] and Bichlmeier *et al.* [15]. In section 4.4.3 we have described our extensions and performance improvements to the original ClearView algorithm. The optimizations were originally motivated by the requirements posed within the stereo video see-through AR environment described in this chapter. Within the AR environment we use a focus and context rendering setup always consisting of one topmost skin isosurface context layer and additional context and focus layers. During composition of the final image, we use the video ghosting techniques proposed by Bichlmeier *et al.* Instead of traditionally rendering the virtual objects on top of the video image, the focus zone is embedded directly into the video image. The transparency of the video image within the focus zone is modulated based on the skin isosurface properties and view parameters as described in section 4.4.3.

A common shortcoming of many in-situ visualization techniques in medical AR is perspective limitation. In contrast to Virtual Reality (VR) the virtual objects can not simply be rotated to reveal the backside or display the object from another point of view.

Bichlmeier *et al.* proposed to add a virtual mirror as an additional perspective within the field of view of the user [14]. In section 4.4.3.2 we have described how we combined the virtual mirror and real-time volume visualization. In this chapter it is merged with the contextual in-situ rendering for medical AR to further enhance the visual perception.

Furthermore, we have added techniques for handling occlusion of the virtual parts of the scene by the physician's hands and tracked surgical instruments. In our application, augmentation on the HMD for minimally invasive interventions, the operating site visible to the physician is relatively small and access to it is restricted mostly to the treating physician. Thus, the main and most frequent sources of occlusion are the physician's own hands and medical instruments. Cuttings *et al.* [27, 28] identified occlusion as one of the most important depth cues of the human visual system. Conflicting depth cues from incorrect occlusion of virtual and real objects in the augmented scene is a well known problem in AR. Without occlusion handling, virtual objects will appear floating over the real objects and any impression of depth gained due to other depth cues will be lost. In the following section we will explain in detail the methods for hand and surgical instrument occlusion handling.

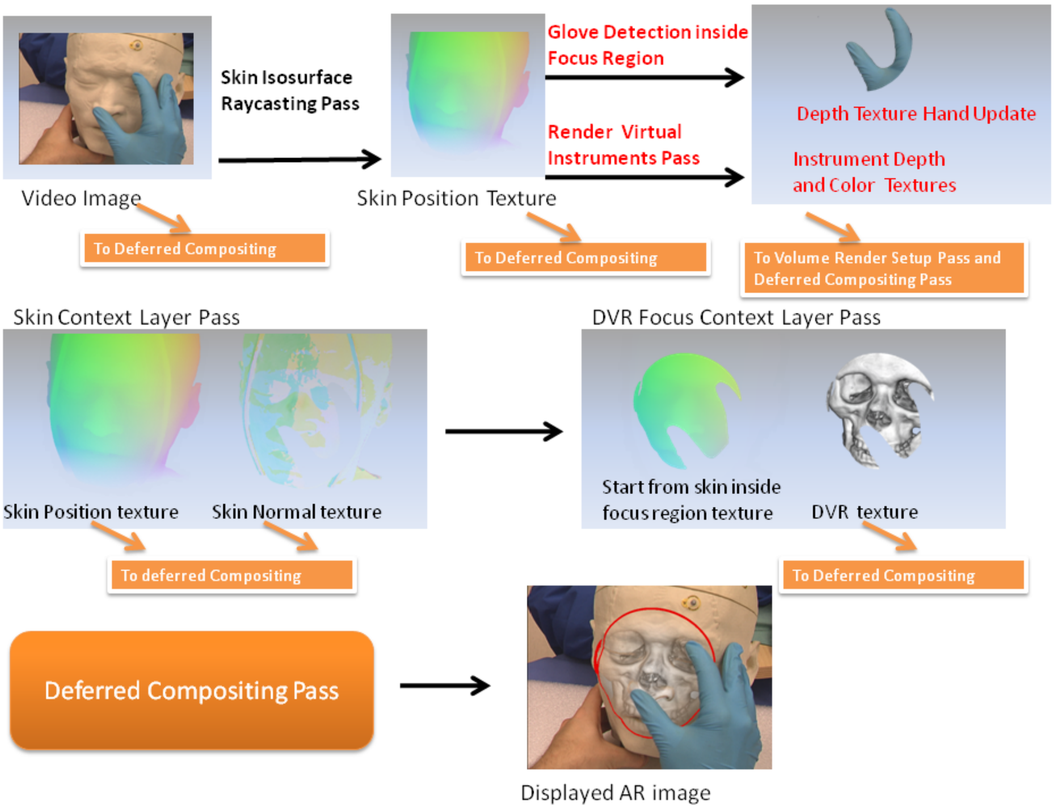


Figure 5.5: Schematic overview of basic rendering pipeline for medical AR focus and context visualization. Not depicted in this figure are the instrument rendering passes and virtual mirror passes.

Figure 5.5 depicts a schematic overview of the extended rendering pipeline for augmented reality visualization on the HMD. For clarity it does not depict the virtual mirror rendering passes, but solely an overview of the rendering passes for the main HMD perspective.

In comparison to the original ClearView pipeline [105] the order of the rendering passes has been rearranged and three additional passes have been added at the beginning of the pipeline. The skin context isosurface is rendered first. It is a required input for the following hand occlusion and tracked instrument rendering passes. Without knowledge of the skin position for the current view, one cannot determine if a fragment is inside or outside the focus region, or above or below the skin.

In the next two rendering passes the video image texture is analyzed and virtual instruments are rendered only below the skin surface to avoid depth cue conflicts caused by the virtual focus zone and real objects in front of it (see figure 5.6 and figure 5.7 for examples). Subsequently all context and focus layers are computed and all resulting temporary images are passed on to the deferred compositing pass which computes the final displayed augmented images.

In case of a virtual mirror in the scene, the virtual mirror image is rendered before the rendering the main perspective. It is rendered after the hand occlusion and directly before the virtual instrument rendering pass.

5.4.2 Handling Occlusion of Real and Virtual Objects

5.4.2.1 Hand Occlusion

The physician's hands inside the video images are detected in the second rendering pass of the rendering pipeline. Detecting and marking these hand pixels is essential for correctly merging the real video and rendered virtual images. Additionally it aids in skipping costly calculations in subsequent rendering passes for occluded pixels. In [170], Sandor *et al.* demonstrated the importance and advantages of hand occlusion handling for video see through HMDs. Figure 5.6 depicts the problem of unresolved hand occlusion.

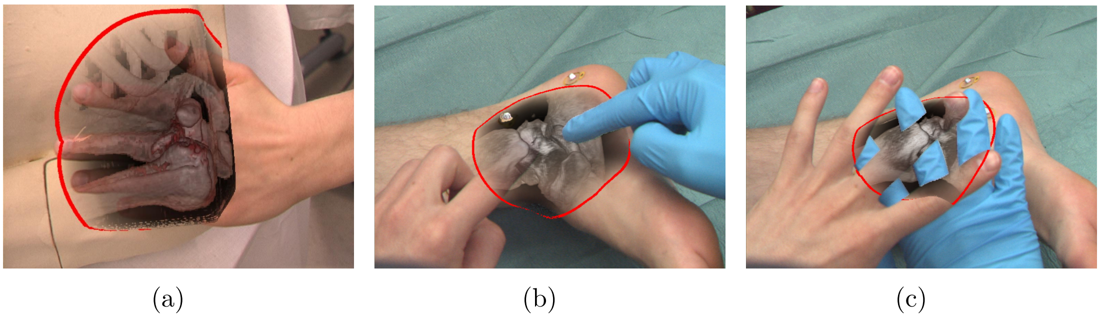

(a) (b) (c)

Figure 5.6: Examples of conflicting depth cues from unresolved occlusion of real and virtual objects within the field of view. (a) Non-handled occlusion of real hand and focus and context rendering. (b) Left hand (without glove) is not detected by the system and merged (blended) with virtual image, wheres as blue glove is correctly occluding it. (c) Same effect depicted even more drastically. The human visual system is highly confused by conflicting depth cues and expected distances.

As mentioned before occlusion is one of the major depth cues and wrong occlusion will immediately destroy the illusion of virtual objects inside real objects. However, fortunately in our setup it is valid to assume that the physician's hand will never be inside the patient. Treatment is applied via ports into the patient's body using minimally invasive instruments. Thus, the physician's hands are always above the skin surface meaning that they potentially occlude the virtual objects rendered inside the patient. The presented approach is sound for minimally invasive medical procedures and diagnostic purposes, where the physicians wear gloves which can be detected reliably in the video images and there is no blood on the gloves that would disturb the hand segmentation algorithm.

The use of white latex gloves is common in a medical environment. A stable separation of a physician's hands from the rest of the video image is usually not feasible due to large similarities of the gloves color and other features in the operating site image, e.g. patient skin. Our to the problem solution is simple, yet effective and inexpensive. Exchanging the white gloves with colored gloves allows foreground background segmentation using a simple color thresholding based image filter. Blue anti-allergic gloves for instance, are less commonly used but available in most clinics. The color filter is implemented as a fragment shader that compares color hue to a reference color where value and brightness are not too low for hue to be reliable. Because of natural shading, it is insufficient to compare only RGB color values. Brightness may vary largely, so conversion to HSV is worth the additional effort and gives much better results. The results of the hand segmentation shader are written to a depth render target, where every foreground (hand) pixel is set to the minimal depth value. Thereby, costly operations are avoided by the Depth-Test of the GPU for every marked pixel for all subsequent rendering pipeline stages.

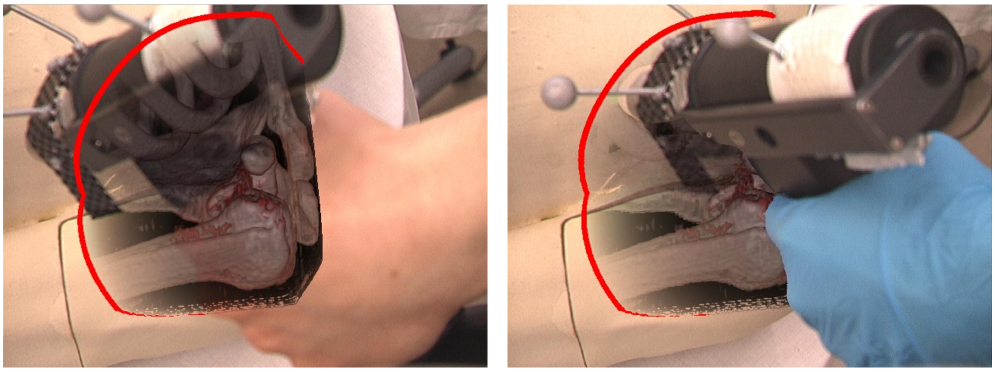
Compared to our previous work [109] we have further improved the stability of the hand segmentation rendering pass. The refined method restricts processing to pixels inside the focus region by incorporating information from the first skin isosurface pass result texture. Thereby, processing load is reduced and segmentation artifacts for pixels outside the focus region are avoided. Due to the smaller number of fragments to process, the color test is applied to a region of 3×3 fragments centered on the current fragment for a more reliable segmentation results and increased robustness to local brightness variations.

5.4.2.2 Instrument Occlusion

A second source for occlusion problems of real and virtual objects are surgical instruments in the field of view. Figure 5.7(a) depicts the problem arising when moving a surgical drill in front of the virtual object embedded inside the phantom in the AR view.

In the intended application of our system in minimally invasive surgery, instruments are inserted into the patient's body through natural or artificial openings (ports). It is therefore desirable to visualize the instrument parts outside of the body by copying the corresponding pixels from the video camera image to the framebuffer and only render the parts inside the human body with virtual models. In [46], Fischer *et al.* presented an instrument occlusion handling method for medical AR based on off-line volume segmentation and on-line first hit isosurface ray-casting. The integration and extension of their approach to real-time occlusion handling in our scenario is straight forward.

Prior to the main context and focus layer rendering passes a virtual model is rendered



(a) (b)

Figure 5.7: (a) Occlusion problem caused by surgical instrument in field of view. (b) Results of our instrument occlusion handling method.

for every tracked surgical instrument present in the scene into depth and two color rendertargets. These rendertargets are passed as input textures to the main rendering pipeline. The depth texture is used by the volume renderer when computing the ray start and end positions prior to the first ray-casting pass.

Both color rendertargets are passed to the deferred compositing pass. The first one contains a copy of the current video frame texture for every fragment located above the skin surface. The second one contains the rendering of the virtual instruments below the skin surface. In the instrument rendering pass depth values of the virtual instruments are compared with the skin surface depth values computed from the skin surface position texture of the first rendering pass in the pipeline.

If the depth value is smaller than the one for the skin isosurface, the instrument is located above the skin surface. The value from the current video frame texture is copied and written to the first color render target. The value in the depth rendertarget is set to 0.0 (near plane). Thus, no further computations will be executed in the following volume rendering pipeline stages for this pixel.

If the depth value is greater than for the skin isosurface, it means that either the instrument is inside the volume or located behind it. The depth value of the virtual geometry is written to the depth rendertarget and the current instrument color is written to the second color rendertarget. Thereby, only the inside the volume embedded parts of the instrument are rendered to the second color render target, and stored for the deferred compositing pass. When computing the ray end position the volume renderer will clip the rays to stop at the instrument if the instrument depth value indicates it is located inside the volume.

The presented occlusion handling techniques have the additional benefit of improving the overall rendering performance. Located at an early stage of the pipeline, the depth masking in fact disables the rendering of the volume for any subsequent render passes in the pipeline. Thereby, computationally expensive techniques are in fact never performed for the occluded parts of the image and it contributes towards high-quality, real-time rendering solutions.

5.4.3 Optimizing for AR environment

The requirements for volume rendering in medical augmented reality differ significantly from those for visualization on medical workstations, e.g. for diagnostic purposes. The renderer has to be optimized for speed first, followed by image quality second. The visualization should at no point result in noticeable latency of the overall system. This task is in general difficult to achieve as the performance depends on many parameters. Volume data size, volume visibility, render target resolution, rendering technique, and rendering parameters all largely affect the average frame rate of the renderer.

Additionally, for in-situ visualization with an HMD as in our setup, the renderer has to be capable of perspectively correct stereo rendering in real-time (30 fps). To achieve this goal we used many of the speed and quality optimization techniques described in chapter 4. Namely empty space skipping, early ray termination, pre-integration in combination with reduced sampling rate, ray start offsets, and deferred shading.

Furthermore, we found that efficient use and re-use of GPU memory is essential in order to achieve high rendering performance for the stereo rendering on the HMD. This is especially important for the focus and context rendering, because of its use of many temporary textures on the GPU to store intermediate results, e.g. surface positions, surface normals, mask buffers, for deferred shading. Simply using two separate instances of the renderer for each eye results in twice the memory compared to single eye rendering. Simple focus and context rendering, one isosurface context layer and one DVR focus layer, can easily consume $\sim$ 100 Mbyte for XGA resolution 2D textures. Including GPU memory for volume data, GPU programs, geometry vertex and index buffers, the required memory can rapidly reach the physically available memory of modern GPUs.

To avoid such situations we used interleaved rendering passes for generating the left and right eye images on the GPU. Only the final result image of the first eye rendering pass is stored, all other temporary textures are reused when rendering the second eye image. Based on our experience and experiments interleaved rendering is the fastest method for stereo rendering (see section 4.4.2).

Reducing the rendered image resolution is another point one can consider for optimizing performance both directly (ray-casting) and indirectly (smaller memory footprint). As discussed in section 4.3.2, the differences between full resolution rendered and displayed images and images rendered with slightly reduced resolution and only upscaled to full display resolution are often not even detected by the human visual system. Especially for moving scenes, these differences are even harder to detect. Considering our setup, two small LCDs inside the HMD, and a constantly changing viewing frustum, it is valid to raise the following question: *Which level of quality differences can be perceived by the user of the system on the HMD?*

From a performance point of view, it makes no sense to render high resolution images if the same subjective user perception can be achieved with slightly reduced image resolution, resulting in a higher rendering frame rate. Thus, it is valid to consider reducing image resolution, to improve performance or give room for integration of more costly rendering techniques, e.g. shading and local illumination, to raise the perceived realism of the virtual scene. In the current setup of the system we have restricted rendering resolution to the video camera resolution. For display on the HMD LCD (XGA resolution) the images are upscaled using bilinear texture interpolation. We have tested rendering with full XGA resolution, however the differences were not noticeable on the HMD display by the human observers.

5.5 Experiments and Results

The described methods have been evaluated both in terms of performance and their benefit for image-guided navigation tasks. The performance was evaluated continuously, during development and integration of the methods into the medical AR system, in multiple experiments for phantom and in-vivo augmentation, using different GPUs over time. The benchmark results (see section 5.5.1) clearly show over time improving GPU performance resolved previously system-performance limiting rendering techniques.

In a second experiment we evaluated the drilling accuracy of six surgeons using the system in a realistic bone drilling experiment on the visible Korean human phantom (see section 5.3.3). The surgeons were asked to complete the drilling task multiple times using different in-situ focus and context visualization modes, without and with the help of the virtual mirror. Drilling accuracy was evaluated based on the drill tip end position computed from the tracking system values. Additionally a questionnaire was completed by the surgeons to evaluate their perceived levels of demand during the task. The detailed experiment description and results are presented in section 5.5.2.

5.5.1 Performance Benchmarks

Measuring the performance of the whole system running under real conditions, testing several rendering methods and parameter sets, is crucial to estimate the overall performance. Rendering is only a small part of the whole system, however with a significant impact on the total system performance. If rendering does not complete within less than the update time of the slowest system component, the synchronizer will be forced to drop frames. See figure 5.8 for an example where every fourth frame is dropped due to a too slow visualization component.

In the depicted example every 30 ms a new image is generated by exposing the camera for 30 ms (Camera), then 25 ms are needed to transfer the image from frame grabber to host memory and compute the tracking information (Tracking). Immediately afterwards the system tries to render the image and to display it on the screen (Visualization), consuming additionally 40 ms. If at that time the visualization of the last frame has not finished, the visualization of the current frame has to wait (Idle). This results in a minimum latency of 95ms from generation of an image until its final display by the system. Incorporating frame idle times, the individual frame latency can even be higher.

The combination of camera update rate and component computation times decreases the maximum possible frame rate from $\sim$ 30 fps to $\sim$ 25 fps, and additionally results in the

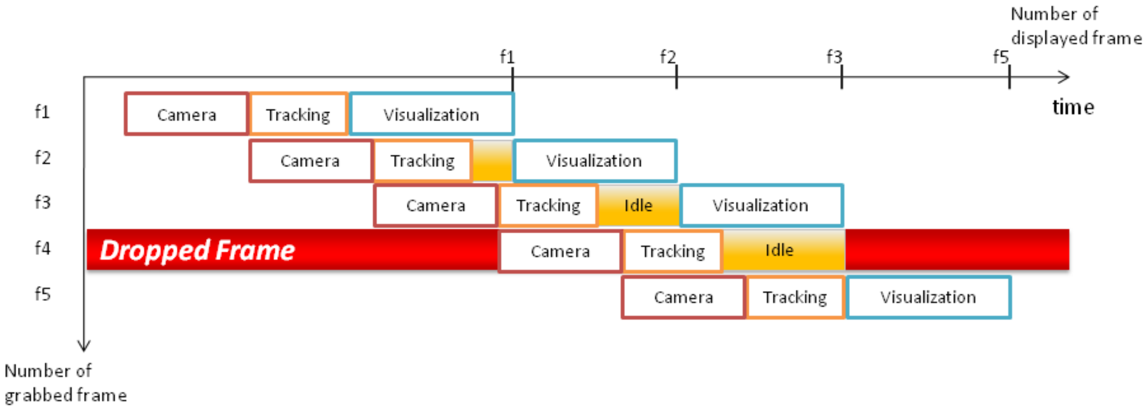


Figure 5.8: Example frame sequence diagram. Camera consumes 30 ms, Tracking 25 ms, and Visualization 40 ms. This setup results in varying individual frame latency and skips display of every fourth image frame.

loss of every fourth frame. After rendering the first three frames, the data synchronizer drops frame four as frame five is already present in memory and is more recent. Starting with frame six the process repeats itself.

We thoroughly evaluated the performance in a series of experiments measuring the system update rate for different rendering modes by augmenting a realistic phantom with and a living's person foot with real human CT data. For all the experiments we used the video-see through HMD described in section 5.1, and a standard PC workstation with two different GPUs (for details about the used hardware see table 5.2). For every timing the used GPU is reported in result tables. As rendering performance is very dependent on the current view of the cameras onto the to be augmented objects, we measured both average system update rate over a fixed period of time, as well as absolute time per frame for extreme viewpoints. For the evaluation of average system update rate a volunteer wore the HMD and inspected the phantom by moving continuously moving around it for two minutes, while the frame rate was measured using Fraps,[3] a freely available benchmarking tool.

5.5.1.1 Phantom Experiments

The performance of the renderer was evaluated in detail for multiple rendering modes on the phantom for three CT datasets from the visible Korean human project: (1) Head - 296x320x420 and (2) Thorax - 256x368x522. All datasets stored data with 8bit. For the performance measurements the internal render target resolution was set to the camera resolution (640x480), the OpenGL viewport resolution was set to the per eye HMD resolution (1024x768). Full screen rendering was enabled. Figure 5.9 depicts six exemplary images generated during the experiments. The results of the experiment for different

---

[3] http://www.fraps.com

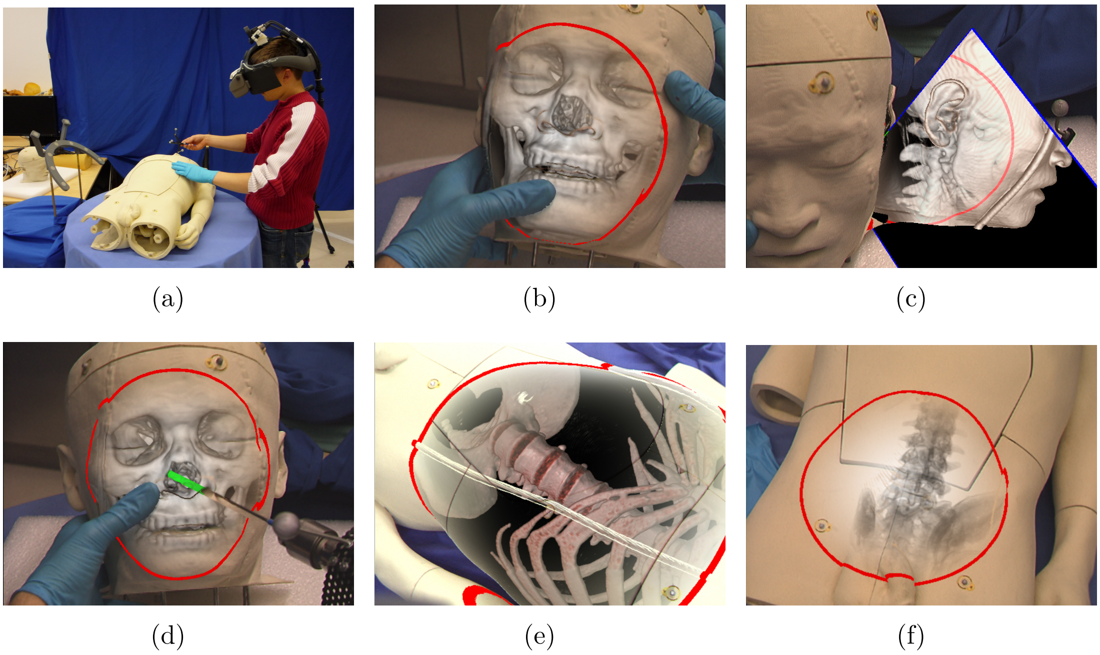

(a) (b) (c) (d) (e) (f)

Figure 5.9: (a) Experiment setup. User is wearing the HMD, holding a tracked pointer in his hand which is used to control virtual mirror (see (c)). (b-f) Focus and context renderings of VKH head and thorax datasets. (b) Hand Occlusion handling, (c) Virtual mirror view of lower head and neck region (d) Insertion of virtual drill and correct hand and instrument occlusion handling (e) Shaded volume rendering inside large abdominal focus region (e) Virtual X-ray (DRR) rendering of lower spine and hip focus region.

render techniques and extreme viewport are described and depicted in the items of the following listing.

- *Conventional DVR:* Compared to our work presented in this chapter DVR presents the old, from our perspective outdated way to visualize volumetric data in a medical AR environment. Traditional DVR comes with all the problems and shortcomings of direct, non-adapted visualization for AR, e.g. floating effect, occlusion. However, DVR is currently used in many systems and for comparing the current standard with our new methods it is important to establish a baseline for comparison of the performance of the two different approaches. The performance of the following four render modes has been evaluated: (1) DVR - DVR with standard 1D post-classification (2) same as (DVR) with added local illumination, (3) PreInt - Volume Rendering with pre-integrated lookup table (4) same as (PreInt) with added local illumination. Local illumination always used the Blinn-Phong model with on-the-fly gradient evaluation using central-differences.

- *Focus and Context Rendering:*

  In contrast to DVR performance of focus and context rendering benefits from the optimizations we added to the original ClearView algorithm (for details see section

Traditional DVR augmentation:

	Head		Thorax	
Rendermode	GPU1	GPU 2	GPU1	GPU2
DVR	15	> 30	12	18
DVR Shaded	13	> 30	5	8
Pre-Int	14	27	10	18
Pre-Int Shaded	12	13	5	7

Table 5.3: Conventional direct volume rendering performance results for visible Korean human datasets 1-3. Average frames per second, estimated over 2 minutes usage.

4.4.3). By only rendering the virtual objects inside the focus zone a huge number of calculations compared to conventional DVR is removed. However, the algorithm comes with the costs of more control program overhead, due to the multi-pass rendering of the individual context and focus layer, and their deferred shading and compositing steps. Table 5.4 depicts the average system update rate using focus and context rendering with one context layer (Skin), using the video embedding technique described in section 5.4.1 and one focus layer rendered with DVR. The DVR modes were the same as in the performance evaluation for traditional DVR rendering (see table 5.3).

Focus and Context Rendering:

	Head		Thorax	
Focus Layer	GPU1	GPU 2	GPU1	GPU2
DVR	23	> 30	17	26
DVR Shaded	20	> 30	9	12
Pre-Int	22	> 30	16	25
Pre-Int Shaded	20	28	9	16

Table 5.4: Focus and context rendering performance results for visible Korean human datasets 1-3. Average frames per second, estimated over 2 minutes usage. A video iso-surface context layer is always rendered and paired with the same DVR modes as in the conventional DVR performance evaluation.

- *Focus and context Rendering with Virtual Mirror and Instruments:* Adding additional render passes, especially the complexity of the virtual mirror, result in an increased computational strain for the rendering pipeline. The following experiments evaluated the performance of our method for the most demanding settings to see if usable system update rates can still be retained under this conditions. One virtual mirror attached to a tracked instrument, and a model of a tracked drilling machine were added to the scene and the measurements conducted in the previous experiments were repeated. We evaluated the same rendering modes as in the pure focus and context rendering experiment to ease the comparison.

Focus and Context Rendering + Virtual Mirror:

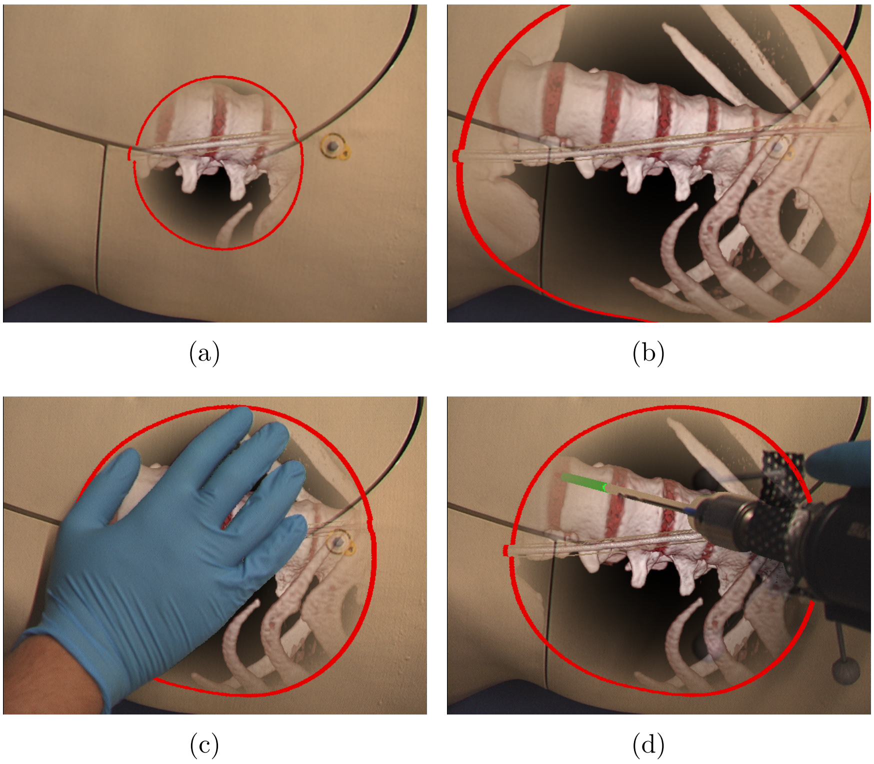
	Head		Thorax	
Focus Layer	GPU1	GPU 2	GPU1	GPU2
DVR	24	> 30	11	25
DVR Shaded	20	> 30	7	12
Pre-Int	21	> 30	10	25
Pre-Int Shaded	20	28	6	15

Table 5.5: Performance results of focus and context rendering for visible Korean human datasets 1-3. Average frames per second, estimated over 2 minutes usage. A video isosurface context layer is always rendered and paired with the same DVR modes as in the conventional DVR performance evaluation. Additionally one virtual mirror and one surgical instrument model.

- *Individual timings for extreme viewpoints:* The performance of our rendering algorithm is largely dependent on the selected view onto the data, render mode, and individual rendering parameters. With the following images we want to give the reader an impression of the cases that increase and decrease performance most drastically.

  All the images depicted in figure 5.10 represent extreme cases that may occur in real situations. Small focus regions (see figure 5.10(a)) are well suited to rapidly scan a large volume without any performance penalty, e.g. like X-ray vision into the patient. In typical scenes, the frame rate never dropped below 30 fps for this settings. Large focus regions (see figure 5.10(b)) will reveal global internal structures, yet providing contextual information. This setting will most probably be used very rarely, as performance decreases significantly due to the large amount of pixels/rays to process. Performance can even be worse than for conventional DVR due to the additional context layer and deferred shading computations. For this example the average frame rate was about 10 fps on a Nvidia Geforce 275 GTX. Our additions to handle occlusion of real instruments (see figure 5.10(d)) and the physicians hands (see figure 5.10(c)) do not only increase the perceived realism of the scene and retain depth cues, but do not decrease performance in worst case, but in many cases actually increase performance. Average frame rates for the depicted settings never went below 25 fps, compared to 20 fps without occluding hands or instrument. Of course the number of occluded pixels relates directly to the gained performance.

  Last but not least DVR focus layers pose one of the most difficult to handle performance impact problems. Dependent on the combination of dataset and selected classification table the performance can reduce dramatically. This is due to the disabled empty space skipping optimization for the default case rendering of DVR layers in our implementation. DVR layers use the hit position texture of the previous isosurface as ray start position texture. Off course the skin is in most cases a rather bad proxy geometry for empty space skipping in DVR, e.g. when rendering only bone structure or vasculature. We encountered this problem especially for the


(a) (b)

(c) (d)

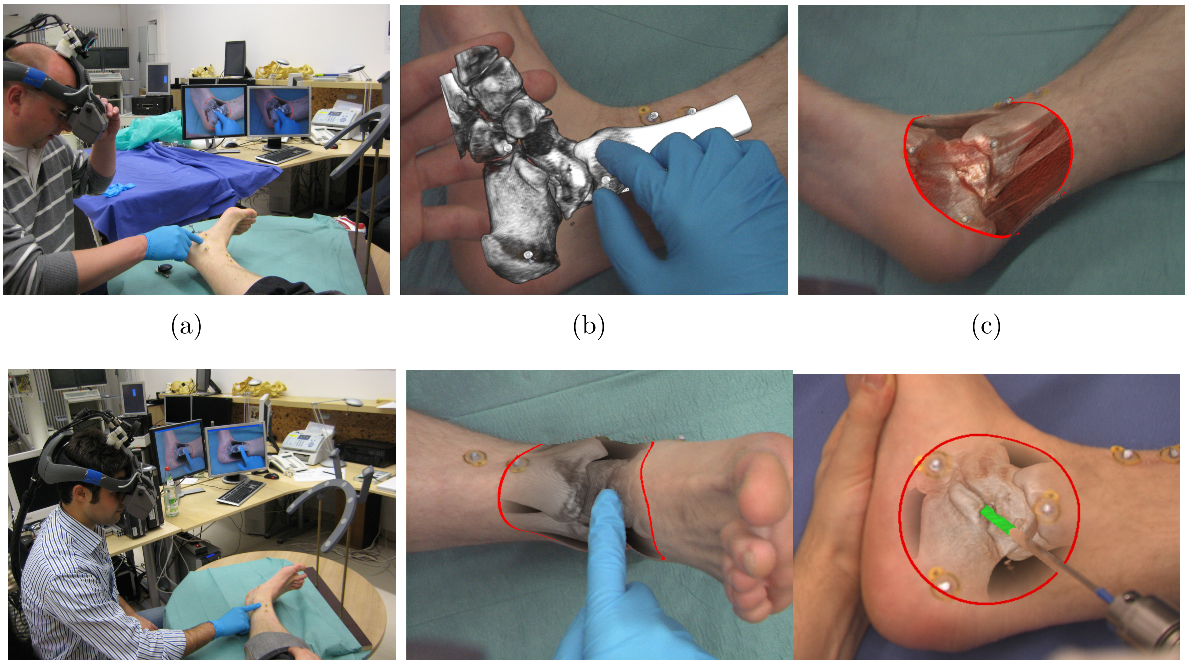
Figure 5.10: Render modes and parameters affecting performance directly: (a,b) Focus Region Size: ray-casting scales mainly with number of rays to compute, thus (a) small focus region size increases performance, while (b) large region results in reduced performance. (b) Performance of DVR focus layer rendering is largely transfer function and data set dependent. If rays are not terminated early, performance is mainly sampling rate dependent. To circumvent this one can use adaptive sampling or empty space leaping in the ray-casting shader. (c,d) Additionally, early culling of computations by masking pixels either by (c) hand or (d) instrument occlusion improves performance relative to the number of masked pixels.

visible Korean human thorax dataset (see figure 5.10(b)). For this configuration rays starting from the skin isosurface frequently have to be traced to far away bone structures, e.g. vertebrae or distal rips. As no empty space skipping is applied by default, many needless samples are performed, reducing the average frame rate to about 15 fps. As a remedy for this problem we integrated adaptive sampling based on empty space leaping inside the volume using a 3D active cell lookup table on the GPU. The table is essentially a binary volume, where every voxel corresponds directly to one octtree cell or brick of the proxy geometry that would be used for conventional empty space skipping. Using this optimization the frame rate is increased to 25 fps.

5.5.1.2 In Vivo Experiment

The visualization system was also evaluated in-vivo with the foot of one of our colleagues who volunteered for the experiment. After a sport accident, radiopaque markers had been positioned on his right foot before a CT scan, using locations that could be identified later on for re-attaching tracking markers. By attaching optical tracking fiducials at these distinct anatomical landmarks the three dimensional CT dataset with the CT fiducials at the same relative positions can be registered to the real foot with ease. In the experiment we evaluated the performance and quality of the rendered image for a CT volume of 256x256x154 voxels @16 bit, and as well the hand occlusion implementation.

Figure 5.11 depicts some of the results we got for the dataset of the foot with shaded direct volume rendering and enabled focus and context rendering.



(a) (b) (c)

(d) (e) (f)

Figure 5.11: (a, d) Trauma surgeons inspecting the foot through the HMD. (b) Volume Rendering of bone, with visible Hand Occlusion problem. Hand without glove is beneath rendering. Hand with glove is placed correctly in front. (c,e,f) Focus and Context Rendering using shaded volume rendering for focus layer. (f) Virtual drilling into ankle joint.

The rendering resolution was set to the camera resolution (640x480), the framebuffer resolution was set to the HMD per eye resolution (1024x768). Empty space leaping and early ray termination were activated, the maximum number of samples along each ray was set to 384. Frame rate was measured using Fraps. For the selected parameters the average frame rate was 30 fps, and never dropped below 28 fps, neither for standard volume rendering nor for focus and context rendering.

In experiments the renderer was also initially tested by two members of the surgical staff of the trauma surgery department. After the experiments they provided us with valuable feedback considering performance, quality and possible future improvements. Especially the stable real-time performance and high quality of the renderer for several different anatomies, head, torso and the successful in-vivo experiment on the foot motivate further studies initiated by our clinical partners.

5.5.2 Quantitative Drilling Experiment

In the previous experiments we evaluated the performance and visual quality of the developed methods. In this experiment the system was used for the first time to support a simulated surgical intervention on the visible Korean human phantom. For the experiment we chose the drilling of fixation screw channels in the humerus. Drilling of such fixation screw channels is encountered in many orthopedic interventions, mainly for fixation of bones by implants, e.g. titanium plates, which are fixated by numerous screws affixed to the bone. In most cases no complications arise if one or more screws perforate both bone cortices. However, periarticular fixation screws should never protrude from the bone into the joint surface to prevent lesions of the joint.

Figure 5.12 depicts an X-ray image of the shoulder joint after fixation plate and screws placement, with one fixation screw protruding from the humerus bone into the shoulder joint. Such protruding screws can cause various complications for the patient, from simple pain during joint movement, restricted and limited joint articulation, inflammations of the joint, to even early development of osteoarthritis. Thus, the correct positioning of the screw channels and correct drilling depth is essential to avoid later on complications. Up to today, the drilling of the fixation screw channels is done manually by the surgeon and is mainly based on his experience, knowledge from pre-operative imaging data, and his/her intuition about drilling depth and direction. To improve drilling direction stability a guiding sleeve can be mounted to the locking plate. However, estimation of the drill depth and screw length is still left to the surgeon's experience and intuition. Varying or locally weak bone density can furthermore impact the surgeon's perceived drill depth. Currently, X-ray imaging is the only means to intra-operatively control the fixation screw placement. However, for validation it requires cumbersome manual re-positioning of the X-ray device for every screw to correctly capture the screw depth in the X-ray projection

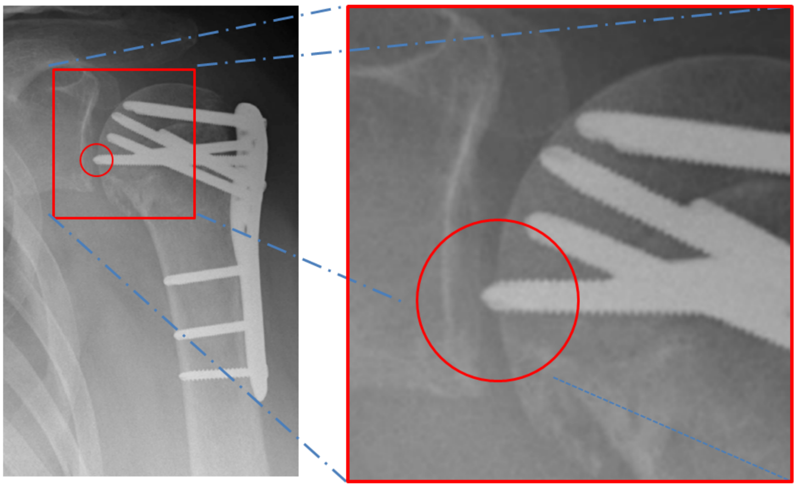


Figure 5.12: X-ray image depicting one fixation screw perforating the shoulder joint surface.

image. Therefore, joint perforation due to wrongly estimated drill depth is one of the most well known complications in orthopedic surgery.

Novel intra-operative tracking and visualization techniques offer a solution for the navigation problem the surgeons are facing. One of the main challenges is the perspective limitation during the drilling procedure. The surgeon essentially can control drill direction very well based on the visual feedback, however visual feedback of the drill depth is not possible. Adding a second perspective, e.g. visualizing the drill progression through a virtual bone model, or graphically or numerically displaying estimated drill distances can aid to improve the drill depth accuracy. In this experiment we add a second perspective, directly into the field of view of the surgeon, by using a virtual mirror. We hypothesize that using a virtual mirror improves the drilling depth accuracy compared to traditional single perspective in-situ visualization. Additionally, we assume that mirror-based visualization modes will result in less perceived frustration, but a higher mental, physical, and temporal demand during task completion. In the following experiment these assumptions are validated.

5.5.2.1 Experimental Setup

For the drilling accuracy evaluation experiment, a drilling site at the left should of the visible Korean human phantom (see section 5.3.3) was prepared.

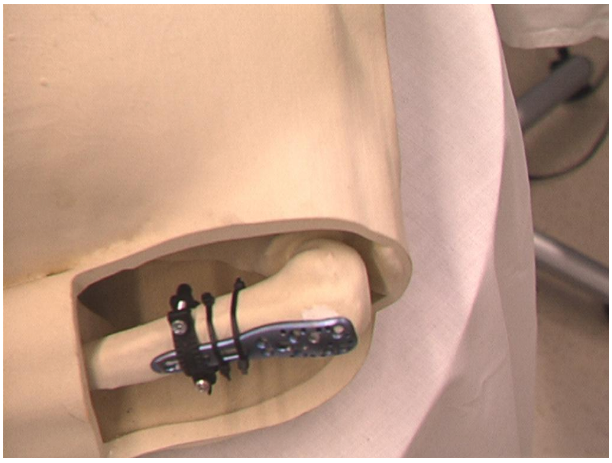


Figure 5.13: Phantom experiment setup.

A fixation plate (Philios plate from Synthes Solothurn, Swiss) was mounted on the left humerus (see figure 5.13). The plate is routinely used for fixation in trauma surgery and equipped with several threaded screw holes. To avoid permanent damage to the phantom and enable multiple repeatable drilling runs with the same direction, a drilling sleeve was mounted onto the plate. The sleeve can be filled with a *bone density mimicking* material with varying filling heights to simulate different drill depths. After evaluation of several materials, wax was chosen, which we found to give the best haptic feedback during drilling. In the experiment the virtual drill is extended by the sleeve's filling height to compensate for the additional material inside the sleeve. Thus, when the drill tip touches the wax surface inside the sleeve the virtual drill tip touches the bone in the AR visualization seen by the surgeon. Additionally the virtual drill model is extended by a safety margin, represented by 1cm long red cylinder in the experiment, to prompt the surgeon to stop before he perforates the bone's outside cortex.

Six surgeons took part in the experiment. Five out of the six participating surgeons were very experienced, having at least 3 years of experience in surgery and having performed more than 50 surgeries in the last year. Beside one participant, all surgeons had

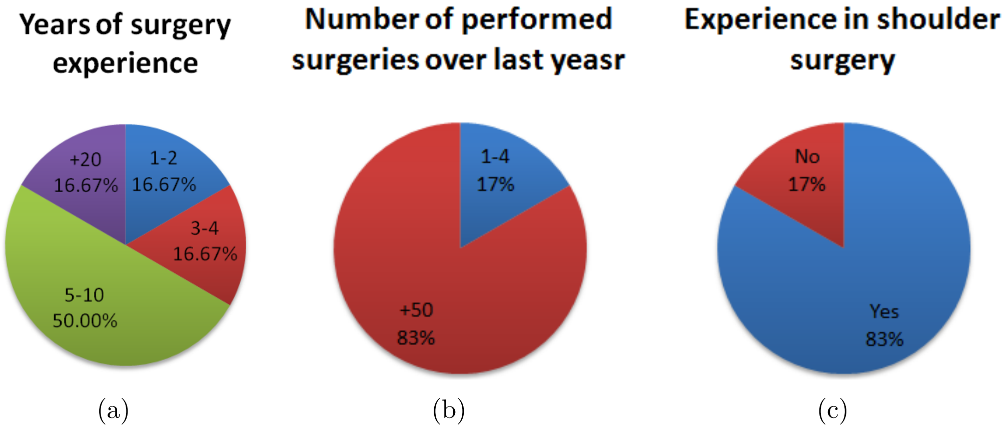
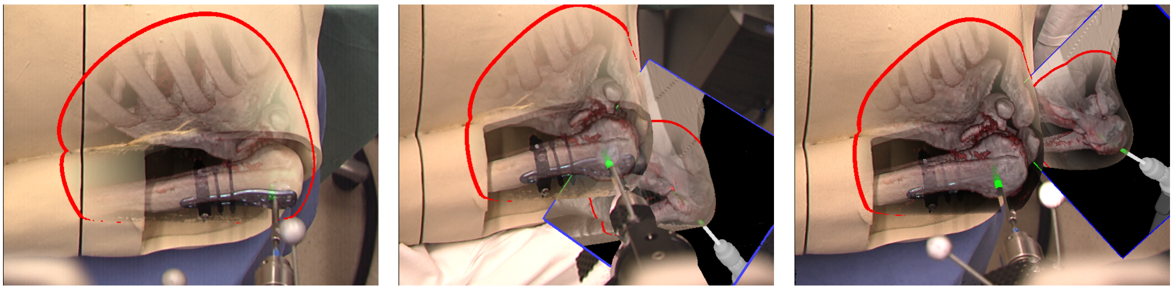


Figure 5.14: Experience of participating surgeons.

prior experience in shoulder surgery. For details about the background experience see figure 5.14. In the experiment the surgeons had to complete the drilling task on the phantom three times for each of the three different visualization modes (see figure 5.15) described in the following listing.



(a) (b) (c)

Figure 5.15: Focus and context rendering augmentation with different virtual mirror modes. (a) Without virtual mirror. (b) Fixed virtual mirror. The position and orientation are fixed prior to the drilling procedure by the surgeon. (c) Dynamic, instrument aligned virtual mirror.

1. *Regular focus and context rendering:* This mode used contextual in-situ visualization with one video context layer and a DVR focus layer (see figure 5.15(a)). A transfer function depicting the skeleton and bone structures inside the focus layer is selected. Hand and instrument occlusion are enabled. A virtual model of the drill is used to only render the parts of the drill beneath the skin surface.

2. *Focus and context rendering with fixed virtual virtual mirror:* This mode (see figure 5.15(c)) adds a fixed virtual mirror into the field of view of the surgeon. Otherwise the rendering parameters are the same as for mode 1. As the position and orientation of the mirror is fixed, the surgeons first have to position the virtual mirror inside

the workspace by using a tracked tool. Once they think they have positioned the mirror correctly, they perform the drilling task.

3. *Focus and context rendering with dynamic virtual mirror:* In contrast to mode 2, mode 3 (see figure 5.15(c)) uses a dynamic virtual mirror, which automatically aligns with respect to the drill as described in section 5.11(f). This removes the need for manually repositioning the mirror during drilling compared to mode 2. A side effect of the automatic alignment in combination with automatic focus point selection in the mirror view is that the focus point is always on the drill tip.

Both virtual mirror modes, mode 2 and 3, have the *automatic focus point selection* in the mirror perspective activated (see section 5.11(f)). The drilling accuracy of the surgeons was evaluated by comparing the final drill tip position values for the three test runs per mode. Additionally a NASA-TLX questionnaire was completed by the surgeons after the experiment. NASA-TLX (Task Load Index) is a standardized questionnaire to measure the subjective workload perceived by operators using human-machine interfaces. Typical measurements include: (a) mental demand, (b) temporal demand, (c) frustration, (d) performance, (e) physical demand, and (f) effort. Additionally, we asked the surgeons to rate the combination of contextual in-situ visualization and occlusion handling techniques.

5.5.2.2 Results

Drilling depth evaluation We used the distance of the drill tip end points to the plane defined by drill direction (defined by drilling sleeve) and the point with ideal drill depth along the drill direction line to compare the three visualization modes. The ideal drill depth was defined by an expert surgeon before the experiment.

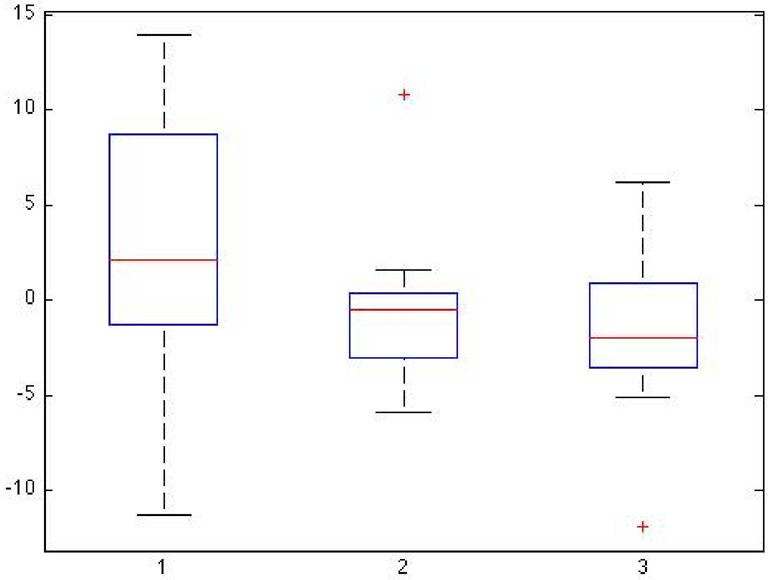


Figure 5.16: Box plot of the drilling depth evaluation for mode 1 to 3.

For each mode the drill tip end points of all test runs of all surgeons are combined and the mean, median point to plane distance and their standard deviation is computed. Figure 5.16 depicts the results of mode 1 to 3 (from left to right) in a box plot diagram. The results of the two mirror modes are both better than for the non mirror mode. For both modes the mean point to plane distance is negative, meaning the second bone cortex was never perforated. In contrast the non-mirror mode mean distance is positive. Also the mirror mode results are much more compact compared to the non-mirror mode, indicating a higher precision and accuracy.

Questionnaire Table 5.6 presents the results for the six categories of the NASA-TLX questionnaire completed by the surgeons. Frustration level (see table 5.5(c)) and performance (see table 5.5(d)) show the highest difference for the virtual mirror modes and the non-mirror mode. Temporal (see table 5.5(b)) and mental (see table 5.5(a)) demand both have increased for the mirror modes, however only moderately. Physical demand (see table 5.5(e)) and effort (see table 5.5(f)) to complete the task show no noticeable distinction between non-mirror and mirror modes.

(a) Mental demand, low (1) to high (20)

Mode	mean	std. deviation
1	6.67	4.50
2	9.50	5.28
2	8.33	6.12

(b) Temp. demand, low (1) to high (20)

Mode	mean	std. deviation
1	3.83	2.86
2	4.67	2.94
3	4.33	2.94

(c) Frustration, low (1) to high (20)

Mode	mean	std. deviation
1	8.83	6.31
2	4.17	1.94
3	4.83	1.47

(d) Performance, good (1) to bad (20)

Mode	mean	std. deviation
1	10.17	6.37
2	4.50	1.38
3	6.5	5.54

(e) Phys. Demand, low (1) to high (20)

Mode	mean	std. deviation
1	6.33	3.32
2	5.167	2.32
3	6.50	2.74

(f) Effort, low (1) to high (20)

Mode	mean	std. deviation
1	8.5	5.21
2	7.83	4.40
3	8.0	4.82

Table 5.6: Results for NASA-TLX workload assessment questionnaire completed by surgeons for the three different visualization modes (1) In-situ contextual visualization, (2) second in-situ perspective using fixed mirror (3) second in-situ perspective using dynamic, instrument aligned mirror. All results rounded to second decimal place.

The reported higher mental and temporal demand of the virtual mirror modes can be explained by the effort of positioning the mirror for mode 2, and the additional perspective to observe and interpret in the surgeon's mind. Especially mode 2 poses the highest temporal demand of all three modes, due to a re-positioning of the virtual mirror during the procedure, if the first selected mirror position and orientation are found to be insufficient. Increase of mental demand for mode 2 is largely due the effort to correctly position the mirror in one attempt. However, the increase in mental and temporal demand is moderate considering that all surgeons used the virtual mirror for the first time in this experiment. We expect mental and temporal demand to be reduced by further training and adaption of the surgeons to the virtual mirror.

Frustration levels and performance have been reported with the highest differences for the non mirror and mirror modes. Completing the drilling tasks using either mirror mode is perceived with a much smaller frustration level, two times better, by the surgeons than the non-mirror mode. This, is most probably due to the possibility to see the second bone cortex in the mirror image, occluded in the non-mirror mode. Additionally, the red drill safety margin aids in stopping the drill at the correct depth. Subjectively perceived performance when completing the drilling task is analog to frustration perceived as twice as good for the mirror modes compared to the non-mirror mode. The results correlate with the findings from the tracking data based drill depth evaluation.

Interestingly, the fixed mirror mode is the least frustrating mode with the best performance ratings. The slightly higher frustration level and less performance of the dynamic mirror mode can be explained by the automatic alignment algorithm and the first exposure of the surgeons to it. Slight rotations of the drill can result in temporally non-optimal positioning of mirror with respect to the drilling VOI. The additional efforts to correct the mirror alignment can explain the slight difference in frustration and performance. With increasing proficiency by training we expect a reduction of the difference for the two mirror modes over time.

In addition to complete the NASA-TLX we asked the surgeons to rate the occlusion handling for instruments and surgeon hands on a scale from very good (1) to very bad (1) (see table 5.7). Questions asked were: (**Q1**) *How did you find the occlusion handling for the glove?* (**Q2**) *How did you find the occlusion handling for the surgical instruments?* (**Q3**) *How did you find the combination of both?*

Question	mean	std. deviation
Q1	1.67	0.82
Q2	1.60	0.89
Q3	1.60	0.89

Table 5.7: Answers to questions **Q1 - Q3** about occlusion handling in AR visualization (Rating: Very good (1), very bad (5))

The results show that the majority of the surgeon judged the currently available occlusion handling as at least good.

5.6 Summary

Introduction of techniques for hardware accelerated volume rendering for medical AR is an important step forward towards the real-time integration of the complete 3D data into the medical AR environment.

We have presented our results of the integration of a high quality hardware accelerated volume renderer into an AR environment. The rendering pipeline and advanced rendering techniques were adapted and optimized for stereoscopic rendering on a HMD. The occlusion problem of real and virtual objects was addressed in this work, by simple yet effective methods for HMD-based in-situ AR visualization for diagnosis and minimally invasive procedures. The method allows thereby an improved, more natural perception of the augmented reality scene.

The performance of the presented techniques has been demonstrated and evaluated in several experiments by visualizing real human CT data of the skull and the complete torso on realistic anatomical phantoms using different rendering techniques. In an in-vivo experiment the renderer was successfully used to augment a human foot by its pre-acquired CT dataset and evaluated by two experienced trauma surgeons.

We have evaluated surgical drilling accuracy and precision using the presented visualization techniques in an experiment with six experienced surgeons. The results show that advanced in-situ visualization aids in improving navigation accuracy and precision. Furthermore, the experiment participants reported subjectively perceived reduced levels of frustration during task completion and better performance using the combination of focus and context rendering and virtual mirror. The perceived better performance concurs with the measured improved drill positioning accuracy and precision using the virtual mirror. Furthermore, all participants judged the implemented occlusion handling as good and beneficial.

5.6.1 Discussion

The ultimate goal is to provide the best visualization of the three-dimensional image data to the physicians at any point during a medical procedure. However, functionality comes before fancy visualization. Thus, determining the best rendering technique and parameters for each medical application is another important issue and interesting subject of research. Future work should therefore focus on evaluation of the already existing rendering modes for a defined set of medical procedures together with our clinical partners.

The specific requirements have to be analyzed for each procedure over the complete workflow, from pre-operative imaging till the end of the intervention. For this purpose we plan to record video and tracking data for exemplary procedures in order to use them as off-line training datasets. For these datasets the visualization modes can be interactively changed and adjusted based on feedback from our medical partners. Thereby, the visualization can be iteratively refined and optimized for the specific medical procedure.

Concerning the currently implemented occlusion handling, we do not claim that the method is suited for augmentation of all medical procedures, especially not for traditional open surgery. Simple color segmentation is not enough for this task. However, for HMD-based AR pre-operative inspection of the patient or for in-situ visualization in minimally invasive procedures, with access restricted by ports into the patient, the approach is fine. The integration of more elaborate methods for real-time occlusion handling, e.g. depth map reconstruction using the stereo camera system, could be seen as future work. However, even by this simple method the subjective perception of the scene was definitely improved demonstrated by the generated AR images and questionnaire results returned from clinical users of the system.

CHAPTER 6

GPU-accelerated US Simulation from CT for Application in Medical Training and Multi-Modal Registration

Ultrasound is one of the most widely used and popular imaging modalities in modern clinical practice. Because of it's many benefits (see section 2.1.4) it is used in a variety of clinical applications. The main drawback of US imaging, however, is the quality of the acquired images and a low signal to noise ratio (SNR), which makes navigation and interpretation of the acquired images, particularly challenging. Both, quality of the images and their interpretation, are highly dependent on the skills and experience of the user manipulating the US probe. Extensive, time-consuming hands-on training of examinations on real patients is necessary for the effective use of US in clinical practice. Therefore, US imaging is often said to be one of the hardest to master imaging modalities in medicine.

Recently US simulation systems have been shown to improve the performance and skills of users, significantly (e.g. see [132]). This is due to the fact that the trainees can practice localization and acquisition of US without the time-constraints imposed by such practice on the patients and can also access a variety of cases which have been collected and stored in the simulation system's database over time. Besides its application in training simulators, US simulation could play a central role for multi-modal image registration [117, 222] of US and CT image data. In Wein *et al.* [222], a simplified US model was used for US simulation from CT for rigid registration of a set of tracked 2D US images with a 3D CT scan. Diagnosis is improved by fusion of information provided from CT and US. Additionally fusion of CT and US can aid in US image guided interventions by simultaneous display of registered pre-interventional CT planning data and interventional US image data.

The work presented in this chapter addresses both application areas of US simulation from CT. It originally started as a research project to use GPUs to accelerate US simulation for multi-modal fusion together with Dr. Wolfgang Wein and Athanasios Karamalis at the Chair for Computer aided Medical Procedures, Munich in winter 2007. In 2008 we were happy to collaborate with Ramtin Shams from National University of Australia and bringing together the methods he presented in [183] with our GPU-accelerated US simulation framework. Results of this work were published in [110, 111] and later integrated into further research projects on multi-modal registration [112, 225] and an AR based US training simulator [19].

6.1 Related Work

A number of systems including commercial products are available for US training (e.g. [1, 4, 38, 69, 70, 193, 202, 204, 221]). These systems allow navigation with a virtual probe within the space of pre-recorded US images. The acquisition protocol is typically 3D freehand US with a compounding stage where 2D US images are combined to create a 3D volume or straight 3D US. At run-time, during training sessions, the position and orientation of the virtual probe is tracked and the relevant US planes are re-sliced from the previously computed volumes. Technically, these systems simulate the US acquisition rather than the US itself.

Fully synthetic simulation of US has been proposed by Jensen *et al.* [82, 83, 84, 85, 86] based on an acoustic wave-propagation model and using the concept of spatial impulse response [197, 209] which is implemented in a program called Field II [82]. The program can be used to simulate any linear US system with single or multi-element transducers, any given apodization, focusing, pulse excitation scheme and aperture geometry [85]. The program requires location and strength of scatterers as input and gives best results with carefully designed and synthetically generated scattering patterns. As such, the program is mostly used to determine the effects of various parameters on transducer design. Additionally, the simulations for even a single B-mode image take an extremely long time and need to be parallelized (the execution time for a B-mode image with 128 RF scan lines and 1.000.000 point scatterers is in the order of 2 days on a single CPU), which makes it impractical for real-time simulation and in training applications.

More recently, simulation of US from CT volumes has attracted interest. Several publications [75, 192, 213, 238] demonstrate the importance of patient specific US simulation, mainly from CT scans, used in training simulators in medical education. Such training systems allow many medical apprentices to practice the acquisition of US images of various pathologies without relying on available US systems and real patients. In Hostettler *et al.* [75], US images were simulated from CT using a ray-casting approach. However, US modeling is not described in detail. The resulting images seem to cover only very basic US phenomena and were used for the training of young physicians.

Additionally these simulators can be extended to simulation of US image guided interventions. A number of publications deals with the simulation of US-guided needle insertion, one of the most common minimally invasive interventions in clinical practice. The standard training methods for these procedures, require the supervision of an experienced physician, which is both highly costly and time consuming. In [192], Soler and Marcescaux use a ray-casting based US simulation from CT for the simulation of needle insertion procedures and training of young physicians. Zhu *et al.* [238] present a training system that simulates US images by raster scanning, whereas user interaction was realized by a tracked US probe and a tracked biopsy needle on a latex phantom. Vidal *et al.* [213] propose a training system combining haptic hardware with US simulation from CT and the rendering of mesh models generated from the CT volume in an off-line pre-processing step. GPU acceleration is used to a very limited extent, namely for the generation of 2D MPR images from CT needed for the US simulation algorithm.

One common problem with traditional US simulation systems (e.g. [1, 4, 38, 69, 70, 193, 202, 204, 221]) is that the simulation is realistic as long as the operator remains within close vicinity of originally acquired positions and orientations. As the probe is navigated further away from the acquisition positions, the images become less realistic, since view-dependent US effects are no longer accurately represented. The acquisition protocol is also complicated and requires the volume of interest to be imaged from various positions and not to contain view-dependent artifacts such as shadowing, and the effect of a fixed gain and focus. Then there is, of course, the issue of compounding the images and accumulated errors due to mis-registration and accumulation of intensity values with varying intensities due to view-dependent artifacts.

Use of CT images as the basis for simulations not only avoids the aforementioned drawbacks but also has the advantage of allowing for patient specific simulations, ease of navigation for novice users as they can practice US navigation with the help of corresponding CT information (this extra assistance is obviously turned off at later stages of training). It also provides easier access to raw data for simulation, as CT images are routinely acquired for diagnostic and planning and the acquisition protocol is uncomplicated and streamlined.

A decisive factor for the practical usability of a US simulation for any clinical application is that it meets the performance constraints defined by the application. US simulation is computationally expensive, even when ignoring most US specific imaging phenomena.

Our team has investigated US simulation using a simple ray-based modeling of US for registration purposes in [224], which uses a simple US simulation. In [183] Shams *et al.* proposed an enhanced modeling of US, which results in a more realistic US simulations from CT images suitable for US training (see figure 6.1(c)). In this work we present a a GPU-accelerated framework for US simulation from CT, supporting ray-based simulation models of varying complexity, unifying and extending the methods proposed by Wein *et al.* [224] and Shams *et al.* [183].

6.2 Ray-based US Simulation

When an US beam travels through a piecewise homogeneous medium, it gets partially reflected at the interface between two media with differing *acoustic impedances*. The change in acoustic impedance is the main physical interaction that makes US visualization possible [68]. The amount of energy which is reflected is determined by the *reflection* *coefficient*, α_R, given by

$$\alpha_R = \left(\frac{Z_2 - Z_1}{Z_2 + Z_1}\right)^2, \tag{6.1}$$

where Z_1 and Z_2 are the acoustic impedances of the media in question. The acoustic impedance itself depends on the speed of sound in a medium and the density and is given by $Z = \rho c$. The remaining energy that passes through the interface is characterized by the transmission coefficient $\alpha_T = 1 - \alpha_R$.

The US simulation from CT images is based on the premise that there is an approximately linear relationship between CT HU values and the acoustic impedance for soft-tissue [224]. We perform automatic segmentation of bone and air interfaces in order to calculate the reflection coefficient for air-tissue and bone-tissue interfaces where CT values cannot be directly used in (6.1). The reflection of US at tissue interfaces is non-specular and subject to scattering. We use a Lambertian scattering model where the intensity of the scattered signal depends on the incidence angle and can be written as:

$$R(\mathbf{x}) = \alpha_R(\mathbf{x})\, I_i(\mathbf{x})\, |\mathbf{r}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})|, \tag{6.2}$$

where $I_i(\cdot)$ is the intensity of the incident beam at the interface, $\mathbf{r}$ is the unit vector in the direction of the beam, $\mathbf{n}$ is the surface normal, $|\cdot|$ is the absolute value operator, and $R(\cdot)$ is the intensity of the reflected signal. According to a Lambertian scattering model, the intensity of the signal, as perceived by an arbitrary viewer, is independent of the viewing angle and only depends on the angle of incidence. If we show the initial intensity of the US by I_0, and the incident intensity at spatial location $\mathbf{x}$ by $I_i(\mathbf{x})$, the accumulative attenuation at point $\mathbf{x}$ will be given by $I_i(\mathbf{x})/I_0$. The reflected signal travels back through the same attenuating medium (ignoring any refraction), and as such the intensity of the signal as sensed by the receiver, $I_r(\mathbf{x})$, is attenuated by the same coefficient as in the forward path and can be written as

$$I_r(\mathbf{x}) \propto \alpha_R(\mathbf{x})\, \frac{I_i^2(\mathbf{x})}{I_0}\, |\mathbf{r}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})|. \tag{6.3}$$

The simulation of refraction, scattering and attenuation is much more complex and requires more precise knowledge of the tissue during ray traversal. To correctly simulate refraction of the ray at interfaces one has to know the tissue interface specific refraction coefficient. Using this coefficient and the incident angle one can compute the refracted beam direction using Snell's law. However, as the simulation of it requires the creation of an additional, data set specific, coefficient image or volume, and refraction contributes only to a small part to the overall visual appearance of an US image it is often neglected in ray-based US simulation models.

Scattering, reflections from very small objects (the size of the wavelength or smaller), is one of the most prominent US imaging effects. Scattering is dependent on many factors: (a) tissue, (b) number of scatterers per unit volume, (c) the acoustic impedance differences at the scatterer interfaces, (d) the size of the scatterers, and (e) used frequency. As the

(a) (b) (c)

Figure 6.1: (a) Schematic of a convex array transducer with a multi-element active aperture, (b) Ultrasound field of view superimposed on the liver of a human subject. (c) Our simulated US (combined reflection and scattering images.), for the region of interest shown in (b). Also notice shadowing on the right-hand side due to an air-tissue interface and in the middle-bottom of the image due to a a bone-tissue interface.

creation of a dataset containing the scatterer information is very difficult and very hard to assess, most methods approximate scattering by various means. Wein's [222] method remaps the CT intensity with an empirically estimated transfer function, Shams [183] creates a scattering map in a computationally highly expensive pre-processing step, Zhu *et al.* [238] use texture banks of real US images for creating realistic appearing simulated US images, other models employ random generated noise data sets, e.g. Perlin or Rayleigh noise.

Attenuation describes the reduction of the sound wave energy during tissue traversal with increasing distance from the source. There exist two sources of attenuation in tissue (a) reflection and scattering at interfaces, and (b) tissue-specific absorption. Attenuation in tissue can be described by a, frequency dependent, attenuation coefficient. Although the attenuation coefficient for most tissues is almost proportional to the frequency the simulation of attenuation is complex. For patient specific, physically correct simulation of US attenuation an attenuation coefficient map of the data set is necessary. In most cases this map has to be created by a time consuming segmentation from a CT or MRI data set.

The effect of a finite beam width produced by multiple transducer elements is modeled by integrating the perceived intensities along the active wavefront at a specified depth using a suitable window function [183]. For a linear array transducer we can write

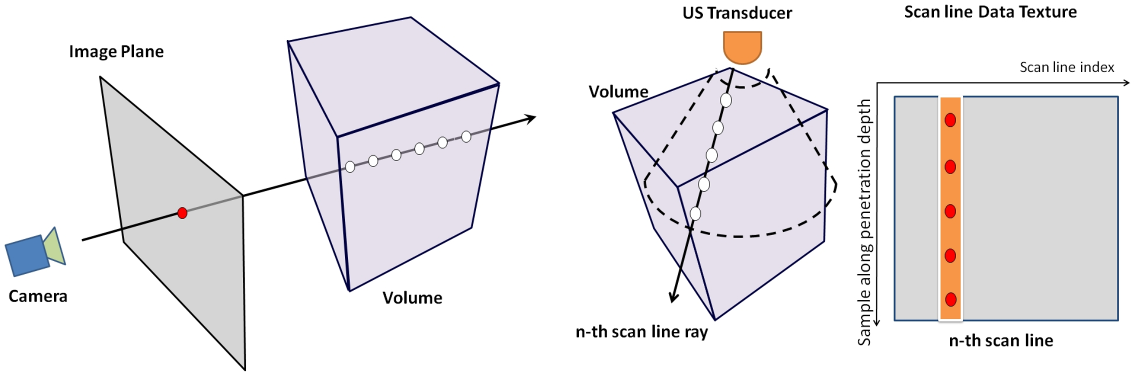
$$I_r(x,y) \propto \int_{x-\ell}^{x+\ell} \alpha_R(u,y) \frac{I_i^2(u,y)}{I_0} \left|\mathbf{r}(u,y).\mathbf{n}(u,y)\right| \omega(u) du, \tag{6.4}$$

where $\omega(\cdot)$ is the window function, and ℓ is length of the active aperture, given by $\ell = n_a(w_e{+}s_e)$, where n_a is the number of active elements, w_e is the width of each element, and s_e is the spacing between adjacent elements. A square, triangular, or a Hann window can be used for apodization depending on how the transducer elements are being activated.

6.3 GPU Implementation

We formulate the US simulation problem on the GPU as a ray casting problem. The (virtual) US transducer is positioned within the space of the CT volume. For every transducer element, and depending on the geometry of the probe (i.e. linear or curvilinear), an US beam is cast and multiple rays are processed in parallel by the GPU. For each sample along a ray, equation (6.3) is computed inside a *fragment shader*. The results are stored as a measure of the acoustic intensity received by a transducer element from a point at a given depth in the anatomy along an US beam and displayed as an image.

The algorithm is implemented in C++, OpenGL, and GLSL. The OpenGL FramebufferObject (FBO) Extension[1] is employed to efficiently render to off-screen render targets (i.e. 2D and 3D textures in GPU memory).



(a) Direct volume rendering (b) Ultrasound simulation

Figure 6.2: Difference of ray-casting algorithms for direct volume rendering and US simulation (a) for DVR, multiple samples along a ray require a single storage (red dot in the image plane), (b) for US simulation, every sample along a ray requires a corresponding storage.

A key difference of our ray-casting algorithm for US simulation compared to traditional ray-casting algorithms (e.g. for direct volume rendering (DVR) [36]) is the need to store sample values along each ray. In a standard ray-casting algorithm, based on the light propagation model, the output for each ray is a single value which is the result of combining the color and intensity contribution of each sampled element along a ray (see figure 6.2(a)). As shown in figure 6.2(b), for US, an acoustic echo is returned from each sample along a ray and needs to be stored separately. For a high quality simulation, we need 256 or more samples along each ray. This largely exceeds the number of output channels per fragment. As such, we use a multi-pass algorithm for efficient implementation of US ray-casting on the GPU.

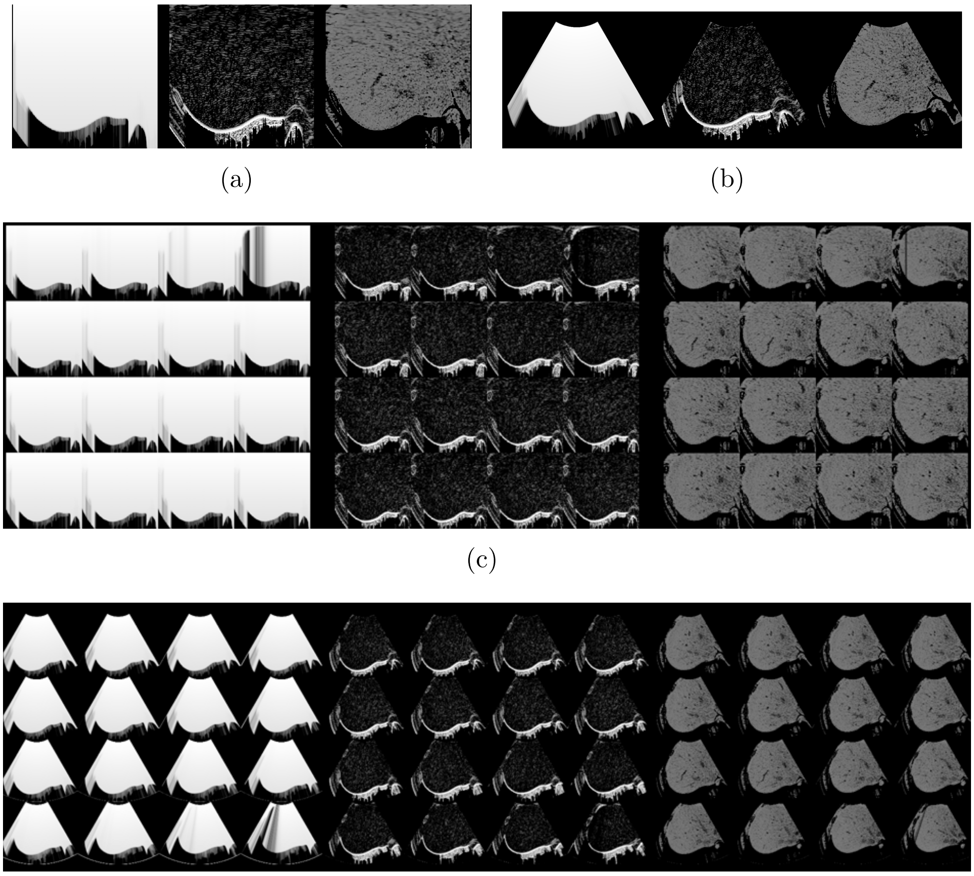
Various data structures are allocated and loaded during the initialization stage and an optimal memory layout is determined. CT data, US ray start and end vectors are

---

[1] http://www.opengl.org/registry/specs/EXT/framebuffer_object.txt

stored in textures. Ray start and end vectors are used to compute position of samples within the CT volume at each pass of the algorithm. Ray start and end vectors have to be re-initialized, every time that the user changes the orientation or position of the probe.

The ray-casting algorithm is designed to be independent of the probe geometry and US dimensions. Scan line information is stored in 2D textures for both 2D and 3D US images. This is a major benefit and allows us to use the same algorithm for simulation of 2D, 3D, linear, curvilinear and freehand US. The original dimension and shape of the US image are restored in the scan conversion stage.



(a) (b)

(c)

(d)

Figure 6.3: Temporary component scan line data and Cartesian images as stored in GPU texture memory from a simulation using Wein's model. (a,b) Single 2D image simulation. (a) component scan line data, (b) Cartesian images. From left to right: transmission, reflection and echogenecity remapped CT. (c,d) Simultaneous simulation of 16 2D images arranged in 4 x 4 tiles. (c) Component scan line data and (d) Cartesian images.

We need three render targets for storage of intermediate results and acoustic intensities. This is to store $I_i(\mathbf{x})$ and $I_r(\mathbf{x})$ (refer to section 6.2). $I_i(\cdot)$ is calculated recursively

$$I_i(\mathbf{x}) = I_i(\mathbf{x} - \Delta\mathbf{d})(1 - \alpha_R(\mathbf{x} - \Delta\mathbf{d})), \tag{6.5}$$

where $\Delta\mathbf{d}$ is the incremental sampling vector along a given ray. Storage of $I_i(\cdot)$ scan line data requires two textures to avoid read/write conflicts and synchronization issues. The algorithms interleaves data read/writes for even/odd rows of the scan lines (ping-pong rendering). This is to ensure that all fragment shaders finish writing into row k, before starting row $k+1$ which requires values of the previous row.

A practical consideration in allocating textures is the memory layout. GPUs typically have an upper-bound for the width and height of the textures. Regardless of the available memory, one cannot allocate a texture that exceeds the limit in one or multiple dimensions. Performance-wise, GPUs typically perform better with square textures whose dimensions are a power of 2. We need a texture of size $n \times d$ for simulating an US with n transducer elements and d samples along each ray. This is not a problem for 2D US as the number of scan lines hardly exceeds 256. However, for 3D US the number of elements and as a result scan lines can easily exceed the limit. Therefore, the memory layout is optimized to be close to square and several tiles of scan lines are arranged within the texture, as needed. (see figure 6.3)

6.3.1 Creating the Ultrasound Image

Using the model presented in section 6.2, we generate an image called the reflection image from CT data. The reflection image simulates view-dependent ultrasonic effects due to reflection and attenuation of the signal. Tissue boundaries are emphasized in the image and shadows due to large impedance mismatches between tissue-bone and tissue-air interfaces are simulated.

We also generate a scattering image using Field II by preprocessing a CT volume from a fixed view-point as described in [183]. Figure 6.4 shows a volume rendered CT image of the abdomen with the corresponding scattering image.

The reflection and scattering images are combined using the following formula:

$$I_{us}(\mathbf{x}) = (G_{\sigma_1}(\mathbf{x}) * I_r(\mathbf{x}) + \alpha G_{\sigma_2}(\mathbf{x}) * \alpha_T(\mathbf{x}))I_s(\mathbf{x}), \tag{6.6}$$

where $I_{us}(\cdot)$ is the US image, $I_r(\cdot)$ is the reflection image, $I_s(\cdot)$ is the scattering image, α is a blending coefficient, and G is a Gaussian filter with 0 mean and adjustable standard deviation (σ_1 and σ_2) used to smooth the output of the image registration process. The blending parameters, α, σ_1 and σ_2 are adjusted by the operator for best viewing results.

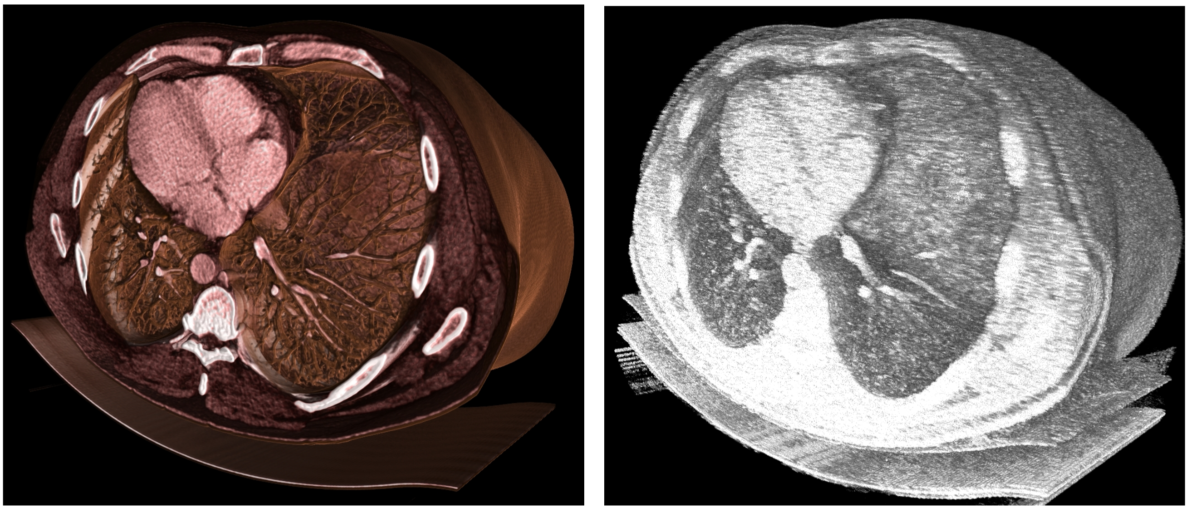
The resulting image has a large dynamic range which far exceeds the dynamic range of the display and range of intensities that can be identified by the human eye. To reduce the dynamic range, we compress the signal using the following log-compression method

$$I_c(\mathbf{x}) = \begin{cases} 0, & I_{us}(\mathbf{x}) < \max\{I_{us}(\mathbf{x})\}10^{-\beta/10} \\ 10\log_{10}^{I_{us}(\mathbf{x})/\max\{I_{us}(\mathbf{x})\}} + \beta, & \text{otherwise} \end{cases} \tag{6.7}$$

where β is the dynamic range of the compressed signal.

6.3.2 GPU Simulation Pipeline

Our simulation pipeline as shown in figure 6.5 consists of five stages: the scan line traversal, pre-scan conversion, scan conversion, post-scan conversion and compositing stages.



(a)

(b)

Figure 6.4: (a) Volume rendering of the CT image, (b) Volume rendering of the corresponding scattering image.

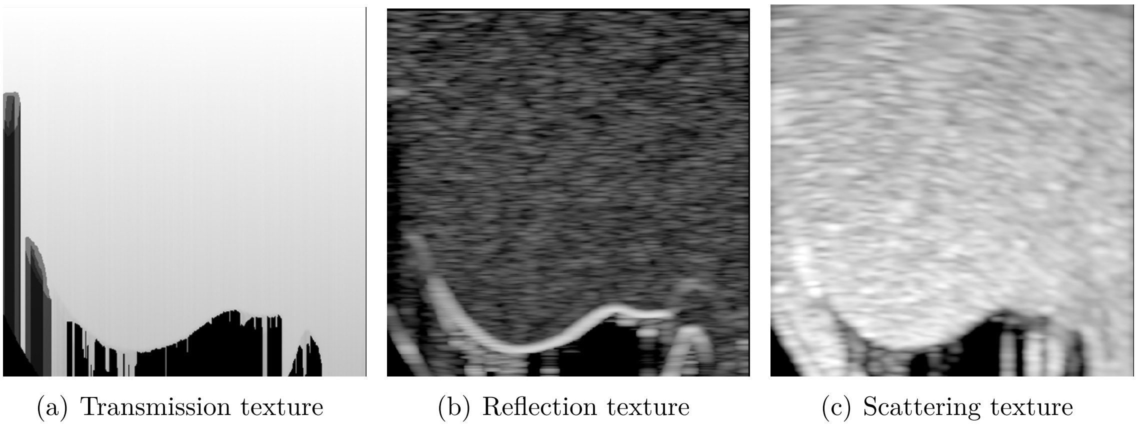
The pre-scan conversion, post-scan conversion and compositing stages are optional and are executed if required by the underlying US model (e.g. the model in [224] utilizes the scan line and scan conversion stages only, while the more complex in [183] invokes all stages of the pipeline).

Figure 6.5: Stages of US simulation pipeline. From left to right: Scan line simulation, scan line post processing, scan conversion, post processing, compositing stages. The scan line simulation and scan conversion stage are always executed, other stages are executed depending on the employed simulation model and application.

- *Scan Line Traversal Stage:* As the first stage of the pipeline, the 3D data-set is sampled along each scan line and the values are stored in a 2D texture per component. Each time probe-related parameters are varied by the user, scan line data has to be recomputed. For simulation of an US image with d samples (pixels) along each beam, the algorithm requires exactly d render passes. For simulation of 2D US, typically a single line primitive is executed at each pass. However, for 3D US or simulation of multiple 2D US images, where scan lines are tiled within texture memory, we run m parallel line primitives, where m is the number of tiles along the vertical texture axis. Running multiple line primitives typically provides a better utilization of the GPU resources. This means that our algorithm reaches its full capacity (in terms throughput) for larger simulations (i.e. multiple 2D and 3D US).

The US simulation may require a re-mapping of the CT values so that they can be directly used in equation (6.1). A transfer function lookup texture is used for efficient re-mapping of CT values.

- *Pre-Scan Conversion Stage:* For efficient computation, equation (6.4) can be reformulated as the convolution of the scan line data with an appropriate 1D window function. This computation is performed in the pre-scan conversion stage. Figure 6.6 shows the content of various textures resulting from pre-scan conversion stage for a sample US simulation of the liver.



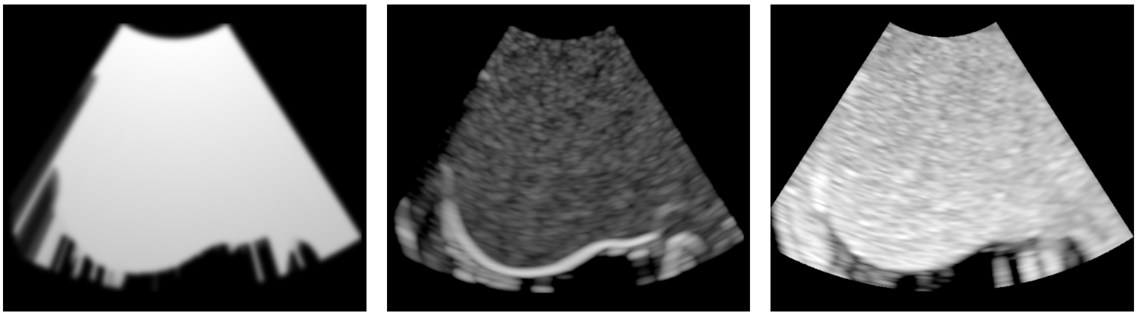
(a) Transmission texture (b) Reflection texture (c) Scattering texture

Figure 6.6: Intermediate results from pre-scan conversion stage (a) Transmission scan line image, no filtering applied. (b) reflection scan line image, hanning window applied. (c) Scattering scan line image, hanning window applied.

- *Scan Conversion Stage:* This stage is used to convert scan line data into a 2D or 3D Cartesian representation. Scan conversion is implemented by backward warping on the GPU using a specialized fragment shader for each probe geometry and dimension. Hardware built-in texture filtering is used for optimal performance. 3D US volume data is scan converted voxel by voxel rendering slices along the Z-axis of a 3D texture in multiple render passes, or in a single render call on most recent graphics hardware supporting GLSL geometry shaders.

- *Post-Scan Conversion Stage:* 2D and 3D simulated images may have to be filtered for improved visual quality according to (6.6). This requires convolution with the appropriate 2D or 3D filter which is implemented by a fragment shader on the GPU. Separable kernels are used in conjunction with two/three render passes for 2D/3D filtering, where possible, to improve the performance. Figure 6.7 shows the content of various textures following the post-scan conversion stage for a sample US simulation of the liver.

- *Compositing Stage:* In the compositing stage, intermediate results from various sources are combined in a fragment shader according to equation 6.6 and equa-



(a) Transmission texture (b) Reflection texture (c) Scattering texture

Figure 6.7: Resulting Cartesian images after scan conversion and post-scan conversion stages. (a) Smoothed transmission image (b) Smoothed reflection image (c) Scattering image.

tion 6.7, which computes the final value for each pixel and prepares the data for visualization.

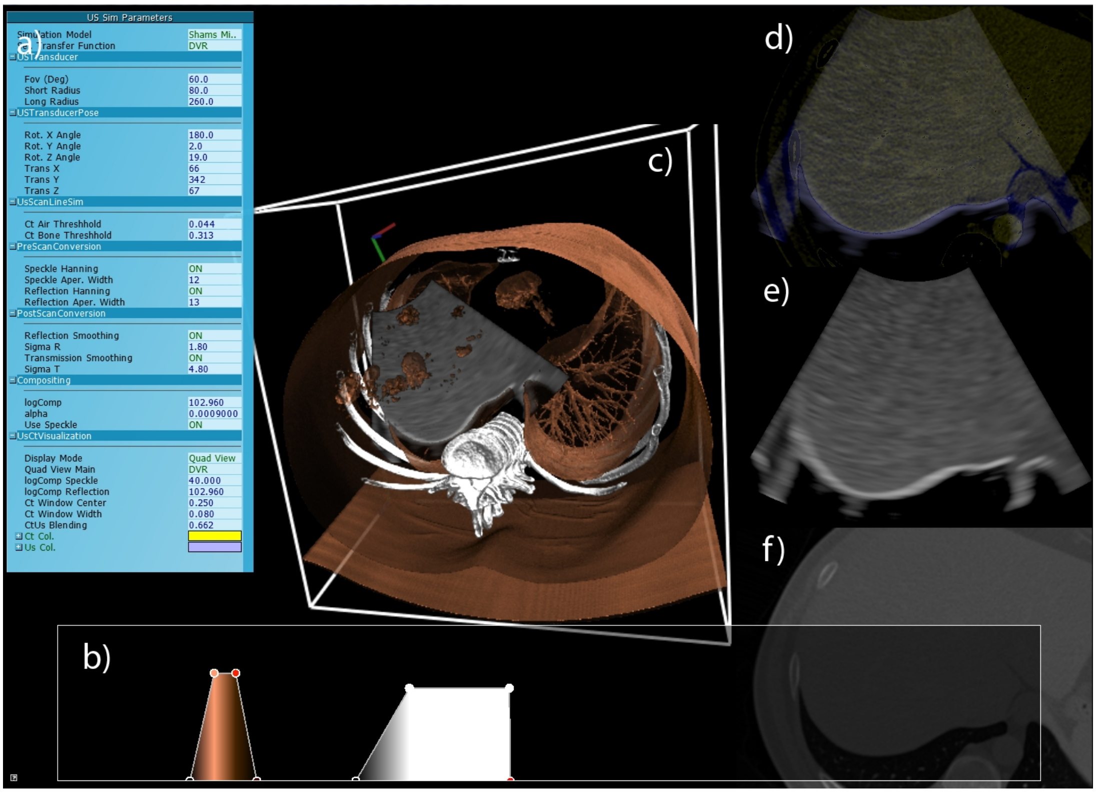
6.3.3 Simulating US Data over time

Our framework supports the simulation of time-varying US image sequences from time-resolved CT data. Similar to real-time volume rendering of 4D data (see 4.4.1) we employ efficient GPU memory transfer techniques to stream single 3D CT volumes of the 4D sequence into the simulation pipeline. The simulation pipeline is executed for each CT volume frame and the resulting simulated 2D or 3D US image is displayed immediately using either 2D or 3D rendering techniques on the screen. For detailed image results we refer to B in the appendix of this thesis, where we present a series of simulated 2D and 3D US images from a 4D CTA data set.

6.4 VR US Simulation Prototype

To test and evaluate the methods described in sections 6.2 and section 6.3 we have developed a prototype application. The parameters of the GPU-accelerated US simulation can be set interactively, and the results are displayed in real-time using different 2D and 3D visualization techniques. Figure 6.8 depicts a screen-shot of the application's user interfaces. The user interface consists of four main views and two widgets for adjusting the simulation parameters and the direct volume rendering transfer function:

- *3D View:* The 3D view displays the 3D CT volume and the 2D US image plane, or 3D US scan volume within the 3D CT volume space. The CT image is rendered using standard DVR techniques. The 2D US image is texture-mapped onto the corresponding plane within the 3D volume. When simulating 3D US data, a wire-frame representation of the 3D scan volume is rendered. The user can change the appearance of the volume rendering (e.g. display internal organs, vasculature, bones or skin surface) in real-time by changing transfer function parameters.



US Sim Parameters

Simulation Model	Shams Mi..
Transfer Function	DVR
USTransducer	
Fov (Deg)	60.0
Short Radius	80.0
Long Radius	260.0
USTransducerPose	
Rot. X Angle	180.0
Rot. Y Angle	2.0
Rot. Z Angle	19.0
Trans X	66
Trans Y	342
Trans Z	67
UsScanLineSim	
Ct Air Threshhold	0.044
Ct Bone Threshhold	0.313
PreScanConversion	
Speckle Hanning	ON
Speckle Aper. Width	12
Reflection Hanning	ON
Reflection Aper. Width	13
PostScanConversion	
Reflection Smoothing	ON
Sigma R	1.80
Transmission Smoothing	ON
Sigma T	4.80
Compositing	
logComp	102.960
alpha	0.0009000
Use Speckle	ON
UsCtVisualization	
Display Mode	Quad View
Quad View Main	DVR
logComp Speckle	40.000
logComp Reflection	102.960
Ct Window Center	0.250
Ct Window Width	0.080
CtUs Blending	0.662
Ct Col.	
Us Col.	

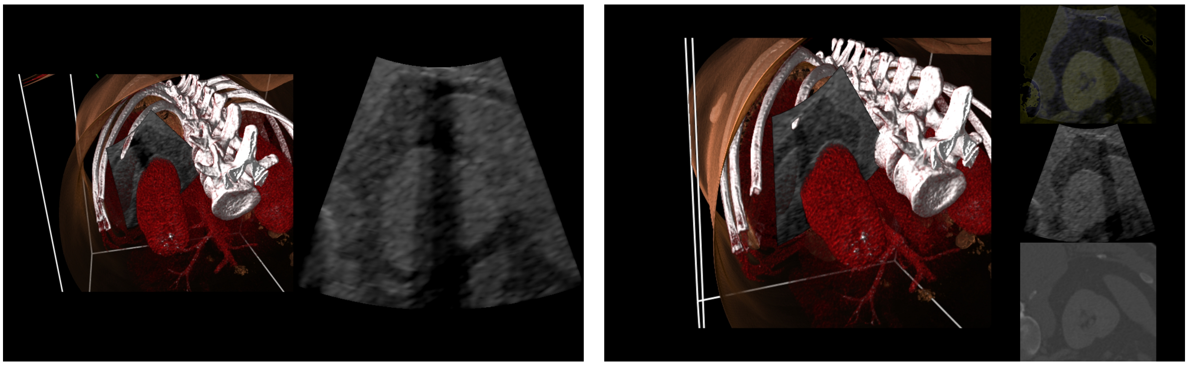
Figure 6.8: Screen-shot of the application: a) application options, b) transfer function widget, c) 3D view, depicting shaded DVR of the CT data and the 2D US image, d) blending of the simulated 2D US image and corresponding CT MPR, e) simulated 2D US image, and f) CT MPR corresponding with the US image plane.

- *Ultrasound View:* displays the simulated 2D/3D US image. For 3D US, the user can choose between the 2D display of coronal, sagittal or axial MPR reconstructions or a 3D DVR of the simulated US volume (see figure B.1, figure B.2, and B.3 for more examples).
- *CT View:* The CT view displays a multi-planar reconstruction (MPR) of a CT plane that corresponds with the current position, orientation and field of view of the US image.
- *Combined View:* shows the fusion of the US and CT images and allows the user to easily compare US and CT features.

Ultrasound simulation and visualization parameters can be adjusted interactively (see figure 6.8 (a), the depicted simulation parameters are for the model by Shams *et al.* [183]). The numerous simulation parameters are organized in groups, e.g. post-processing, transducer orientation. Certain groups are shared among all simulation models, others are specific to a particular US simulation model.

- *Transducer Geometry and Pose:* allows for the selection of the probe geometry (i.e. linear or curvilinear), setting the probe position and orientation, field-of-view, and minimum and maximum penetration depth.

- *Scan Line Traversal:* parameters in this group affect the scan line traversal stage. For instance, for the model in [183] these are the air and bone segmentation thresholds.
- *Pre/Post-Scan Conversion:* the options include type of filters, window sizes, standard deviation of the filters, etc.
- *Compositing:* the options include log compression, component image blending factors and boolean flags denoting whether certain operations should be executed in the compositing shaders.
- *Visualization:* the options include CT window level and window width for 2D CT slice visualization, and blending factors and colors for the combined CT/US visualization.



(a) (b)

Figure 6.9: Real-time direct visualization of a CT volume and simulated 2D US. (a,b) Simulation with a narrow angle curvilinear transducer scanning the left kidney. Note the occlusion artifacts due rays intersecting the ribs.

Real-time Visualization A key component of the prototype application is the concurrent real-time visualization of the simulated US images within the CT data-set (see figure 6.9). Many of the techniques,e.g. pre-integration, empty-space skipping, described in chapter 4 are used to realize a real-time, high-quality, lag-free visualization of the CT and simulated US data. A benefit of the combination of simulation and visualization entirely on the GPU is the absence of having to transfer the simulation results to the GPU, leaving performance to implement advanced visualization techniques.

6.5 Computational Performance Evaluation

The ray-based simulation of US is very efficient on the GPU. In this section, we present detailed performance results for the two main application areas of the simulation framework, simulation and visualization in US training and simulation for registration of US and CT images. The requirements for the two applications are different. Ultrasound training requires more realistic simulation of US images and uses a more accurate simulation model, whereas for registration only a few US specific effects have to be simulated and a more simplified US model can be employed. We first describe the test environment and the data-sets and parameters used for the performance evaluation. Then, we describe the performance of the simulation and simultaneous visualization for US training using the US model presented in this paper. We conclude with an analysis of the throughput performance of the US simulation for registration of US and CT images using the US model by Wein *et al.* [224].

6.5.1 Test Environment

The performance of the US simulation and visualization was evaluated on four different hardware configurations:

1. AMD Opteron 165 CPU (2x 1.8 GHz), 2 GB RAM, AMD/ATI Radeon x1950Pro, 256 MB RAM, 36 Fragment Shader Cores[2]
2. Intel Core Duo 2 CPU (2x 2.66 GHz, mobile), 4 GB RAM, NVIDIA Quadro FX3600M, 512 MB RAM, 64 Shader Cores
3. Intel Core Duo 2 CPU (2x 2.66G GHz), 4 GB RAM, NVIDIA Quadro FX5600, 1.5 GB RAM, 128 Shader Cores,
4. Intel Core Duo 2 CPU (2x 2.66G GHz), 4 GB RAM, NVIDIA Geforce GTX 280, 1.0 GB RAM, 240 Shader Cores

For our performance measurements, we used a CT volume of the abdomen of a human subject with a resolution of 512x512x484 voxels (16-bit, 242 MB). A speckle volume of the same size was pre-computed from the CT data (32-bit float, 484 MB). The full-size volumes were used with Quadro FX5600 (1.5 GB RAM) and Geforce 280 GTX but the volumes had to be down-sampled for the Quadro FX3600M and Radeon x1950Pro cards to fit within their smaller on-board memory.

---

[2] The Radeon x1950 Pro features separate vertex and fragment shader hardware units and implements Shader Model 3.0. The performance of our algorithms on Shader Model 3.0 GPUs is almost completely determined by the number of available fragment shader cores. All other used GPUs from NVIDIA are based on a unified shader architecture, implementing Shader Model 4.0.

6.5.2 Performance of Simulation and Visualization for Medical Training Applications

Training applications require interactive frame rates for operation of a virtual transducer and provision of a smooth and uninterrupted visual feedback. In this section, we first evaluate the combined performance of the simulation and visualization and then compare the performance of simulations on the different GPUs and with a CPU implementation.

Visualization is typically the more time-consuming part of the algorithm. The performance of visualization is dependent on many parameters, e.g. direct volume rendering technique, local illumination model, and the chosen transfer function. For our experiments, we adjusted the parameters for high quality visualization using pre-integrated classification for DVR, local illumination with Blinn-Phong shading and on the fly-gradient evaluation. A fixed resolution of 640x480 pixels was used for ray-casting, the number of samples per ray was set to 512, early ray termination and empty space leaping optimizations were also activated.

Changed Param.	Radeon x1950	FX3600M	FX5600	GTX 280
None - Simulation Only	45	79	162	180
Volume Pose	5	15	32	60
Transducer Pose	4	16	35	75
Transducer Shape	4	15	34	70
Sim. Param.	40	76	157	173
Transfer Function	9	31	47	64

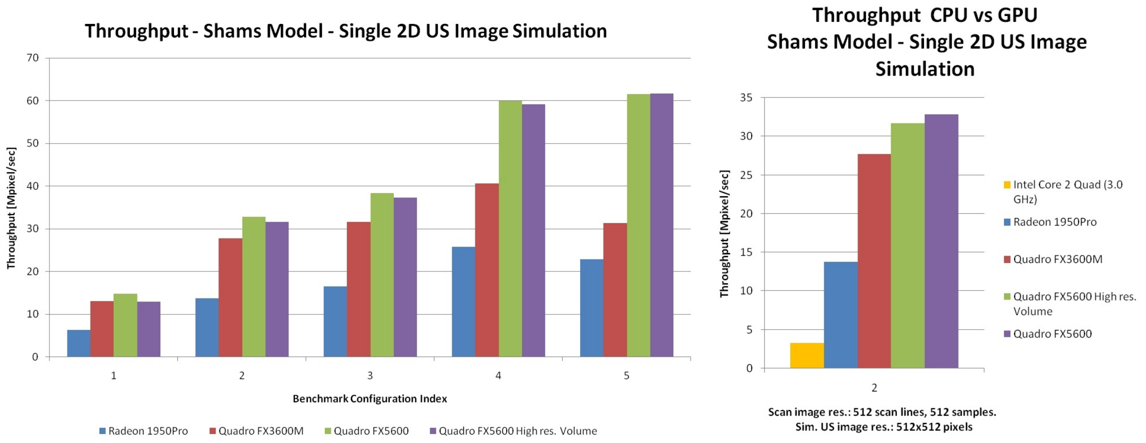
Table 6.1: Performance in frames per second for the combined simulation and visualization.

Table 6.1 shows the average frame rate of the combined simulation and visualization as the operator varies simulation and visualization parameters. Changes in relative orientation or position of the volume with respect to the camera, the US transducer with respect to the volume, or transducer geometry affect the performance. This is due to the fact that these changes require the entire rendering pipeline to be re-executed. As can be seen, the algorithm performs interactively on midrange and higher end GPUs such as Quadro FX5600 and Geforce 280 GTX, under all conditions. The performance is equally good for mainstream GPU models (e.g. 8800GTX/GTS and 9800GTX) that have around the same number of stream processors as FX5600. However, for lower end GPUs the rendering quality has to be reduced in order to achieve interactive frame rates, under all conditions.

The performance of the simulations was also measured by throughput in mega-pixels rendered per second for varying numbers of scan lines and samples, and US image resolutions (see table 6.2 for benchmark configuration details). The results were compared with the throughput of a CPU implementation measured on an Intel Core 2, Quad 3.0 GHz processor. The results are given in figure 6.10. Unlike the GPU version, the throughput for the CPU implementation does not (noticeably) vary with the image size. The GPU implementation outperforms the CPU by up to ~ 20 times (see figure 6.11).

Benchmark Index	Scan lines	Depth Samples	Ultrasound Image Resolution
1	256	256	256x256
2	512	512	512x512
3	512	512	640x480
4	512	512	800x600
5	1024	1024	1024x1024

Table 6.2: Benchmark configuration parameters for performance evaluation of single 2D US image simulation.



(a) (b)

Figure 6.10: (a) Benchmark results for simulation of a single 2D US image using the simulation model by Shams *et al.* for benchmark configurations denoted in table 6.2. (b) CPU versus GPU performance comparison for simulation of a single 2D US image: 512 scan lines with 512 samples, 512x512 pixels image resolution.

6.5.3 Simulation Performance for Registration Applications

Registration of US and CT images requires the repeated simulation of US images for various different orientations and positions from the CT data during optimization of the registration parameters. The simulation of a single 2D US image using the model described in [224] barely utilizes the computational resources of the GPU.

Figure 6.12(b) depicts the throughput for a single 2D US image using the scan line and US image resolutions given in table 6.2. The throughput is limited by the number of active fragment shaders/stream processors and control program overhead in the scan line traversal stage. As can be seen, the throughput improves as the size of the US image increases, since GPU resources are being more optimally utilized for larger images. The authors of [224] kindly provided us with the timings of their CPU US simulation C++ implementation on a 2.2 GHz Intel Core 2 Duo mobile processor. The simulation of a single 2D US image, 128x96 took $\sim$ 3.5ms. To compare the performance, we used this value for estimation of the GPU speedups compared to the CPU for the simulation of single 2D US images (see figure 6.12(b)) and multiple 2D US images on the GPU (see

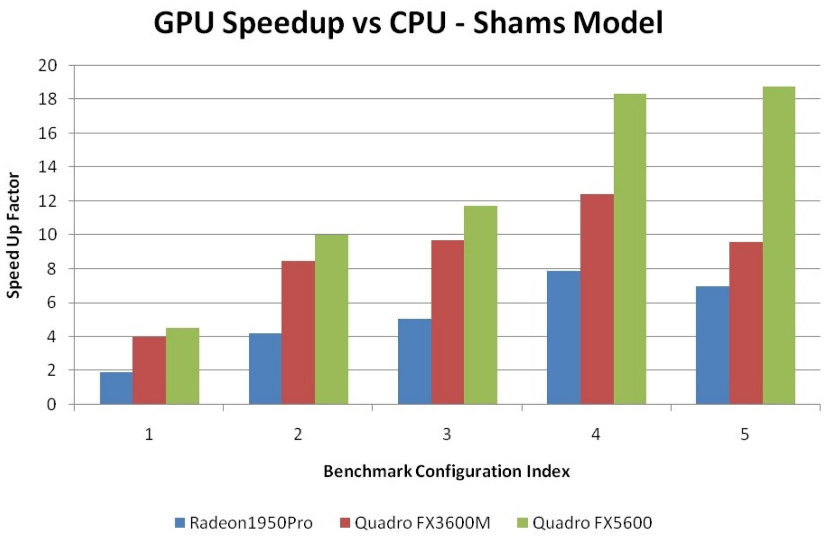


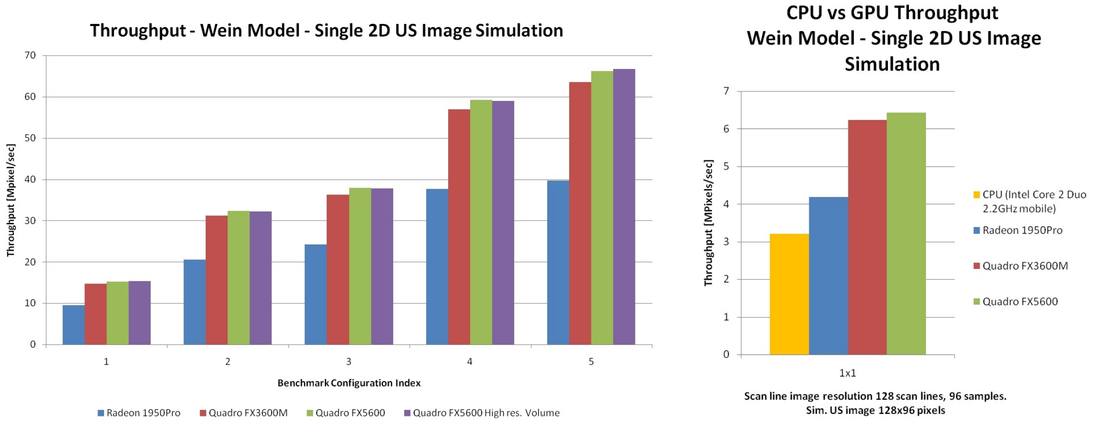
Figure 6.11: GPU speedups for the simulation of a single 2D US image using the model by Shams *et al*. for benchmark configurations in table 6.2.

figure 6.14) of the same resolution.

As can be seen from results depicted in figure 6.13, the key to increase the throughput of the simulation is to process more fragments in a single pass of the scan line simulation stage. We achieve this by packing multiple 2D US images into one large 2D texture on the GPU. Thus, with the number of simultaneously simulated images, increasing speed-up factor can be explained by a better GPU hardware utilization, due to higher number of fragments processed at the same time. More fragments result in more threads executed on the many cores of the GPU and thus hide the latency of GPU texture memory accesses by the large number of additional calculations. For more detailed information specific for NVIDIA's GPUs and tuning for the CUDA architecture we refer the interested reader to [119, 147]. For further detailed performance charts for simulating multiple 2D US images simultaneously we refer to figures B.5 and B.6 in the appendix of this thesis.

Figure 6.13(a), figure B.5 and figure B.6 depict the throughput achieved by the parallel simulation of multiple US images. The throughput increases by the number of image tiles. Using a tile configuration of 32x16 images, each with a 256 scan lines with 256 samples and an US image resolution of 256x256 pixels, we achieved a throughput of >700 MPixels/sec on a NVIDIA Quadro FX5600 board.

Figure 6.14 depicts the speedup for our multi-image GPU simulation compared to the CPU implementation by Wein *et al*. for the simulation of US images of 128x96 pixels resolution and 128 scan lines with 96 samples. With a Quadro FX5600 a speedup of more than 200 times can be achieved.



(a) (b)

Figure 6.12: (a) Simulation performance for a single 2D US image using the benchmark configuration parameters specified in table 6.2. (b) Comparison of GPU and CPU throughput for simulating a single 2D US image of 128x96 pixels.

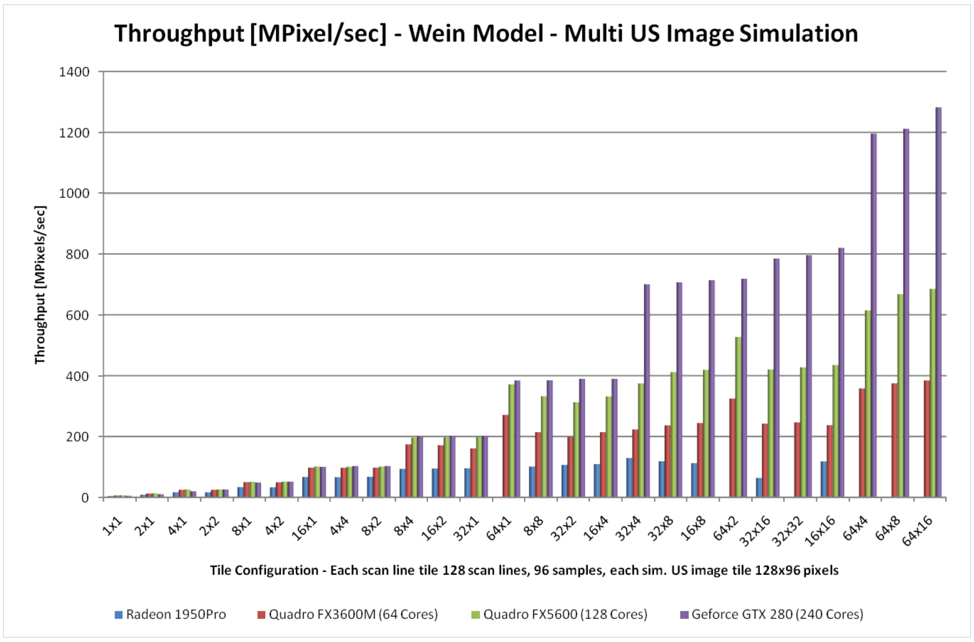
6.6 Summary

In this chapter we have presented a novel framework for GPU-accelerated simulation of 2D, 3D, and 4D US data using ray-based US simulation models, its real-time visualization, which can be executed on a wide range of standardly available GPUs.

The modular design of our US simulation framework supports multiple ray-based US simulation models with various degrees of complexity. Our results demonstrate the superior performance our method for the main areas of US simulation: (a) patient-specific US simulation for medical training and education, and (b) high performance accelerated simulation for multi-modal registration of US and CT data.

The presented techniques have been integrated into a framework for multi-modal registration of US and CT data and successfully applied and evaluated for (a) fully automatic registration of multiple 3D US scans with one CT dataset [112], (b) fully automatic dense deformable registration of pairs of 3D US and CT scans for multiple clinical datasets [225].

Furthermore, the US simulation has been integrated with the stereo view HMD based medical AR system presented in chapter 5 and demonstrated its real-time simulation and visualization performance for patient-based simulation in an initial evaluation by clinicians [19].



(a)

Figure 6.13: (a) Throughput of the simulation pipeline measured on four different GPUs with varying number of processing cores.

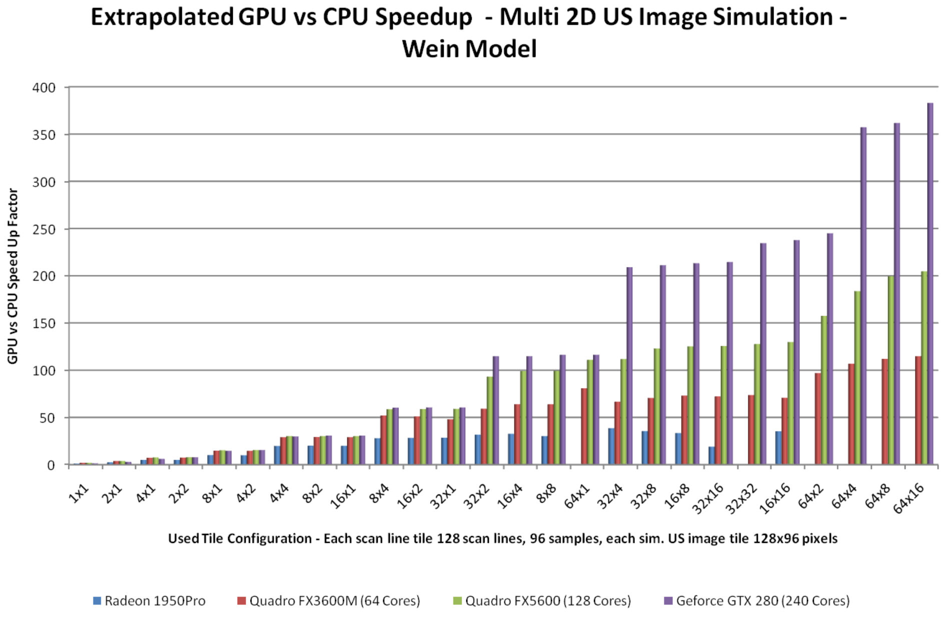


Figure 6.14: Speedups for GPU implementation of Wein's model and simulation of multiple images of resolution 128x96 compared to performance of CPU implementation for simulation of the same total number of images. CPU implementation performance values estimated from timings provided by Wein *et al*.

CHAPTER 7

GPU-based Mono-Modal and Multi-Modal Ultrasound Registration

Registration of two or more medical image datasets is one of the most important key technologies in medical imaging. For many diagnostic and therapeutic medical applications as well as studies over large patient groups automatic data registration is desirable. As pointed out in the introduction of this thesis a large number of applications and problem specific registration algorithms has been developed in the recent years. For a general introduction to the topic we refer to the book [63], survey and review papers in the literature [72, 130, 242]. Probably due to the huge increase of published works on medical image registration algorithms we did not find any more recent survey papers about image registration in general after 2003. Therefore, the interested reader is referred to the chapters about rigid [232] and non-rigid [67] registration from the book [154].

This chapter focuses on efficient, GPU-accelerated methods for mono-modal US-US and multi-modal US-CT registration. The presented work treats two registration problems. As the first problem, in section 7.3 we present GPU-accelerated algorithms for fast simultaneous, rigid registration of multiple 3D US datasets for mosaicking. In this work we cover the following registration problems: (a) Simultaneous image-based registration of multiple 3D US datasets, (b) simultaneous distance-based registration of multiple 3D Power Doppler volumes and (c) simultaneous multi-modal registration of multiple 3D US and one 3D CT datasets. As the second problem, we investigate in section 7.4 fully automatic multi-modal dense deformable registration of a pair of 3D US and 3D CT images.

Before detailed presentation of the developed methods, we present an overview of related work for mono-modal US registration, multi-modal registration of US with CT and MRI, and GPU-accelerated image-based registration.

7.1 Related Work

7.1.1 Mono-modal US Registration

Mono-modal US registration is a challenging task. Due to artifacts, specific to US imaging, e.g. low signal-to-noise ratio, and view dependent effects such as occlusion or attenuation, it is in general complex to define robust similarity metrics for mono-modal US registration. Before discussion of multi-modal approaches for registration of US to CT or MRI data we give an overview how the problem of mono-modal US registration has been addressed in the recent years.

Shekhar *et al.* investigate rigid and non-rigid registration of cardiac 3D US datasets, using the Mutual Information (MI) similarity metric, [187]. In [188] the authors extend their previous work to spatio-temporal registration of pre-stress and post-stress 3D stress echocardiography datasets.

Rousseau *et al.* propose a statistical texture-based similarity metric for rigid registration of US volumes [164]. Gabor filter banks are used to extract feature vectors, denoting radial frequencies and orientations, from the 3D US data. The feature vectors are subsequently used for estimating the statistic similarity metric.

Grau *et al.* describe a phase-based approach for rigid registration of multiple US views from 3D echocardiographic sequences. Use of local phase information is motivated by its invariance to image brightness, contrast and noise. This has the potential to improve registration of lower quality US data compared to pure intensity-based approaches, as the intensity constancy assumption under correct alignment is often invalid. Before registration, the 3D US data is pre-processed by an anisotropic diffusion filter and phase and orientation volumes are estimated. During registration a cost-function describing alignment of phase and orientation is minimized using Powell optimization [56].

In [241], Zikic *et al.* present a variational-based method for estimating a dense displacement field for non-rigid registration of 3D US data. The method is evaluated on five pairs of 3D Freehand US volumes of healthy human liver. Sum of squared differences (SSD) is used as similarity metric during registration and cross-checked with Normalized Cross Correlation (NCC) and MI results.

Woo *et al.* present a method for non-rigid registration of 3D US datasets combining both intensity and phase information in a novel similarity metric using a variational approach [230]. The use of phase information is based on the same rationale as in [56]. The approach is evaluated on phantom and in-vivo 3D echocardiography data of mice and human subjects and reported to outperform standard purely intensity-based similarity metrics.

Registration for US Mosaicking Although recent US systems allow direct real-time acquisition of 3D volume data, acquisition of large volume data is still limited by the US transducer field of view and restricted scanning windows into the patient. Analog to 2D US image mosaicking, 3D US volume mosaicking has been proposed to generate large US volumes from several 3D US volumes. Such 3D mosaic volumes of entire organs help to improve diagnostic confidence, e.g. by reducing scanning orientation induced artifacts, and transcend the role of ultrasound in radiology by providing self-contained 3D data for the reading room. 3D US mosaics have been proposed as databases for generating simulated US images for training and education of physicians [5].

Furthermore, such mosaics provide global anatomical reference, for guidance of minimally-invasive interventions, and allow for automatic mono- and multi- modal registration with pre-operative data. Several recent works have addressed the problem of efficient and robust 3D US mosaic generation.

Gee *et al.* propose a direct pair-wise registration approach [52]. The complexity of the 3D/3D registration problem is reduced to a 2D/2D registration at volume intersection planes. Poon *et al.* proposed two block-based methods for improving the local alignment in the overlap volumes to compensate for the errors introduced by position trackers [157].

Wachinger *et al.* propose a simultaneous registration approach for US mosaicking [218, 219]. The presented framework extends standard pair-wise (bi-variate) similarity metrics to the multi-variate case. In [217] the authors refine the theoretical framework and introduce efficient gradient-based optimization techniques. However, the presented methods were exclusively tested on four 3D US volumes of a baby phantom. Realistic evaluation of the methods with real patient data has not been addressed.

In [142], Ni *et al.* present an indirect pair-wise registration approach, using SIFT feature extraction for US mosaicking. The method is evaluated on 3D US scans of an tissue-mimicking abdominal phantom and 3D US scans of real human liver anatomy.

7.1.2 Multi-modal Registration of US to MRI or CT

Multi-modal registration of intra-operative US images to pre-operative CT and MRI has attracted a lot of interest in the research community. For many medical procedures US imaging is the only viable intra-operative imaging modality. However, because of the low image quality and difficulties to interpret the images, registration to pre-operative planning data for improved image guidance and navigation is desirable. Various multi-modal registration methods, mainly for registration of 2D or 3D US images of (1) brain for neurosurgery, (2) heart for cardio-thoracic surgery, (3) abdominal anatomy, e.g. liver and kidney for guiding minimally invasive interventions, and (4) bone for orthopedic surgery, have been developed in the recent years.

7.1.2.1 US-MRI Registration

Roche *et al.* present a method for rigid registration of 3D US to 3D MRI brain scans [161]. They propose a bi-channel version of the well known Correlation Ratio (CR) similarity metric, using intensity and gradient information from MRI, for registration. In [151] Pennec *et al.* extend the techniques and incorporate non-rigid registration for intra-operatively tracking brain deformations in neurosurgery.

Penney *et al.* [153] propose a training-based approach for registration of sparse Freehand 3D US and MRI scans of liver anatomy. Prior to registration MRI and US images are converted into vessel probability images. The MRI and US to vessel probability mapping functions are estimated from manually segmented training datasets. NCC is used as similarity metric for comparing the vessel probability images during registration.

In [77], Huang *et al.* present a method for spatio-temporal registration of pre-interventional dynamic 3D MRI and real-time 3D echocardiography volumes acquired with a tracked probe. The method is divided into an initial pre-interventional registration and interventional real-time registration part. An initial transformation is obtained by pre-interventional registration of the MRI sequence and a single 3D echocardiography dataset. The temporally closest MRI volume is interpolated from the MRI sequence based on ECG gating and rigidly registered with the 3D US dataset using the MI similarity metric. Interventionally, the pre-registration transformation is used to rapidly spatio-temporally register echocardiography volumes in real-time acquired to the MRI sequence. In [78], the previous work is extended, addressing optimizations to reduce the costs of evaluating the image similarity metric, e.g. reducing number of sampled voxels, extracting most important features. The method is evaluated for multi-modal registration of CT-US and MRI-US data sets. NCC and MI are used as similarity metrics.

Mellor *et al.* propose phase-based, non-rigid MI as similarity metric for US to MRI registration [134]. An initial evaluation of the approach for simulated 2D US images from MRI data and 2D MRI images is presented. In [235], the method is applied for global affine registration of 3D US and 2D MRI images of the heart.

7.1.2.2 US-CT Registration

Ionescu *et al.* present one of the earliest works on automatic image-based registration of CT and US images [81]. They propose a hybrid segmentation/registration approach for fusing tracked 2D US images and 3D CT. The method is tested on prostate and vertebra scans.

Leroy *et al.* present a method for rigidly fusing Freehand 3D US and CT kidney scans [117]. Prior to an intensity-based registration, speckle in the US images is reduced and shadow regions are masked. Tissue interfaces in CT data are amplified to increase correlation with US. Subsequently an automatic registration by maximization of the correlation ratio of the pre-processed images is performed.

In [152], Penney *et al.* extend their earlier work on US-MRI registration [153] to US-CT registration. CT and US images are fully automatically transformed to probability images, based on pre-labeled training datasets. The approach is successfully validated in cadaver experiments for tracked 2D US scans of the femur and pelvis bone.

Wein *et al.* present a series of works dealing with fully automatic rigid registration of 3D US and CT scans. In [226], an intensity-based registration combining a weighted MI similarity metric, correlation of US intensity and extracted CT edge information, and simulation of US occlusion and skin clamping is presented. The approach is evaluated on four patient dataset pairs of Freehand 3D US scans of the neck region and corresponding CT data.

In [222] the previous work is extended and refined. Combination of simulation of US imaging effects, transmission, reflection and echogenecity, from CT data and the novel similarity metric Linear Correlation of Linear Combination (LC^2) is proposed for fully automatic registration of Freehand 3D US and corresponding CT scans. The method is evaluated successfully in a study with 25 patient CT and Freehand 3D US data sets of abdominal anatomy.

In [223], a novel method for spatio-temporal, image-based registration of cardiac CT scans with a single or multiple small 3D US volume acquired by an inter-cardiac catheter is presented. In a pre-processing step the 3D US volume(s) temporally closest to the CT phase are selected based on ECG gating, and the CT volume is transformed to a view-independent US reflection volume. Then, image-based registration using Correlation Coefficient (CC) is performed. As a single inter-cardiac US volume covers only a small field of view, an extension for simultaneously registering multiple US volumes to the CT data is described. The method has been evaluated for US-CT datasets from phantom and in-vivo swine.

Lange *et al.* present a hybrid intensity/landmark registration approach for non-rigid registration of 3D US-CT liver scan pairs [113]. A thin-plate spline based non-rigid registration is performed combining the intensity-based Normalized Gradient Fields (NGF) similarity metric, and an anatomical landmark distance-based cost function. Liver vasculature branchings are chosen as easy to identity anatomical landmarks. The algorithm is successfully validated on three clinical patient datasets.

Gill *et al.* present an automatic, groupwise algorithm for fusing intra-operative 3DUS and pre-operative CT data of the lumbar spine for orthopedic surgery [53]. Prior to registration the CT is divided into sub-volumes for each lumbar vertebra. During registration, the position and orientation of all vertebra subvolumes is simultaneously optimized with respect to the intra-operative acquired US volume. A biomechanical spring model is employed to constrain the motion of the individual vertebrae. The registration is based on the LC^2 measure, proposed by Wein *et al.* [222].

7.1.3 GPU-accelerated Image-based Registration

With increasing performance,programmability and recently introduced high level programming languages, GPUs have become increasingly popular for implementation of image-based registration algorithms. In [92] Khamene *et al.*, use GPU-accelerated DRR generation to speed up 2D/3D registration for positioning in radiation therapy. In [93], the authors extend the framework with GPU-accelerated computation of standard similarity metrics. Kubias *et al.* perform the evaluation of the similarity measure on the GPU [108]. For robust registration they simultaneously evaluate eight similarity metrics on the GPU without performance loss compared to the CPU version.

Deformable intensity-based registration is another problem well suited for execution on GPU stream processors. In [198] one of the first implementations of GPU-accelerated deformable registration is presented. A pair of 2D images is non-rigidly registered using a regularized gradient flow approach.

Vetter *et al.* present a GPU-accelerated, learning-based, multi-modal non-rigid registration of 3D volume data [212], implemented in OpenGL and GLSL. The method is evaluated for multi-modal registration of CT to PET. The authors report a speedup of up to five times compared to a CPU version of the method. In [40] an optimized version for DirectX 10 compatible GPUs is presented.

Ruijters *et al.* present a method for GPU-accelerated deformable registration of 2D and 3D image data using cubic B-spline deformation fields [167]. The method is implemented in OpenGL and Cg, using recursive texture filtering to speedup cubic interpolation.

Multi-modal image registration frequently uses intensity statistics-based similarity metrics, such as CR or MI, which require estimates of joint probability distributions of the image intensities. The estimation of the joint probability distributions is mostly based on the discrete histograms of the images. A key problem for an efficient, data-parallel implementation of these metrics is the computation of the histograms directly on the GPU to avoid costly transfers between host and GPU. In [184] Shams *et al.* present two efficient histogram algorithms for Compute Unified Device Architecture (CUDA) compatible GPUs. In [185], the GPU histogram algorithm is further improved and used for GPU-accelerated MI computation. The performance of the framework is evaluated by multi-modal registration of CT, MRI and PET volume datasets from the Vanderbilt image database. A speedup of 50 times compared to a CPU implementation is reported.

For a more detailed survey and discussion of GPU and multi-core accelerated registration algorithms we refer to [186].

7.2 Contributions

7.2.1 Multi-variate Registration

In this chapter we present efficient techniques for simultaneous, rigid, mono- and multi-modal intensity-based image registration. Simultaneous, also denoted multi-variate, registration of multiple images has the potential to improve the registration result by incorporating all available information at once compared to sequential pair-wise registration. However, the increased complexity arising from the simultaneous integration of all images also greatly increases the demands on computational performance, compared to the standard sequential pair-wise registration approach. We address this limitation by implementing the complete registration algorithm on GPU stream processors, greatly accelerating the evaluation of the multi-variate cost function evaluation compared to a regular CPU implementation.

Furthermore, we present a novel, simultaneous, multi-modal algorithm to register multiple 3D US volumes to a single CT volume, for generating large field of view US mosaics. For many clinical scenarios a high quality CT scan of the patient is available. Our method uses the additional information from CT to improve the US-US registration by simultaneous registration of US to US and US to CT data. US-US registration problems caused by small overlap, or lack of features in the overlap volume, can be compensated by the additional information from CT. A variation of the method for US-CT registration introduced by Wein *et al.* [222] is used to estimate the registration parameters by simultaneously computing the registration of all US volumes to each other and to the CT scan.

The methods have been evaluated on several phantom and in-vivo 3D US and CT data set pairs within the scope of this thesis. The methods and first results have been published in [112]

7.2.2 Deformable Registration

Rigid registration can only recover global transformations between two datasets. However, in general tissues are displaced non-rigidly between pre-operative and intra-operative data acquisition, e.g. due to breathing motion, different patient positioning, tissue growth and various metabolic processes.

In this work we focus on GPU-accelerated, dense deformable multi-modal registration of 3D US-CT dataset pairs. Compared to rigid CT-US, deformable CT-US registration is more difficult to solve, due to modality-specific differences in local anatomic appearance can result in incorrect estimated displacements. In this work we combine the methods presented in chapter 6 for GPU-accelerated simulation of ultrasonic imaging effects and volume reconstruction [90] with a GPU-based framework for dense deformable registration. We compute a dense displacement field mapping the source CT image to the target US image. During registration, the 3D US data is simulated from CT by taking the current estimated displacement field into account, to properly address orientation-dependent imaging artifacts. A modified version of the LC^2 similarity metric is used to drive the optimization of the displacement field within a variational deformable registration framework. For optimal computational efficiency, all operations (US simulation, US volume reconstruction, and deformable registration) are completely implemented on the GPU, eliminating any need for costly transfers between host and GPU. We present first results from a number of hepatic cancer patient Freehand 3D US and CT dataset pairs.

To best of our knowledge, this is the first time that a non-linear, purely image-based mapping of CT and 3D B-mode ultrasound is established in a fast, robust and fully automatic manner.

7.3 Simultaneous Registration

Simultaneous rigid registration addresses problem of aligning multiple images to each other, by simultaneously estimating the pose parameters of all images by optimization of a multi-variate cost function.

$$[\hat{\boldsymbol{p}}_1, ..., \hat{\boldsymbol{p}}_N] = \arg \min_{[\boldsymbol{p}_1,...,\boldsymbol{p}_N]} \frac{\sum_{k=1}^{N} \sum_{l=1}^{N} \omega_{(k,l)} \cdot \sum_{\boldsymbol{v} \in \Omega} \Gamma \left[I_k(H_k(\boldsymbol{v})), I_l(H_l(\boldsymbol{v})) \right]}{\sum_{k=1}^{N} \sum_{l=1}^{N} \omega_{(k,l)}} \tag{7.1}$$

The value of the generalized multi-variate cost function, as defined in equation 7.1, is the sum of the pair-wise cost functions Γ for all image pairs (I_k, I_l) over all elements (voxels) v in the overlap domain Ω. Note that in this formulation Ω is chosen as global reference frame and encloses all images. H_k is a rigid transformation that maps an element $\boldsymbol{v}$ in overlap coordinates into the k-th volume. $\omega_{(k,l)}$ is a weighting factor for the contribution of the costs for pair (I_k, I_l) to the global cost function. If one chooses $\omega_{(k,l)} = 1$ for all pairs the total cost function value is equal to the average of the individual costs. In [219], Wachinger *et al.* propose to select $\omega_{(k,l)} = |I_k \cap I_l|$ for US mosaicking, to introduce a weighting factor favoring larger pair-wise image overlap.

For registering multiple images to each other several approaches exist. They can be classified by the use of either direct or indirect cost functions, and pair-wise and simultaneous optimization approaches. Indirect approaches first pre-process the 3D image data to extract features for registration, in contrast direct approaches directly use the complete image intensity information. Pair-wise approaches decompose the global registration problem into several sequentially executed pair-wise registration problems, which are then used to compute the global registration by propagation of transformations. Simultaneous approaches directly minimize the cost function by simultaneously solving for all parameters and are more stable towards pair-wise local minima. However, they also greatly increase the computational effort per cost function evaluation. For symmetric cost functions, equation 7.1 can be simplified to

$$[\hat{\boldsymbol{p}}_1, ..., \hat{\boldsymbol{p}}_N] = \arg \min_{[\boldsymbol{p}_1,...,\boldsymbol{p}_N]} \frac{\sum_{k=1}^{N} \sum_{l=k+1}^{N} \omega_{(k,l)} \cdot \sum_{\boldsymbol{v} \in \Omega} \Gamma \left[I_k(H_k(\boldsymbol{v})), I_l(H_l(\boldsymbol{v})) \right]}{\sum_{k=1}^{N} \sum_{l=k+1}^{N} \omega_{(k,l)}} \tag{7.2}$$

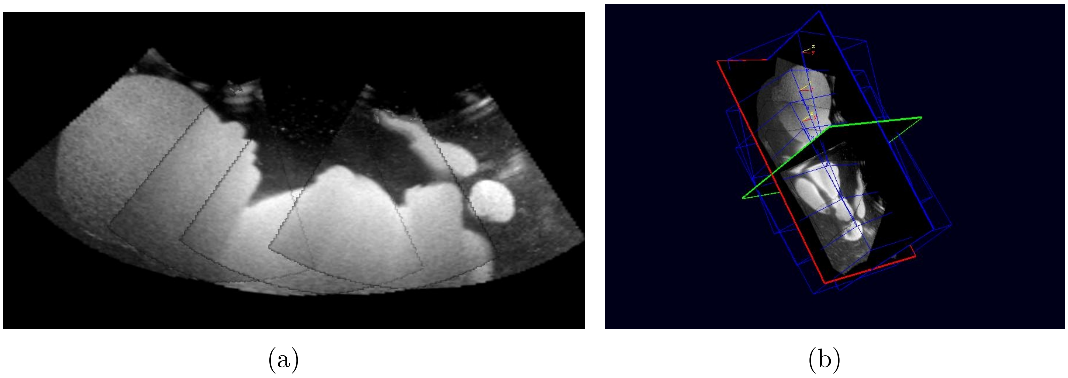
which reduces the evaluation costs significantly.

Mosaicking for large 2D panorama image generation, also denoted panoramic stitching, has attracted much attention in the Computer Vision community. Several related works propose advanced mosaicking algorithms for generating high resolution, seamless, panoramic 2D images from a large number of individual, partially overlapping 2D images [22, 201]. Special emphasis is put on the compositing function and how to improve local resolution by combining information from multiple images or reducing motion artifacts and compensate for low image quality. Inclusion and adaption to 3D US data of these compositing methods has not been addressed in this work, and is one of the possible directions for future research.

Solving the registration problem is only the first step in generating a single mosaic volume with a large field of view. For mosaicking the reference space, enclosing all individual volumes, is discretized into a regular Cartesian grid. For every grid element the mosaic volume intensity values are determined by incorporating the intensity information from all involved, individual volumes. Within this work we focus on accelerating and improving registration algorithms, for mosaicking we used standard mean accumulation. However, generating high resolution, high quality mosaics from several 3D US volumes is challenging itself for multiple reasons: (1) Image quality and voxel intensities of the same anatomical regions can vary for different US scans due to different scan directions, scanning positions, e.g. resulting in occlusions and shadowing artifacts. Hypoechogenic regions can cause negative shadows in scan direction. (2) Motion, either induced by pressure of the transducer onto the patient, internal organ motion, and of course breathing motion may cause geometric variations in the images. (3) Furthermore, US system internal signal processing, noise, speckle and local imaging controls (mainly time gain compensation) may cause variations for different scans.

7.3.1 Applications

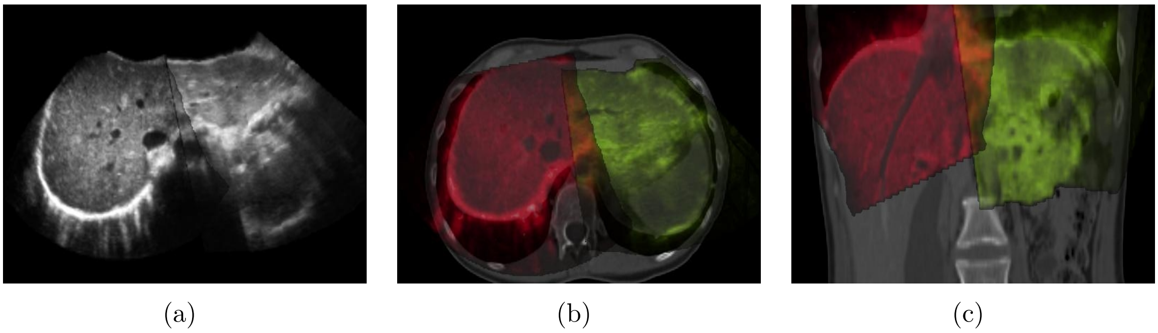
The methods for GPU-accelerated simultaneous registration of multiple 3D US volumes have been evaluated for three different applications.



(a) (b)

Figure 7.1: Exemplary mono-modal simultaneous registration results for four 3D US volumes of a baby phantom. (a) 2D MPR Visualization, (b) 3D MPR visualization of fused 3D volume.

- *Mono-modal US Mosaicking:* Rapidly generating large field of view US mosaics is of interest for many medical procedures, especially for cardiac and obstetrics applications. Fully automatic, robust and rapid mono-modal mosaicking is possible if the individual US volumes have enough overlap and intensity consistency can be assumed. In this thesis we have successfully evaluated mono-modal simultaneous registration on 3D US scans of a baby phantom (e.g. see figure 7.1) and Freehand 3D US of human liver anatomy. The GPU-acceleration techniques discussed in this chapter allow for automatic registration of multiple 3D US scans, requiring no pre-alignment for initialization, within few seconds compared to several minutes of a CPU implementation. For details see section 7.3.3.1.

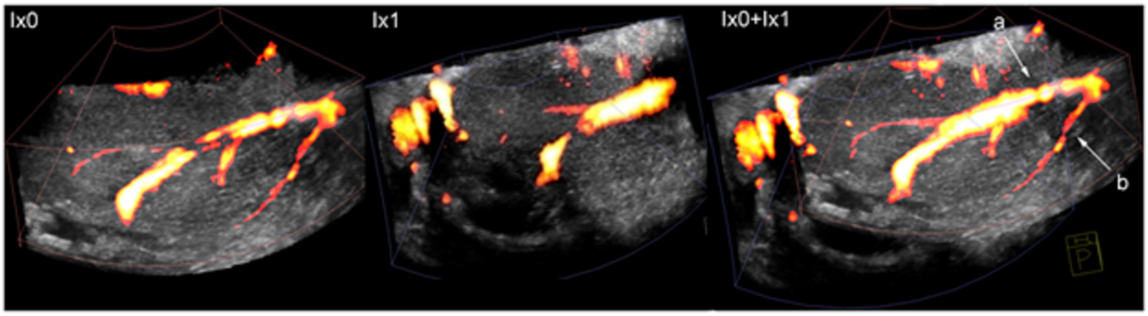


(a) (b) (c)

Figure 7.2: (a) Mosaic from two volumes with very small overlap in the middle. Registration works only if CT information (b,c) is used to constrain the registration. Use of mono-modal approach will lead to breakup of the volume configuration.

- *Multi-modal US Mosaicking:* Variations in US brightness per volume, variable pair wise image overlaps and view dependent artifacts such as occlusion of anatomical structures in one or multiple 3D US volumes can inhibit successful mono-modal registration. For many clinical cases, especially interventions, additional high quality CT data is available. In this work we introduce multi-modal CT-US mosaicking to overcome limitations of mono-modal US mosaicking. Additional global anatomical reference from available CT scans improves registration of multiple 3D US volumes by simultaneously registering US to US and CT to US.

Figure 7.2 depicts an example of a challenging registration problem that benefits from the proposed algorithm. Using mono-modal registration alone the two 3D US volumes fail to register, due to a very small overlap volume in the area of the spinal column. The problem of the small overlap volume is furthermore worsened by shadowing artifacts caused by the vertebrae in each 3D US volume. By registering the two US volumes at the same time to each other and to the CT data set enough anatomical information is available for successful registration.



(a)

Figure 7.3: Exemplary results for Power Doppler Volume registration. Left and middle image depict individual volumes. Right image depicts registration volume (a) arrow points on the main common femoral, while (b) points on the profunda femoris artery, note improved clear visualization of vessel bifurcations.

- *Power Doppler Volume Registration:* The developed methods have also been adapted and evaluated for automatic registration of 3D US Power Doppler volumes (see figure 7.3) acquired with a novel multi US device cuff. The US device is developed by Siemens Ultrasound within the DARPA founded Deep Bleeder Acoustic Coagulation (DBAC)[1] program. The aim is to provide a US technology powered device to save the life of soldiers wounded in combat situations by stopping life-threatening bleeding. US is used for both detection and to stop internal bleedings quickly by focused US energy to increase blood coagulation. Rapid, automatic registration of multiple US Power Doppler is a key requirement of the envisioned system to generate a single 3D US of the vascular structures and to facilitate automatic bleeding detection by successive image processing and segmentation algorithms.

---

[1] http://www.darpa.mil/STO/smallunitops/dbac.html

7.3.2 Methods

7.3.2.1 Similarity Measures

Beside the optimization algorithm, the multi-variate similarity measures form another core component of the registration algorithm. Wachinger *et al.* first proposed the extension of standard pair-wise similarity measures to the multi-variate case for the simultaneous registration of multiple images. For detailed discussion of deriving the multi-variate version from the bi-variate one we refer to [216, 219]. In the following listing we briefly describe the similarity measures supported for registering images of the same or different modalities without introducing special extensions to the similarity measures. We will use I and J to denote the two images used in pair-wise similarity measures and $I_1, ..., I_N$ for the images used in the multi-variate case. $\nabla I(x)$ denotes the gradient of image I at sample position x, estimated using either central differences or Sobel operators.

- *Mean sum of absolute and squared Differences:*

  $$\mathrm{SD}(I,J) = \frac{1}{N}\sum_{k=1}^{N} ||I_k - J_k||^n \tag{7.3}$$

  Sum of absolute differences (SAD) (the L_1 norm of the difference image; $n = 1$ in equation 7.3) and SSD (the L_2 norm of the difference image; $n = 2$ in equation 7.3) are well known, standard similarity measures. Both measures assume an identity intensity relationship between the two images and that the intensities are only subject to zero mean Gaussian noise. Optimal alignment is achieved when the intensity difference error is minimal. Because of that property both are also frequently used as error measures. SSD and SAD are mainly used for registering images of the same modality and are highly sensitive to structures present only in one of the images, e.g. surgical instrument, implant. The multi-variate version of SAD/SSD is defined as given in equation 7.4.

  $$\mathrm{SD}_{mv}(I_1, ..., I_N) = \frac{\sum_{k=1}^{N-1}\sum_{l=k+1}^{N}\ \sum_{\boldsymbol{v}\in\Omega} ||I_k(H_k(\boldsymbol{v})) - I_l(H_l(\boldsymbol{v}))||^n}{\sum_{k=1}^{N}\sum_{l=k+1}^{N}\omega_{(k,l)}} \tag{7.4}$$

- *Normalized Cross Correlation (NCC):* In general the assumption of an identity intensity relationship and a mean zero Gaussian noise added to the images intensities does not hold. Thus, in practice often a linear relationship: $J[\boldsymbol{p}] = s * I[{}^{I}\boldsymbol{H}_J * \boldsymbol{p}] + t$, expressed by a scaling and bias of the intensities in the images is assumed. One of the most popular and commonly used similarity measure for registering images with an assumed linear relationship is NCC (see equation 7.5)

  $$\mathrm{NCC}(I,J) = \frac{\sum_{k=1}^{N}\left(I_k - \overline{I}\right)\left(J_k - \overline{J}\right)}{N\sigma_I\sigma_J} \tag{7.5}$$

  with $\overline{I} = \frac{1}{N}\sum_k^N I_k$ and $\sigma_I = \sqrt{\frac{1}{N}\sum_k^N (I_k - \overline{I})^2}$ and $\overline{J} = \frac{1}{N}\sum_k^N J_k$ and $\sigma_J = \sqrt{\frac{1}{N}\sum_k^N (J_k - \overline{J})^2}$.

The mean intensities $\overline{I}$ and $\overline{J}$ and the square root of the statistical variances σ_I and σ_J have to be computed first before equation 7.5 can be evaluated. It is possible to compute all components in a single traversal of the overlap domain by accumulation of six temporary sums, that are then used to compute the final value. The value of NCC will go towards 1.0 for an increasing linear relationship and towards -1.0 for a decreasing one. If the two images are linearly independent their correlation, the value of NCC is around 0.0.

Equation 7.6 depicts the multi-variate version of NCC.

$$\mathrm{NCC}_{mv}(I_1, ..., I_N) = \frac{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)} \mathrm{NCC}(I_k, I_l)}{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)}} \tag{7.6}$$

Local Normalized Cross Correlation (LNCC) is an extension of the standard NCC measure which aims to overcome the problem of NCC for a spatially varying linear relationship between the image intensities. For the computation of LNCC (see equation 7.7) the overlap domain is split into several equally sized regions and the sum of all locally per region computed NCC values is returned as the final value. The individual regions can be overlapping or non-overlapping, and in the extreme case a single region can be centered on each image pixel.

$$\mathrm{LNCC}^r(I, J) = \frac{1}{M} \sum_{n=1}^{M} \mathrm{NCC}^r(I, J) \tag{7.7}$$

with M equal the number of regions and r the region size in every image dimension.

Because of the local correlation estimation LNCC can cover more than the simple linear relationship assumed by NCC, and can also be used for registering images of different modalities. To a certain degree LNCC is able to successfully register pairs of images with objects present in only one of the two images, e.g. pair of pre-operative image and post-operative images of the same patient with surgical implant present in post-operative image. Variations of the algorithm exists that selectively include region NCC values into the total sum based on actual overlap region size or weight the individual values with the local variance of one of the images [114]. The major drawback of LNCC is the relatively high computational costs when using a large number of local regions. This is even more a performance problem for the multi-variate version (see equation 7.8), as not only local samples per region, but local samples times the number of valid pairings of the N images have to be computed. Thus fast filtering and interpolation methods are required.

$$\mathrm{LNCC}^r_{mv}(I_1, ..., I_N) = \frac{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \mathrm{LNCC}^r(I_k, I_l)}{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)}} \tag{7.8}$$

- *Normalized Gradient Fields (NGF):* The NGF similarity measured is motivated by the following definition of similarity: *"Two images are considered similar, if intensity changes occur at the same locations"* [58]. Instead of the image intensities, the dot product of the normalized gradient vectors is used for computing the similarity metric. The dot product of two unit-length vectors is 1.0 if the vectors have the same direction or -1.0 if they are parallel but are pointing into opposite directions. For all other cases the dot product magnitude is between zero and one. In equation 7.9 the sum of all per overlap domain element evaluated local dot products defines the total NGF similarity measure.

$$\mathrm{NGF(I,\ J)} = \frac{1}{N}\sum_{k=1}^{N}\left\langle \frac{\nabla I_k \nabla J_k}{\|\nabla I_k\|_\varepsilon \|\nabla J_k\|_\varepsilon} \right\rangle^2 \tag{7.9}$$

with $\|\nabla I\|_\varepsilon := \sqrt{\nabla I^T \nabla I + \varepsilon^2}$ and $\varepsilon = \frac{\eta}{N}\int_\Omega |\nabla I(x)| dx$ where η is the estimated noise level in the image I and N is the number of elements in the overlap domain Ω.

In case of correct alignment the sum will be close to 1.0. The ε component is added to remove regions with very small gradient magnitude, e.g. constant intensities in homogeneous image regions. In essence it serves as a threshold to control when to consider a change in intensity as significant. If ε is much larger than the magnitude of the gradients, then $\|\nabla I\|_\varepsilon$ will be close to zero with little to no effect on the total result.

A strength of NGF is that it does not make implicit assumption about pixel intensity relationships as other similarity measures do and is thus well suited to deal with mono- and multi-modal registration problems. An additional benefit are its cost function smoothness properties, with very few local optima [58]. However, the use of gradient vectors rather than intensity values to determine image similarity is also one of the major limitations in case of low quality images, e.g. high level of noise, or orientation dependent artifacts. Thus, robust estimation of the image gradients and fine tuning of the parameter ε becomes very important.

$$\mathrm{NGF}_{mv}(I_1, ..., I_N) = \frac{\sum_{k=1}^{N-1}\sum_{l=k+1}^{N}\omega_{(k,l)}\ \mathrm{NGF}(I_k, I_l)}{\sum_{k=1}^{N-1}\sum_{l=k+1}^{N}\omega_{(k,l)}} \tag{7.10}$$

A performance limitation for the multi-variate version of NGF (see equation 7.10) is the necessity to recompute the gradient fields for every image in every iteration, as all images are transformed simultaneously.

7.3.2.2 Multi-modal Simultaneous US-CT Registration

The similarity measures described in the preceding section directly allow for simultaneous, mono-modal registration of multiple 3D US volumes. To incorporate CT information into the registration algorithm a variation of the method and similarity measure presented by Wein *et al.* [222] is used. In every evaluation of the similarity measure the major US reflections are simulated from CT data and reconstructed into a Cartesian volume for each real US scan. The simulation of the US reflections is handled by the US simulation operator θ which computes the simulated US reflection data $US_k = \theta(CT, US_k)$ from CT and parameters (e.g. US probe position and orientation, US field of view) of the real US scan US_k. Simulated and real Freehand 3D US data is compounded on-the-fly on the GPU using the method presented by Karamalis *et al.* [90].

The LC^2 similarity measure proposed by Wein *et al.* [222] for US-CT registration proved numerically unstable on recent GPUs using GLSL due to lack of double precision floating point support. However, the simulated major reflections provide sufficient information for stable US-CT registration for US scans with large tissue interfaces. Therefore, we chose φ, the weighted sum of the multi-variate NCC of all 3D US data, and real 3D US and simulated US reflections (see equation 7.11).

$$\begin{aligned}\varphi = & \frac{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)} \mathrm{NCC}\left(US_k, US_l\right)}{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)}} \\ & + \alpha \frac{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)} \mathrm{NCC}\left(\theta(CT\ , US_k), US_l\right)}{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)}}\end{aligned} \tag{7.11}$$

For final refinements after registration one can exchange the global NCC in φ with LNCC. LNCC is not selected right from the start as NCC is stable enough for aligning the major features in US-US and US-CT data from the initial pose.

7.3.2.3 Automatic Simultaneous Registration of 3D Power Doppler Data

For registering multiple 3D Power Doppler volumes of vascular structures the standard similarity measures presented in section 7.3.2.1 are not sufficient. In this section we present a voxel-based similarity measure that uses binary vasculature and distance transform volumes to register the Power Doppler Data. In our method the data is first pre-processed to obtain two volumes for each input volume. First a segmentation algorithm separates the voxels of the vessel tree from the background data and noise to obtain a binary volume containing only the vessel tree. Then a distance transform is performed on the binary vessel tree volume, which encodes for every voxel the distance to the closest vessel tree voxel.

For registering the individual volumes the following cost function is minimized:

$$\frac{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \sum_{\boldsymbol{v} \in \Omega} D_k(H_k(\boldsymbol{v}_l))}{\sum_{k=1}^{N-1} \sum_{l=k+1}^{N} \omega_{(k,l)}} \tag{7.12}$$

where $v_i \in V_i$ is a voxel that lies on the vessel in the i-th dataset and , and D_k is the distance transform volume for the k-th dataset. $D_k(H_k(\boldsymbol{v}_i))$ returns the closest to vessel distance for the k-th dataset, of a voxel that lies on a vessel in the i-th dataset. When all Doppler Doppler volumes are correctly aligned the cost function is minimal.

7.3.2.4 Transformation Model and Optimization

During registration for each 3D US volume the transformation from local volume coordinate frame to overlap coordinate frame Ω is optimized. Per volume global transformations are used. For most cases 6 Degrees of Freedom (DOF), rigid transformations are sufficient to estimate the global alignment. However, to compensate for variations in scale and shearing due to speed of sound, motion, scan conversion, and compounding we selectively support up 12 DOF during optimization.

To estimate the pose parameters simultaneously for all 3D US volumes a Downhill Simplex Optimizer is employed. Alternatively a Best Neighbor or Powell Brent Optimizer can be used. Downhill Simplex was chosen from the three optimizers based on an empiric evaluation for a number of registration problems based on number of successful registration and speed of convergence.

To speed up convergence of the registration, a multi-resolution volume pyramid is used. The registration problem is first solved on higher levels of the pyramid coarsely aligning resolution reduced volumes. Successive, iterative registrations for the lower pyramid levels use the result of the previous level for initialization and refine the registration parameters for resolution increasing volumes.

In contrast to conventional pair-wise image-based registration, in general the pose parameters of all volumes are optimized during simultaneous registration. This can result in slow convergence or long computation times if e.g. too large step sizes are chosen resulting in an increased number of cost function evaluations. To that end we added two modifications to the optimization. (1) A transformation penalty term is used to penalize physically impossible or very unlikely transformations per volume. The penalty can be activated and parametrized selectively per volume. Especially for 3D US volumes acquired with a pose known from a tracking system, the penalty term can speed up the registration. (2) In the case of simultaneous, multi-modal CT-US registration only the transformation parameters of the 3D US volumes are optimized. The CT coordinate frame is defined to coincide with the global reference frame, thus it is already considered registered.

7.3.2.5 GPU Implementation

Evaluation of the multi-variate, intensity-based similarity measure for N volumes is the computationally most expensive part of the registration. For similarity measure evaluation the global reference frame Ω is discretized into a regular Cartesian grid of dimension $\mathbf{X}$ x $\mathbf{Y}$ x $\mathbf{Z}$, the element spacing is selected based on the smallest element spacing of the individual 3D US volumes. Each grid element represents one $v \in \Omega$ as described in equation 7.1. Every grid element has to be considered during evaluation and up to N interpolations from the individual volumes have to be performed. Namely the large number of non-local volume accesses for interpolation limits the performance of the algorithm for CPU implementations.

In this work we have implemented the multi-variate similarity measures for mono-modal US-US (see section 7.3.2.1), multi-modal US-CT (see section 7.3.2.2), and Power Doppler volume registration (see section 7.3.2.3) on the GPU.

Each similarity measure is implemented within a GLSL program. For optimal efficiency the GLSL program vertex and fragment shader source code strings are generated by the registration application program itself at runtime. By using this GPU program factory technique optimal source code for any combination of number of volumes and selected similarity measure is created.

For the implementation of the actual similarity measure evaluation we distinguish between local and non-local similarity measures, e.g. NCC versus LNCC.

Globally evaluated similarity measures are computed by rendering **Z** parallel quadrilaterals, using orthographic projection and enabled additive blending to a viewport of resolution **X** x **Y** in a single render pass. Each quadrilateral covers exactly one slice along the Z-axis of the global reference volume. For every generated fragment the reference grid position $\boldsymbol{v}$ is computed based on slice index and 2D viewport position. The individual volumes are sampled by transforming $\boldsymbol{v}$ to the local volume coordinate systems and the temporary results for every valid volume pair are written to one of multiple attached 2D render targets. After this first rendering pass the similarity measures is already partially evaluated along the Z-axis of the reference frame. In a second rendering pass a reduction shader accumulates the results to yield the final similarity measure value by combining and weighting the values from the 2D render targets of the previous render pass.

We have experimented with an alternative implementation very similar to orthographic ray-casting. In this approach only a single quadrilateral is rendered activating a fragment shader for that marches through the reference grid along the Z-direction in a loop. Essentially the same result as in the used implementation is computed. However, reduced number of parallelism, **X** x **Y** versus **X** x **Y** x **Y** fragments, and increased GPU program complexity did result in a decreased performance compared to the slicing and blending approach.

Naive, straightforward implementations of locally evaluated similarity measures can result in extremely long computation times due to inefficient volume sampling strategies. For the similarity metric evaluation of a local k^3 neighborhood up to $N * k^3$ interpolations have to be performed (for N overlapping volumes at the current sample position). Without employing some scheme to reuse sampling and computation results from neighboring grid positions, a huge number of repeated texture accesses and computations is performed when traversing the global reference grid, imposing an overall performance limitation, especially for larger neighborhoods.

Recent GPU APIs, e.g. CUDA or OpenCL, offer shared memory accessible by a group of hardware threads to allow for communication of local computation results between threads. However, GLSL does not provide a similar mechanism to share memory between a group of fragment shader instances. In our implementation we therefore implement a semi-recursive local similarity measure evaluation.

A single quadrilateral is rendered to the viewport activating the aforementioned ray-marching shader. While marching along the ray the local similarity measures is evaluated for every voxel. However, local memory (arrays) per ray inside the fragment shader is used to store the neighborhood samples values per volume. Thus, instead of sampling a k^3 for every volume, we perform $2 * k^2$ samples for every volume with z-offsets $\pm(\lceil k/2 \rceil + 1)$ from the current sample position. The samples with offset $(\lceil k/2 \rceil + 1)$ are added to the set of currently stored local samples, while the ones with offset $-(\lceil k/2 \rceil + 1)$ are removed. The local similarity measure is re-evaluated for the current position and added to a variable storing the per-ray similarity measure results. The results of all rays are written to a number of rendering targets and reduced in a successive rendering pass to yield the final global similarity measure value. Although this techniques does not completely eliminate multiple accesses to the same sample locations, it improves performance considerably compared to the brute-force, per voxel local implementation.

7.3.3 Results

The simultaneous registration framework was evaluated for a number of real and phantom datasets for all three applications discussed in this chapter. In the following section we present the results for evaluating the registration stability and performance of mono-modal and multi-modal simultaneous registration. For all experiments a computer equipped with an Intel Core Duo processor (2.6Ghz), 4Gbyte RAM and a NVIDIA Geforce GTX 280 with 1GByte of VRAM was used. The methods developed for simultaneous registration of multiple 3D Power Doppler volumes was evaluated externally by Siemens Corporate Research, Princeton and partners in a series of phantom and in-vivo swine experiments. Thus, we do not present detailed results within the scope of this thesis. Figure 7.6 depicts an exemplary result for registration for two sets of Power Doppler scans.

7.3.3.1 3D US-US and US-CT Registration

Mono-modal, simultaneous registration of multiple US volumes was evaluated on phantom datasets. Four 3D US scans, each 256x256x256 voxels, of a baby phantom (see figure 7.1) were used in the registration stability study. The volumes were acquired using a 3D US Wobbler probe, that was moved in cranio-caudal direction over the phantom, without a major view direction change. Therefore the intensity variation in the individual scans is mostly due to noise. An optimization of translation and rotation parameters was performed 100 times with random offsets of +/- 10mm and +/- 5 degrees from a defined ground truth pose. SSD was used as similarity measure.

Vol.	tx [mm]	ty [mm]	tz [mm]	α [rad]	β [rad]	γ [rad]
1	3.2134	3.0474	3.1312	0.0183	0.0288	0.0204
2	3.1112	3.0478	3.2231	0.0198	0.0269	0.0212
3	3.1507	3.0117	3.1160	0.0198	0.0269	0.0206
4	3.1338	3.2669	3.3408	0.0176	0.0302	0.0227

Table 7.1: Simultaneous registration stability experiment results for four 3D US phantom scans. Standard deviations for translation and rotation parameters for random study from defined ground truth pose.

Table 7.1 depicts the results for the standard deviation of the estimated position and orientation parameters from the ground truth ones for 100 test runs.

The same stability study was performed for registration of four patient Freehand 3D US scans of the liver region and one abdominal CT scan. Each dataset had a resolution of 256x256x106 voxels. Figure 7.4 depicts three exemplary 2D MPR visualizations from the mosaic volume. For the multi-modal registration NCC was selected as similarity measure.

Table 7.2 depicts the standard deviations for the translation and rotational parameters for the patient dataset. The results show that mono- and multi-modal direct simultaneous registration performs as well on phantom as on real patient data.

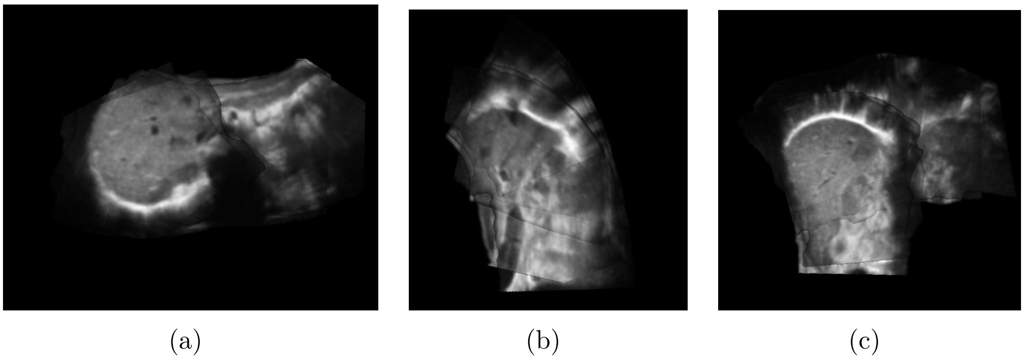

(a) (b) (c)

Figure 7.4: Axial along mosaic volume generated 2D MPRs from four Freehand 3D US scans of human liver.

Vol.	tx [mm]	ty [mm]	tz [mm]	α [rad]	β [rad]	γ [rad]
1	3.2829	3.3165	3.5544	0.0212	0.0331	0.0230
2	3.2813	2.9114	2.9758	0.0173	0.0304	0.0232
3	3.2195	3.2878	3.8525	0.0176	0.0319	0.0259
4	3.2345	2.8773	3.1018	0.0170	0.0311	0.0243

Table 7.2: Simultaneous registration stability experiment results for four Freehand 3D US scans and one CT scan of the same patient. Standard deviations for translation and rotation parameters for random study from defined ground truth pose.

Another aspect of interest is the performance of the registration methods. In a second experiment we evaluated the registration speed for both mono-modal and multi-modal registration. The performance of the mono-modal registration was evaluated on the Baby phantom test dataset for varying volume resolutions and compared to a straightforward, non-optimized and single-threaded CPU implementation. Thus, the CPU implementation presents the lowest performance baseline of to expect CPU performance. Of course, a carefully optimized and multi-threading enabled CPU implementation will perform better, but will also require a lot of manual code tuning and profiling.

The multi-modal registration performance was evaluated for two US/CT volume configuration. The first configuration consist of two Freehand 3D US scans and one CT scan, the second configuration adds two more 3D Freehand US scans.

Table 7.2(a) and table 7.2(a) depict the performance of the framework for both mono and multi-modal registration. Table 7.2(a) shows an impressive speedup of more than 400 times of the GPU compared to the CPU for the mono-modal registration problem. Another interesting finding is that the GPU performance is barely affected by an increase in volume resolution than the CPU performance. We consider the drastically decreasing CPU performance in case of increasing volume resolution to be caused by the huge increase in additional interpolation operations for the also much finer sampled reference frame. The massive parallel architecture of the GPU and its hardware accelerated texture sampling can be attributed with the almost constant performance of the GPU.

The multi-modal registration results show that the method is applicable but not as fast

(a) GPU vs CPU Performance for Baby Phantom datasets

	GPU [sec]	CPU [sec]
4 x 256^3	11.19	> 36,000.00
4 x 128^3	9.71	4006.34
4 x 64^3	7.11	397.21

(b) Performance of multi-modal registration for two US/CT configurations

Volume Config.	Time [sec]
1 CT, 2 US	333.93
1 CT, 4 US	1076.57

Table 7.3: (a) Performance of mono-modal registration for four 3D US Baby phantom scans. Evaluation for several resolutions and comparison with performance of CPU implementation. (b) Computational performance of simultaneous multi-modal registration for two and four US volumes with one CT scan.

yet as envisioned. The reduced performance of the multi-modal case is to a large part due to the additional simulation and volume compounding steps for every similarity measure evaluation. Additionally the similarity measure described in section 7.3.2.2 results in much more complex GPU program code than the simple SSD measure used for the mono-modal test case.

Additionally, the multi-modal registration was evaluated qualitatively by visual inspection of alignment of major anatomical regions, e.g. organs interfaces, vasculature inside the liver. The improvement of alignment before and after registration is clearly visible (see figure 7.5). We validated the improved registration for problematic configurations by a test case with two US volumes with very small overlap and one CT scan (see figure 7.2). Using only US intensity information the registration fails and breaks the link of the US volumes. Using the proposed simultaneous registration method, the US volumes are registered successfully to each other and to the CT data (see figure 7.5).

7.4 Deformable CT-US Registration

In this section we present a framework for GPU-accelerated CT-US registration using a variational approach. The algorithm requires a pair of CT and 3D US volume of the same resolution that have already been rigidly registered. Our method computes a dense displacement field mapping the 3D CT to the 3D US volume. In the following section we will give a brief overview on the general framework for GPU-accelerated deformable registration, which is closely related to the one presented by Chefd'Hotel *et al.* [25]. Followed by a presentation of the extension and changes that were made for deformable CT-US registration.

7.4.1 Methods

7.4.1.1 Variational-based Dense Deformable Image Registration

Deformable image registration seeks to find a displacement function $\boldsymbol{u} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, which defines a deformation field $\varphi = Id + \boldsymbol{u}$ which warps the source image S, such that it is aligned to the target image T. Thus, the problem becomes a minimization of the following energy function:

$$\hat{u} = \arg\min_{\boldsymbol{u}} \mathcal{D}(T, S \circ (\mathrm{Id} + \boldsymbol{u})) \tag{7.13}$$

in which $\mathcal{D}$ is an appropriate difference measure. Regularity of the deformation u is an important property and can be achieved by various methods. One option is to include a regularization term directly into the energy formulation. For an overview of the possible regularization and optimization options compare [137]. Further regularization can be achieved by parameterization of the deformation, e.g. using B-Splines [165]. For this work, we follow the approach of [25], which does not require an explicit regularization term in the energy formulation. The deformation is constrained approximately to the manifold of diffeomorphisms by computing the result as a consecutive composition of small update fields. The updates are computed by computing the gradient of the energy function and smoothing it by applying a low-pass filter. The solution to the energy minimization problem (see equation 7.13) is found by iteratively computing an update du to the initial displacement field estimate $u^{(i+1)}$ over iterations i in the following form

$$\begin{aligned} du &\leftarrow -G_\sigma(\nabla\mathcal{D}(T, S \circ (\mathrm{Id} + \boldsymbol{u}^{(i)}))) && (7.14) \\ \boldsymbol{u}^{(i+1)} &\leftarrow \boldsymbol{u}^{(i)} \circ (\mathrm{Id} + \tau du) && (7.15) \end{aligned}$$

Here, G_σ is a Gaussian filtering operation with standard deviation σ, $\nabla\mathcal{D}(T, S \circ (\mathrm{Id} + \boldsymbol{u}^{(i)}))$ is the derivative of the similarity measure, Id is the identity transformation, and the positive scalar τ is the update magnitude control parameter.

The displacement field is computed by successive composition of small update fields. Each update field is computed by the weighted sum of the gradient of the (dis-) similarity measure and the gradient of the regularization operator. As all computations are evaluated only locally on a per voxel-basis, the iterative approach lends itself very well to a straightforward, parallel implementation on GPU stream processors. For speeding up the convergence, a multi-resolution image pyramid can be used. This aids especially to speed up the computation of large displacements. The computation of the displacement field can be aborted if the current update field contains only displacement vectors with a magnitude smaller than a selected threshold or if the number of iterations is exceeded.

7.4.1.2 Extensions for CT-US Registration

The challenging task of deformable CT-US registration requires some modifications to the general deformable registration framework described in the previous section. In the following paragraphs we will describe the additions. For the remainder of this section the source image S refers to the CT image, and the target image T refers to the US image.

Ultrasound Simulation from CT In [222], a ray-based model for patient-specific US simulation from CT for rigid US-CT registration is presented. The model is based on the assumption that the acoustic impedance of human tissue is proportional to its density, which in turn relates to the CT Hounsfield measurements from CT data. To enable a better assessment of structural alignment in CT and ultrasound, we simulate two ultrasonic imaging effects from CT, as previously described in [222] using the framework for GPU-accelerated US simulation described in chapter 6.

Let $\mu : \Omega_T \mapsto \mathbb{R}$ be the volume of original CT intensity values (i.e. Hounsfield units). Using the model of Wein *et al.* the reflection component image S_1 is computed by

$$\begin{aligned} S_1(\boldsymbol{x}) = \log \Big[a I_0 \exp \Big(- \textstyle\int_0^{\lambda_x} \Big(\frac{|\nabla \mu(\boldsymbol{x_0} + \lambda \boldsymbol{d})|}{2\mu(\boldsymbol{x_0} + \lambda \boldsymbol{d})} \Big)^2 d\lambda \Big) \\ \Big(\boldsymbol{d}^T \nabla \mu(\boldsymbol{x}) \Big) \frac{|\nabla \mu(\boldsymbol{x})|}{(2\mu(\boldsymbol{x}))^2} + 1 \Big] \frac{1}{\log(a+1)} \end{aligned} \tag{7.16}$$

In words, for a voxel location $\boldsymbol{x}$, the reflection S_1 can be computed by integrating over reflection and transmission along an ultrasonic scan line originating at $\boldsymbol{x_0}$ with direction $\boldsymbol{d}$. It is embedded in a log-compression with parameter a. This creates a reasonable simulation of ultrasound reflections, including shadowing effects (i.e. after a strong reflection the remainder of that scan line will have no reflection).

The second imaging effect is tissue echogenecity, which results from the scattering behavior of tissue inhomogeneities, that are smaller than the ultrasound pulse wavelength. It is approximated by a heuristic mapping of the CT soft tissue intensity range, and modulated with the ultrasonic transmission from the integral in equation 7.16, in order to clear its value if shadowing has occurred. We define this second term as $S_2(\boldsymbol{x})$.

Because those simulated values S_1 and S_2 are derived from line integrals through the CT volume, they implicitly depend on the entire deformation field $\varphi = \mathrm{Id} + \boldsymbol{u}$. For S_1, this results to

$$\begin{aligned} S_1(\boldsymbol{u}, \boldsymbol{x}) = \log \Big[a I_0 \exp \Big(- \textstyle\int_0^{\lambda_x} \Big(\frac{|\nabla \mu(\varphi(\boldsymbol{x_0} + \lambda \boldsymbol{d}))|}{2\mu(\varphi(\boldsymbol{x_0} + \lambda \boldsymbol{d}))} \Big)^2 d\lambda \Big) \\ \Big(\boldsymbol{d}^T \nabla \mu(\varphi(\boldsymbol{x})) \Big) \frac{|\nabla \mu(\varphi(\boldsymbol{x}))|}{(2\mu(\varphi(\boldsymbol{x})))^2} + 1 \Big] \frac{1}{\log(a+1)} \end{aligned} \tag{7.17}$$

Back to the variational notation introduced before, our source image becomes a two-channel representation $S : \Omega_S \mapsto \mathbb{R}^2$, whose voxel values are implicitly dependent on the whole deformation field, i.e.

$$T(\mu, \varphi, \boldsymbol{x}) \doteq \Theta_{\mu,\varphi}(\boldsymbol{x}) \tag{7.18}$$

Θ is therefore our "ultrasound simulation operator" which computes S_1 and S_2 given a CT volume μ and deformation field φ.

Similarity Metric In [222], the LC^2 similarity metric is proposed, which is independent of how much the simulated channels S_1 and S_2 locally contribute to the actual ultrasound intensity T. However, the parameters describing this intensity contribution are solved in a least-squares sense for every local evaluation of the metric. This makes it difficult to analytically derive its gradient; besides, solving for the missing parameters at every pixel location causes numerical problems on our implementation platform (i.e. limit of 32-bit floating-point precision on GPUs). We therefore use the following approximation for $\mathcal{D}_{\text{CT-US}}$:

$$\mathcal{D}_{\text{CT-US}}(I_{US}, I_{CT}, \varphi) = -\mathcal{D}_{\text{LCC}}(I_{US}, S_1 \circ \varphi) - \mathcal{D}_{\text{LCC}}(I_{US}, S_2 \circ \varphi) \tag{7.19}$$

where $\mathcal{D}_{\text{LCC}}$ is the local cross-correlation (LCC) measure, defined in [71] for images I_1 and I_2 as

$$\mathcal{D}_{\text{LCC}}(I_1, I_2, \varphi, \boldsymbol{x}) = \frac{v_{1,2}(\varphi, \boldsymbol{x})^2}{v_1(\boldsymbol{x}) v_2(\varphi, \boldsymbol{x})} \tag{7.20}$$

where $v_1(\boldsymbol{x})$ is the variance of I_1, $v_2(\varphi, \boldsymbol{x})$ is the variance of $I_2 \circ \varphi$, and $v_{1,2}(\varphi, \boldsymbol{x})$ is the covariance of I_1 and $I_2 \circ \varphi$ at $\boldsymbol{x}$. The major idea of the LCC measure is that the joint probability of the two images I_1 and I_2, from which the means and variances are computed, is weighted locally by a Gaussian kernel G_γ with standard deviation γ. Therefore the variances and the covariance are dependent on the location in space $\boldsymbol{x}$.

The rationale behind replacing the linear combination from [222] with equation 7.19, is that the change of the individual LCC metrics depends on how much content each simulated channel S_1, S_2 provides. For example, if in a particular voxel neighborhood, no reflection term S_1 contributes to the similarity, it's influence on the displacement update will be minor. Therefore, the sum of two individual LCC measurements can account for the unknown sum of ultrasound reflection and echogenecity.

From equation 7.19, the gradient of $\mathcal{D}_{\text{US-CT}}$ is

$$\nabla\mathcal{D}_{\text{US-CT}}(I_{CT}, US, \boldsymbol{u}) = -\nabla\mathcal{D}_{\text{LCC}}(I_{US}, S_1 \circ \varphi) - \nabla\mathcal{D}_{\text{LCC}}(I_{US}, S_2 \circ \varphi) \tag{7.21}$$

The gradient of the local cross-correlation with respect to φ, at position $\boldsymbol{x}$ in space, is given by [71]:

$$\nabla\mathcal{D}_{\text{LCC}}(\varphi)(\boldsymbol{x}) = L(I_1, I_2, \varphi, \boldsymbol{x}) \nabla I_2(\varphi(\boldsymbol{x})) \tag{7.22}$$

For the case of LCC, the term L is defined as [71]:

$$\begin{aligned} L(I_1, I_2, \varphi, \boldsymbol{x}) = G_\gamma \star \frac{-2}{\mathcal{G}_\gamma} \Bigg[&\frac{v_{1,2}(\varphi, \boldsymbol{x})}{v_2(\varphi, \boldsymbol{x})} \left(\frac{I_1(\boldsymbol{x}) - \overline{I_1(\boldsymbol{x})}}{v_1(\boldsymbol{x})} \right) \\ &-\mathcal{D}_{\text{LCC}}(I_1, I_2, \varphi, \boldsymbol{x}) \left(\frac{I_2(\varphi(\boldsymbol{x})) - \overline{I_2(\varphi, \boldsymbol{x})}}{v_2(\varphi, \boldsymbol{x})} \right) \Bigg] \end{aligned} \tag{7.23}$$

where $\star$ is the convolution operator, $\mathcal{G}_\gamma$ is the integral of the local Gaussian kernel inside the image domain, $\overline{I_1(\boldsymbol{x})}$ is the local mean intensity of I_1 at $\boldsymbol{x}$, and $\overline{I_2(\varphi, \boldsymbol{x})}$ is the local mean intensity of $I_2(\varphi)$ at $\boldsymbol{x}$.

By substituting I_1 with the target T and I_2 with S_1 and S_2, respectively, the gradient of $\mathcal{D}_{\text{CT-US}}$ can be computed.

7.4.1.3 Importance of GPU Implementation

The approach for automatic, deformable CT-US registration presented in this chapter is computationally highly expensive. The performance limiting factors are the re-simulation of the US component images S_1 and S_2 from CT dependent on the current deformation field φ and their reconstruction from scan line to Cartesian space in every iteration of the algorithm. The approach is only made feasible by consequently implementing every aspect of the algorithm on the GPU. Our implementation uses the OpenGL graphics library and shading language (GLSL).

In every major iteration of the algorithm, a 3D Freehand US sweep is simulated [111], followed by the reconstruction of two Cartesian volumes for reflection and echogenecity components [90]. Then, the displacement field update is computed per voxel, smoothed by a low-pass filter, and the source image is warped by the current deformation field. Simulation, reconstruction and registration are entirely computed on the GPU, which completely avoids costly transfers in order to communicate results between the individual sub-algorithms. Convolution performance of current graphics hardware is texture-read bound and can be even worse then recursive convolution on multi-core CPUs for larger kernels. Therefore, the LCC computation utilizes a semi-recursive convolution along the 3D texture slice stack to optimize its performance. Similar to the performance optimization presented in section 7.3.2.5 for simultaneous registration, the costs for computing the local similarity measure value are reduced by reusing in the previous step stored values for a local voxel neighborhood and only updating along the neighborhood borders in Z-direction for the current voxel position $\boldsymbol{x}$.

7.4.2 Results

The deformable registration algorithm was evaluated for pairs of CT and 3D Freehand US scans from 12 different patients with hepatic cancer. See figure 7.7 for an exemplary registration result for patient 1. Simulated and real US data is displayed using color overlay, the displacement field is depicted by a deformed regular grid.

Each patient data comprised a portal-venous phase contrasted CT scan and a 3D ultrasound volume of the liver. The ultrasound has been recorded using a magnetically tracked 2D probe, swept in a transversal orientation over the patient's liver during breathhold. In order to validate registration accuracy, a physician manually identified anatomical landmarks and clinical targets in both modalities, resulting in 5 – 16 point correspondences for each patient.

Our registration converges in 10 – 30 iterations, depending on the complexity of the recovered displacement. A typical computation time per iteration is 240ms, which consists to roughly 18% of the ultrasound simulation operator Θ , 49% compounding into a Cartesian grid, and 33% computing D and the displacement update, the overall 3D voxel size of the system is 128^3. The entire execution time is 2 – 10s, depending on the actual abortion criteria, and the voxel size / slice count of the data sets. After the first few iterations, the size of the displacement updates becomes small enough that a re-simulation of ultrasound from CT is not necessary, as it does not significantly influence the orientation-dependent artifacts. Each iteration then encounters a 3-fold speedup, resulting in a possible overall computation time of less than one second. A rigid motion computed from the point correspondences comprises the *Ground Truth (GT)*. Initialized with this, a rigid image-based registration [222] was launched, increasing the RMS error, but better aligning large-scale peripheral structures. Our deformable algorithm was executed from this configuration, always with the same parameters (ϕ, update magnitude, convergence criteria, simulation parameters etc.) to demonstrate possible automation.

GT	Rigid	Def.	Improved	Points	Setup
3.51	10.3	9.6	yes	12	LPO
4.29	10.9	11.2	no	6	LPO
8.02	10.3	8.3	yes	9	decubitus
9.25	11.1	11.2	no	11	supine
4.03	5.43	5.39	yes	11	supine
3.31	4.32	3.78	yes	11	supine
3.68	5.10	5.51	no	7	supine
3.23	8.97	7.99	yes	7	supine
8.34	12.3	11.2	yes	16	LPO
5.25	6.26	5.54	yes	8	supine
6.15	9.89	9.15	yes	8	supine
6.42	10.3	9.2	yes	6	LPO

Table 7.4: Root-Mean-Square (RMS) errors of point correspondences at the ground gruth (GT) alignment, as well as after rigid and deformable registration, for 12 patient data sets. Also listed are the number of point correspondences, and the patient setup during the ultrasound exam (LPO = left portal oblique).

The average RMS error at the ground truth is 5.5mm, after rigid registration 8.8mm, after deformable registration 8.2mm. See table 7.4 for a detailed breakdown with respect to the individual data sets. Our algorithm improved the error from rigid registration in 9/12 cases, on average 60% of the point correspondences were improved. Visual alignment improved in all but one case (LPO setup with strongly misaligned hepatic vasculature after rigid registration).

7.5 Summary

In this chapter we have presented two methods we have investigated for multi-modal registration of 3D US and CT scans.

The first investigated approach focused on efficient, GPU-accelerated techniques for simultaneous registration of multiple 3D US volumes for generating large field of view mosaics. In this chapter we have described methods addressing and improving two limitations of the current state of the art approach for simultaneous registration of multiple 3D US volumes.

Simultaneous registration is currently mainly performance limited. In contrast to conventional pair-wise registration the evaluation of multi-variate similarity measure is computationally much more complex and costly. We have addressed this limitation in this work with a framework for GPU-accelerated multi-variate similarity measure evaluation. For optimal efficiency similarity measure GPU code is generated on the fly for the specific registration problem and GPGPU programming techniques are used to accelerate the evaluation and volume sampling. The performance of the methods was evaluated and compared to a CPU implementation demonstrating highly impressive speedups.

In this chapter we have presented a novel, multi-modal, and multi-variate approach to 3D US mosaicking. We identified the absence or with only poor quality imaged shared anatomical features for two or more overlapping 3D US volumes due to different scan directions and in general varying US image quality as one key problem inhibiting successful mono-modal, multi-variate registration. We have extended the mono-modal, multi-variate algorithm to a multi-modal, multi-variate one by introducing additional anatomy information from CT scans of the same patient. CT scans are in general available for most medical work-flows, especially as pre-interventional planning data, and thus do not require a change of existing work-flows. During registration the CT scan serves as a global anatomical reference frame guiding the registration of the multiple 3D US scans. The method was successfully evaluated on multiple real Freehand 3D US of liver anatomy and abdominal CT scans of patients with hepatic cancer.

The second presented approach investigates multi-modal, dense deformable registration of 3D US and CT scans using a variational approach. In this chapter we have described a novel method for automatically computing a dense displacement field mapping pre-interventional CT data to interventional acquired 3D Freehand US data of the same patient. The method is based and becomes feasible thanks to the combination of multiple powerful techniques: (1) Patient-specific simulation of US imaging effects using the framework presented chapter 6 of this thesis. (2) A two channel, LNCC based similarity measures. (3) Fast 3D US volume reconstruction and compounding techniques [90]. (4) Implementation and acceleration of all algorithm parts (US simulation, US compounding, US to CT registration and optimization) on the GPU for making the whole approach feasible in terms of computation time.

We presented first results for a study on 12 patient datasets for deformable registration of pairs of 3D Freehand US and CT scans. The algorithm offers both qualitatively (visual inspection) and quantitatively convincing results. Using the method differences due to patient setup, breathing, and organ motion could be effectively compensated. For 75% of the test datasets the algorithm improved the registration of physician selected anatomical and control landmarks by applying a deformable mapping compared to a rigid one.

7.5.1 Future Work

In this section we briefly want to discuss possible directions for future research related to the methods presented in this chapter.

One aspect, common to both methods, is the transition from OpenGL and GLSL to OpenCL to eliminate the implementation limits encountered within this work. The current absence of double precision floating point arithmetic in GLSL inhibits a successful implementation of the LC^2 similarity measure. Additional API features, e.g. shared memory and inter-thread communication could definitely ease the implementation efforts and promise to increase performance compared to the current OpenGL/GLSL implementation. Furthermore, an OpenCL implementation offers an unified code base for development and testing of the algorithms on GPUs, CPUs, current and future High Performance Computing (HPC) architectures.

Simultaneous Registration There are multiple directions for future work investigating simultaneous registration. Our general vision is to extend the presented framework to a general multi-modality, multi-variate image registration framework. The objective is to perform multi-modal registration not only for data from a pair of modalities but to incorporate all information into the registration framework.

One important aspect for future work is quantitative evaluation of the methods on phantom and real clinical datasets. Physician defined landmarks in CT and US data have to be used to assess the quality and correctness of the method.

Another aspect is to furthermore improve the performance of the method, especially for more complex and locally evaluated similarity measures. One approach could be to analyze the individual volumes and use a spatial subdivision scheme of the reference frame, e.g. octtree, bricks and blocks, or Binary Space Partitioning trees, to constrain the similarity measure evaluation to overlap volumes. With knowledge about the overlap volumes additionally special programs for the number of intersection volumes within each overlap volume could be used.

A second idea is to re-arrange the similarity measure evaluation and split the similarity measure program into multiple less complex programs on the GPU. Instead of on-the-fly interpolating samples from the individual volumes as done currently, first all volumes are warped to the reference frame. The similarity measure is then evaluated on the warped volumes, which would help to improve data locality and remove the need for interpolated access. This approach might require a lot more memory, especially if the individual volumes are positioned without much mutual overlap in the reference frame. However, if this method is combined with the spatial subdivision one it becomes feasible and attractive both in terms of memory usage and the prospective performance improvements.

Deformable Registration Further work of deformable US-CT registration could focus on: (1) Comparison of global versus only local displacement field estimation. The presented algorithm could be restricted to only locally improve prior global rigid or affine registration around selected anatomy of interest. Thereby, the problem of multiple local optima, caused by the fact that global transformation models can never truly match the data, could be avoided. This extensions could be integrated with relative ease by additional mask volumes into the algorithm. (2) Investigate a refined US simulation model for deformable registration. Currently, the method occasionally computes wrong displacements, mainly in regions where the simulated US does not resemble the real US images. (3) Investigate parametrized methods for deformable image registration. In [194] we have already started to investigate the impact of GPU-acceleration, using Nvidia's CUDA, for parametrized deformable registration using efficient discrete optimization, which does not require the computation of gradients. Future work could continue this work and extend it to deformable CT-US registration.

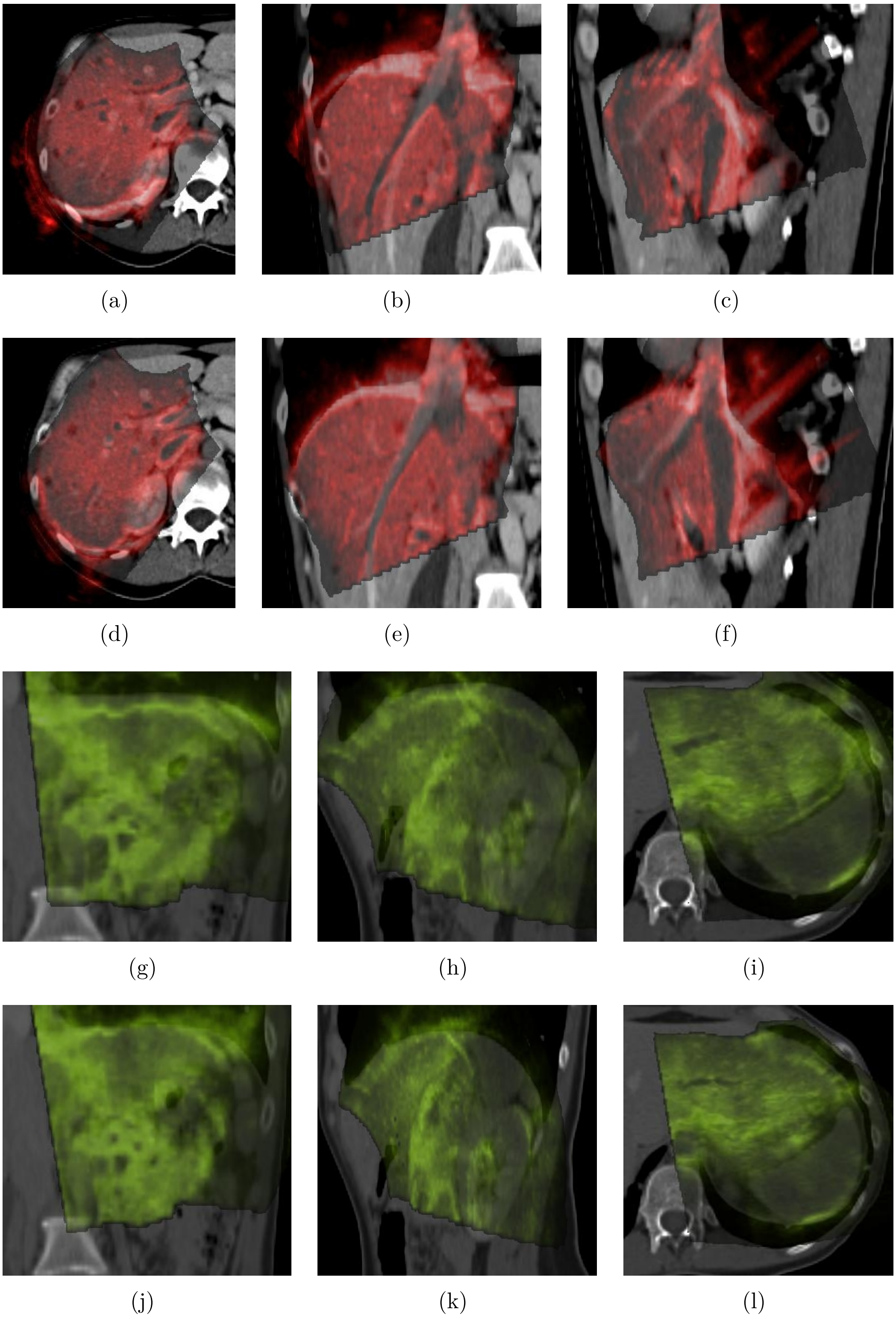


Figure 7.5: Comparison of US-CT alignment before (a-c, g-i) and after registration (d-f, j-l) for the two US volumes with small overlap depicted in figure 7.2

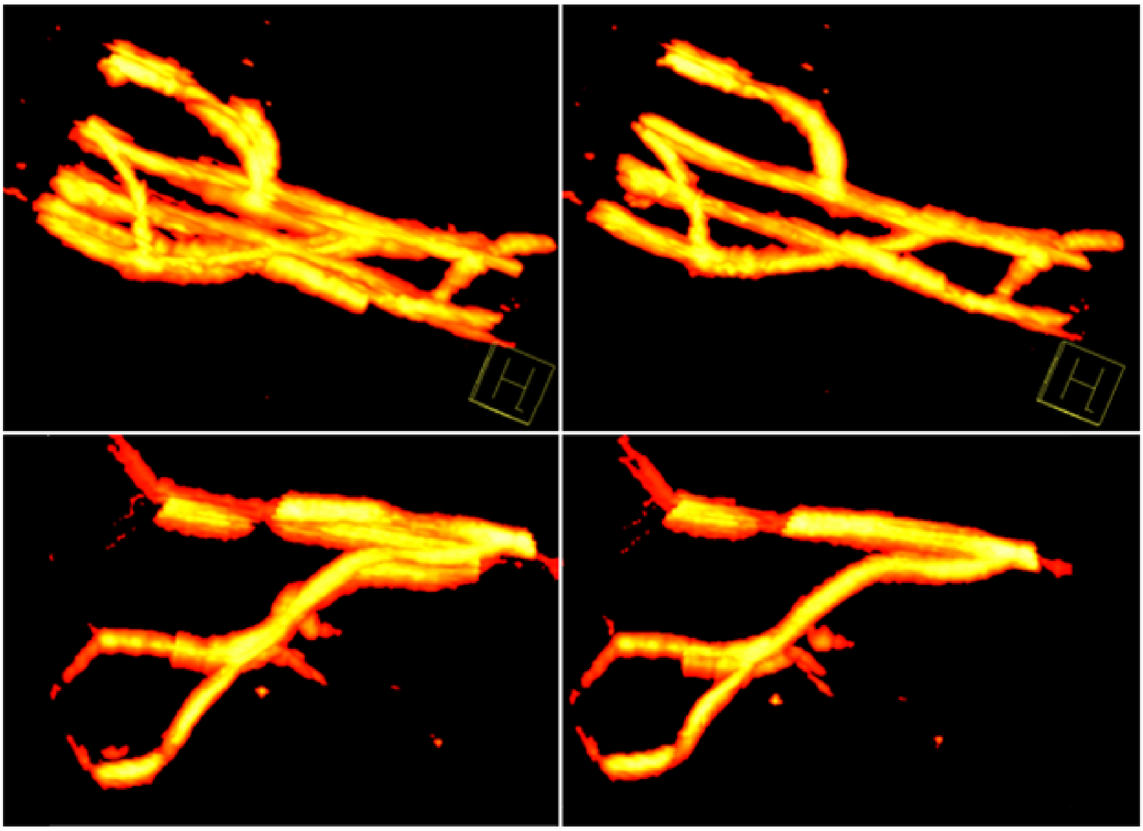
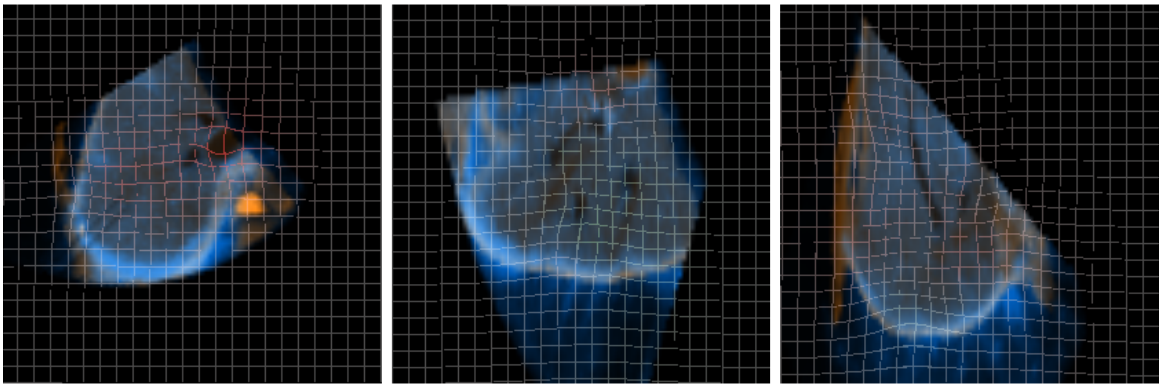


Figure 7.6: Result of simultaneous registration of multiple Power Doppler volumes for two test data set configurations. Left column: Alignment after application of calibration transformations to individual volumes. Right column: Alignment after simultaneous registration with distance-based similarity metric. The improved alignment is clearly visible for both cases.



(a) axial (b) sagittal (c) coronal

Figure 7.7: Color overlay of registration result (US simulated from CT $S_1 + S_2$ in brown, reference US blue) and resulting deformation grid for datasets from patient 1. The US data was acquired with patient in supine position during the exam.

CHAPTER 8

Conclusion

This chapter summarizes the contributions and achievements made in the dissertation. It is concluded by a discussion of its proposition within the medical image computing and computer assisted invention community, remaining challenges and ideas, starting points for future work.

8.1 Summary

This work focuses on improving computer assisted interventions by integrating GPU-accelerated algorithms for intra-operative image processing and visualization. As this work is situated at the intersection of different domains: medical image computing and high performance computing and visualization on GPUs this work first presents general overviews of the respective fields. Chapter 2 provided a comprehensive overview on medical imaging modalities, medical image computing and interventional imaging and navigation. In Chapter 3, modern GPU architectures, differences to CPUs, and a brief introduction to programming on GPUs are presented. Chapter 4 discussed GPU-based direct volume visualization concepts, which are used throughout this thesis.

The integration and application of GPU-accelerated algorithms in computer assisted interventions is discussed and its feasibility is evaluated in detail for three potential clinical applications in chapters 5, 6, and 7 summarized in the following paragraphs.

Advanced Medical AR Visualization In this exemplary application domain we present a system for real-time, in-situ, visualization of volumetric data on a stereo video see-through. We address AR specific environment requirements by advanced and optimized rendering techniques for (a) merging real and virtual images, (b) enhancing depth perception, (c) adding additional perspective by a virtual mirror and (d) handling occlusion of virtual objects with surgical instruments and the physician's hand. By using direct volume rendering for the visualization of medical data while retaining real-time performance, the whole volume information is available interventionally compared to indirect volume visualization. Thereby, the system also allows for interactive update of clinical data during an intervention, without an intermediate segmentation and mesh generation step. Furthermore, real-time 3D image data, e.g. from 3D US system, can be directly integrated within the rendering pipeline. The system's performance and its potential benefits in surgical navigation have been evaluated in experiments by several surgeons, with overall satisfying results. This will have in particular have an impact on the usability of surgical navigation solutions.

Ultrasound Simulation from CT We present a novel framework for GPU accelerated simulation US images from CT, using ray-based simulation models of varying complexity. Our simulation framework covers simulation of 2D US, freehand 3D US and 3D US acquisition from 3D CT data, as well as simulation of 2D US and 3D US imaging over time from 4D CT data. In this work we have investigated different, US imaging physics based, simulation models targeted for visually convincing simulated US image generation, and simulation for multi-modal registration between US and CT data.

Fast US simulation from patient CT data is of interest for (a) medical training and education and (b) multi-modal registration of US and CT data. This work demonstrates the feasibility of the implementation on the GPU and the largely increased performance compared to traditional CPU-based implementations. Furthermore, the presented framework has been integrated successfully into an AR based US simulation system, as well as in frameworks for multi-modal rigid and deformable registration of US and CT data.

GPU-accelerated Registration In this exemplary application we present novel solutions for GPU-based simultaneous of multiple US volumes, multi-modal registration of multiple US volumes with one CT volume, and dense deformable registration of 3D Freehand US to CT datasets. This work leverages on results from modeling US simulation from CT and introduces new concepts for fast and automatic multi-modal registration.

In [112] we investigated simultaneous registration of multiple 3D US datasets for generating large field of view 3D mosaics. The work introduces a new approach to improve simultaneous US-US registration by integrating available CT data into the registration algorithm. Thereby, the new and improved method can be applied to successfully register challenging US volume configurations, e.g. small volume overlaps, complications by noise or artifacts within overlap region, of multiple US datasets.

Furthermore, we investigated fully automatic deformable registration of CT to 3D US data. We have presented first results of successful registration of multiple real patient datasets in [225].

Both simultaneous and rigid registration methods are only made feasible by carefully modeling all problem aspects, starting with US simulation from CT, similarity measure design and selected optimization approaches. High-performance, GPU-based implementations of the presented novel solutions is the key factor to make the methods feasible in terms of computation time and therefore usable for image guided interventions.

All methods have been tested with multiple phantom and real clinical patient dataset with overall satisfying results.

8.2 Discussion and Future Work

The current trend in minimally invasive therapy in medicine will be expanding in the coming years. Real-time imaging modalities, such as 4D US and X-ray imaging, and real-time medical image computing solutions will play a central role in facilitating these new procedures, and supporting surgical navigation and decision making.

One such example of future minimally interventions, is cardiology and cardio-thoracic surgery where we expect real-time imaging based navigation and guidance to become of high importance in the next years.

Recently developed minimally invasive surgical instruments and novel techniques for implant deployment allow for endovascular treatment in a growing number of cardiac procedures, e.g. aortic valve implantation. The next step will be minimally invasive performed surgical procedures on the beating heart, eliminating the need for heart-lung machine, and reducing the trauma for the patient. However, this requires real-time solutions for image based tracking of surgical instruments and cardiac anatomy, as well as advanced real-time 3D and 4D visualization. Real-time 4D US imaging, either trans-thoracic or trans-esophageal, made feasible by the latest technological improvements in US transducer design, has a great potential to facilitate these next generation minimally invasive cardiac procedures. In contrast to nowadays frequently used real-time X-ray imaging of cardiac anatomy, US imaging allows for continuous, non-invasive, real-time acquisition of 3D functional and anatomical imaging. Recent works have investigated the feasibility of off-pump US image guided surgery [200] as well as real-time instrument tracking [146] in 3D US data. They proposed navigation and visualization solutions for minimally invasive cardiac surgery [122, 143, 145]. However, much more research, engineering and clinical investigation is required to make the transition from first prototypes into the operating room of the future and establish the supporting technology for everyday clinical routines.

Minimally invasive cardiac procedures are just one exemplary set of applications, for which real-time US imaging based navigation and visualization solutions will play an important role in the future. In the following paragraphs we will describe how US imaging guided procedures in general could potentially benefit from the methods presented in this thesis.

- **Motion Compensation:** One further point for future research starting from this work, might be investigation of motion compensation methods from real-time US imaging, using the presented frameworks for US simulation from CT and fast rigid and deformable US-CT registration. Using application targeted versions of the presented methods, one can envision to register a set of 2D US images, or a small 3D US volume to a pre-operative imaging data and motion model on the fly, while moving the US probe or manipulating surgical instruments within the US field of view. Especially for narrow or small field of view interventional US probes, real-time registration to previous US frames/volumes or CT data will be in particular of great potential [223].
- **Tracking:** Real-time, image-based tracking of surgical instruments and patient

anatomy within the US field of view is of high importance for surgical navigation. Due to artifacts in the US image data, instruments can be partially invisible, deformed or displayed at the wrong position with respect to the anatomy.

In [146], US image-based localization of surgical instrument is proposed, requiring a rigid shaft shaped instrument with attached markers. However, for flexible instruments, e.g. attached to the tip of endovascular catheters, the approach is not feasible.

High-performance, real-time implementations of rigid and deformable registration, can potentially enable image-based tracking of surgical instruments frame by frame after an initialization. The same applies to image-based tracking of anatomy for motion compensation during navigation and targeting applications. Looking even further into the future, real-time image-based tracking of instruments and anatomy could facilitate automatic positioning of interventional imaging probes and instruments by using a robot arm and visual servoing approaches.

- **Registration:** Rapid registration of pre-interventionally acquired image data to interventional image data is a another important component for image guided procedures. Using the presented methods for rapid registration of multiple US volume to each other, or to one pre-operatively acquired CT dataset allows for creation of large field of view US mosaics aiding navigation and localization of the anatomy of interest in general. Furthermore, interventional deformable US-US and CT-US allow for compensation of deformations caused by differences in patient positioning, breathing and cardiac motion.

  By reducing the time needed for registering multiple datasets, procedures can be improved by registering interventional images at different time points during the procedure to control the current state by comparison with data acquired previously during the procedures as well as pre-interventional planning data without introducing major waiting times for the computation.

- **Advanced Visualization and Medical AR:** For best presentation of complex and heterogeneous multi-modal image data during interventions advanced visualization techniques are required. For future interventional procedures it will become more and more important to intelligently combine pre-interventional and interventional image data by fused visualization to support the physician in navigation and decision making.

  Future work not only involves the algorithms for generating the images, but also the means and interfaces of presentation. In this work we have focused on HMD-based AR visualization, which has a large potential for many minimally invasive procedures. Besides the discussed navigated orthopedic surgery use case, the presented system could also be directly used for minimally invasive cardiac surgery, providing fused in-situ visualization of pre-interventional image data and interventional real-time 4D US data. One interesting aspect could the fused visualization of CT, real-time US and cardiac anatomy models estimated from patient data [80, 121].

One technical aspect of the presented work that could potentially addressed by future work is the occlusion handling of virtual and real objects in the AR scene. Using either depth information estimated from the used stereo camera setup or from time of flight cameras the occlusion problem could be solved in general. Knowledge of all real and virtual surfaces in the AR scene combined with hand tracking could facilitate interesting novel interaction concepts, e.g. picking and manipulating virtual objects using hand or surface gestures, multi-touch interaction on the patient skin surface for placing ports or virtual incisions.

However, before actual use in the operating room of the future the HMD hardware will have to undergo several refinements to further improve it, especially from an ergonomics point of view. Future HMD hardware should be less bulky, and much lighter to wear to increase the user's comfort. Furthermore, the currently narrow HMD field of view and display resolution need to be improved to provide a more natural visual user experience. Further practical improvements could be wireless transfer of video between workstation PC and HMD, or integration of the HMD in the operating room similar to surgical microscopes.

Additionally, besides AR-based visualization further presentation methods using new visualization and interaction devices, e.g. projectors, 3D screens,multi-touch surfaces, or the recently re-introduced multi-touch capable tablet PCs could be investigated to provide the best possible visualization and workflow for each specific minimally-invasive procedure.

Visualization of medical data itself provides various interesting points for future research.

One such direction could be advanced visualization of 3D/4D US data. Currently the clinical use of 3D/4D US imaging is mostly limited to applications in cardiology, cardio-thoracic surgery, gynecology and obstetrics. For these applications real-time 3D/4D US imaging provides clinically useful anatomic and functional images as the anatomy of interest can be directly visualized without being disturbed by US imaging specific artifacts. The major anatomical structures are either enclosed, baby floating in amniotic fluid, or filled, blood in cardiac chambers, by body fluids which reduces speckle and attenuation artifacts and increases contrast for visualization.

However, for many other clinical applications the use of 3D or 4D US imaging is avoided and traditional 2D US imaging is used instead. The main reason is that currently available 3D/4D US visualization techniques fail to provide the user with an automatic visualization of the anatomic region of interest for general 3D/4D US imaging. In contrast to e.g. CT data, 3D US visualization is many times more challenging as there exists no standardized intensity scale as for CT image data for voxel classification. In US imaging the voxel intensity values are largely view-dependent and furthermore subject to various view and tissue specific artifacts.

Future research could address this problem by developing new visualization techniques that can deal with challenging US data. One idea could be to make use of the view-independent speckle pattern for larger 3D regions of voxels and thereby improve classification stability for direct volume visualization.

A second approach to investigate is that several aspects of US imaging conceived as artifacts at first do actually carry valuable clinical information. E.g. occlusion artifacts at inflamed/healthy tissue interfaces or darkened regions in otherwise bright anatomy, or negative shadows can be a hint to lesions inside an organ. Integrating this information directly into the rendering algorithm could facilitate automatic focus and context visualization based on fuzzy classification. The user of the envisioned system would not directly manipulate a classic transfer function, but would rather pick one or multiple labels from a pre-defined set of classification labels, e.g. highlight all lesions, or display bone/tissue interfaces.

A third idea could be to make use of multi-modal information available from co-registered CT or MRI data. This approach could work for instance by employing multi-dimensional classification tables, being indexed with US and CT/MRI intensity or gradient values. A second approach for this idea would be to employ automatic learning techniques to learn classification schemes off-line based on multiple 3D US data and CT/MRI data sets and.

Further exploration of fused visualization of multiple datasets from the same or different modalities is just one other example. The challenge is to develop visualization algorithms able to deal with heterogeneous data of multiple sources, e.g. varying data sizes, sampling resolutions and dimensionality, automatically present the user the maximum available combined information, and respond interactively.

Besides the here mentioned examples there are, of course, many more clinical application and medical image data specific directions for future research on medical visualization.

The many times faster growing size of medical image data than available processing power and fast local memory, is one of the big open challenges for future medical visualization and image processing algorithms. The capability to handle, process and visualize, high-resolution datasets exceeding the Tera byte boundary many times will become more and more important with improvement of existing and newly developed imaging modalities. Today, already a single digital histo-pathology slice consumes several Gigabytes. A volumetric dataset can consume one or even multiple Tera bytes. The challenge for the design of future medical image processing and visualization systems to handle data of this size is to make optimal use of the available resources for data storage, caching, streaming and computing, to enable interactive level-of-detail visualization and fast image processing, e.g. registration, segmentation. Thus, some selected image processing and visualization applications might migrate processing and visualization tasks from single high performance workstation PCs to dedicated high performance computing servers, featuring a large number of dedicated CPUs and GPUs. The application itself will most likely be a thin client, forwarding user inputs and receiving a stream of images, able to be executed on a various devices ranging from smart phones to standard PCs and accessible from anywhere using low latency network connections.

Appendix

APPENDIX A

Further Applications not discussed in detail in this thesis

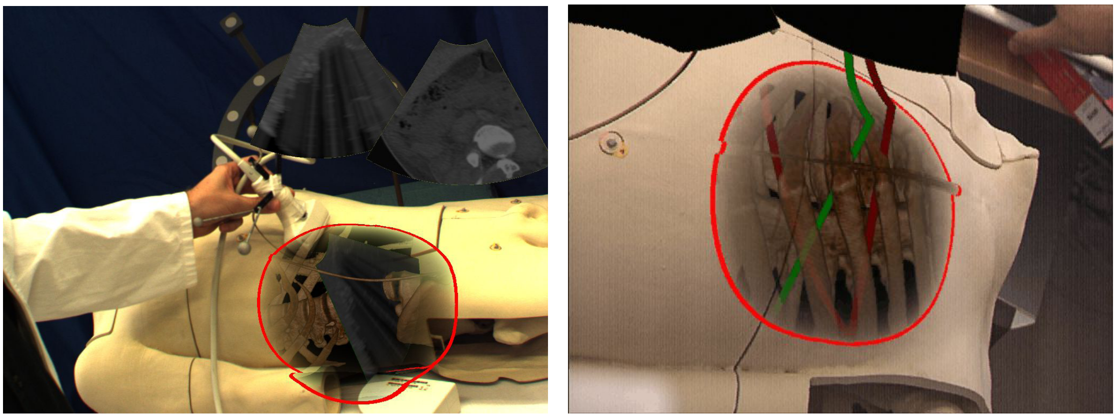
In the following sections I want to give a brief overview on further applications and projects, related to the core work presented in this thesis, I was involved and contributed to during the dissertation.

A.1 AR based US Simulation and Training System

Ultrasound (US) imaging is one of the hardest to master and interpret medical imaging modalities in todays clinical practice. The quality of the acquired images is affected by various US specific artifacts e.g. low signal to noise ratio, speckle, shadowing, and occlusion, many of them being view-dependent. Therefore, users experience in correctly manipulating the US probe to visualize the anatomical region of interest in the US image is one of the most important factors on image quality.

Learning how to manipulate the US probe requires (1) hands-on training, (2) deep understanding of human anatomy, (3) high degree of spatial sense and hand-eye coordination, and (4) feedback from an US modality expert. Many of these requirements are hard to master individually for every student in todays medical education systems. In clinical daily routine, time and money are two of the most limiting factors, e.g. every student can only train for a limited amount of time with real patients, expert US users supervising the training likewise do not have time for extensive training sessions and are also too costly for general education.

In this project an Augmented Reality (AR)-based US simulation and training system, with integrated after action review (see figure A.1), is investigated to improve training of US skills. The system is made feasible by several techniques and algorithms developed in the scope of this thesis. It uses the GPU-accelerated framework for patient-based US simulation from Computed Tomography (CT) data presented in chapter 6, as well the real-time visualization techniques for Head Mounted Display (HMD)-based medical AR presented in chapter 5.



(a) (b)

Figure A.1: AR visualization during simulated US exam. (a) The simulated US image is displayed inside the patient using focus and context rendering. Additionally for better understanding the context anatomy, the US slice and corresponding CT slice are displayed above the training phantom. (b) After action review mode visualization. Synchronized replay of expert (green) and trainee (red) probe movement during examination.

Thereby, the simulator allows in-situ visualization of a simulated US slice and human anatomy, in order to achieve a deeper understanding of the relative positions of the probe, the US slice and anatomy. Also the effect of view-dependent artifacts, most importantly occlusion in US images, which is very important to understand, can be studied in detail. Furthermore, the system allows to record the performance of the trainee. Using Dynamic Time Warping, a synchronization to the performance of an expert is performed and a synchronized replay is shown together with the trainees.

For a detailed presentation of the AR based US simulator, discussion of different possible training and feedback modes, as well as an outlook to future work using the system we refer the reader to [19]

A.2 GPU-based 2D-3D Registration

Besides the work on GPU-accelerated registration of US and CT data presented in chapter 7, I have been involved in research projects on 2D-3D registration. My focus was on GPU-accelerated 2D-3D registration, mainly methods for generating virtual 2D images and estimating similarity measures directly on the GPU. Therefore, methods presented in chapter 5.11(f) and parts of the registration framework presented in chapter 5.11(f) were adapted and extended for efficiently solving 2D-3D registration problems.

X-ray to CT Registration

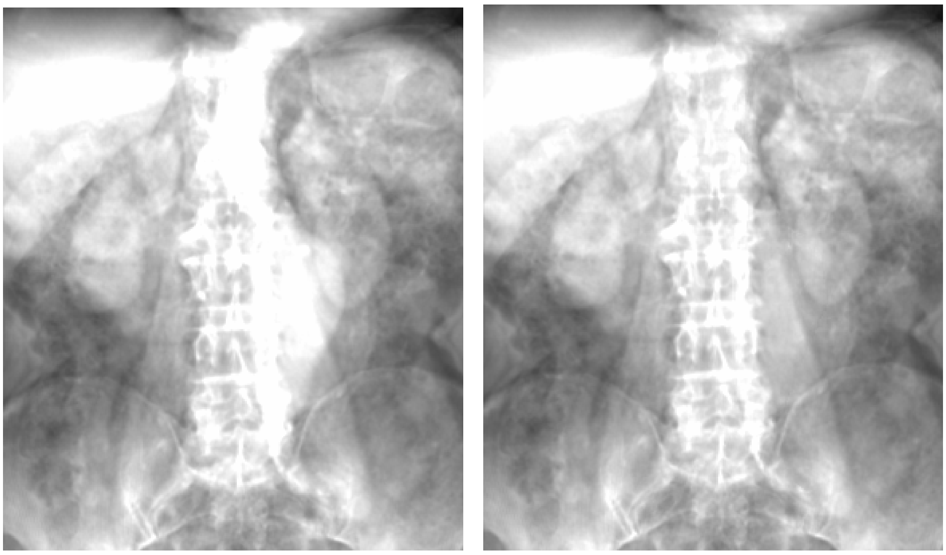
Registration of intra-operative 2D X-ray images and pre-interventional 3D CT/CTA data is a frequent task for a multitude of todays X-ray imaging guided procedures. By 2D-3D registration, pre-interventional image and planning data is registered to the X-ray imaging coordinate system in the intervention room. Fusion of pre-interventional and interventionally acquired images provides the basis for interventional X-ray image-based guidance and navigation of surgical instrument, decision making and application of therapy.

The general idea of image based 2D-3D X-ray to CT registration is to estimate the parameters of the transformation, mapping the pre-interventional data to the X-ray imaging coordinate system, by comparing real 2D X-ray images with virtual X-ray images, also known Digitally Reconstructed Radiographs (DRRs), by a suitable similarity metric and optimize the transformation parameters iteratively. DRRs, virtual 2D projection X-ray images, are generated from the 3D CT data by volume rendering and an emission only based light transfer model.

The simulated X-ray image intensity E along a ray r through the CT data Ct is given by:

$$E = -\int_{p \in r} \mu(Ct(\boldsymbol{p}))dr \tag{A.1}$$

where $\boldsymbol{p}$ is the current sample position along the ray, and μ is a conversion operator mapping Hounsfield values to X-ray attenuation values, often implemented by a 1D transfer function.



(a) DRR from CTA

(b) DRR from CTA with removal of contrasted aorta

Figure A.2: DRRs generated for 2D-3D registration of pre-interventional 3D CTA and interventional 2D X-ray images to support implantation of endovascular stent inside aneurysm in abdominal aorta. To improve registration with non-contrasted 2D X-ray images the contrasted aorta is removed in (b) when generating the DRRs compared to the standard approach (a).

GPU-accelerated volume rendering has been proposed in the past [92, 108] to accelerate 2D-3D registration by moving the most costly parts of the algorithm, namely DRR generation and similarity measure evaluation, onto the GPU. In [32] and [240], the GPU ray-caster implemented in this work is combined with 2D similarity measures, entirely evaluated on the GPU, to speed up 2D-3D registration for research works investigating Markov Random Field (MRF) optimization for 2D-3D registration, and C-arm positioning using virtual fluoroscopy.

Another 2D-3D registration problem that was addressed during this dissertation is registration of pre-interventional and interventional images with large dissimilarities, e.g. contrast dye or instrument present in only one of the two images. The focus was on improving 2D-3D registration by removing the perturbing features in either virtual or real X-ray images [31]. The clinical application, was 2D-3D registration of CTA and X-ray image data to aid navigation during endovascular stent implantation in the thoracic or abdominal aorta, for the treatment of dissections or aneurysms of the aorta. The presence of contrast dye in the aorta in the pre-operative CTA data can cause large dissimilarities between DRRs and real X-ray images acquired during the intervention. Figure A.2(a) depicts a DRR generated from a CTA dataset. The aneurysm in the abdominal aorta results in a saturation and shadowing of the vertebrae, important features for 2D-3D registration. In this work, we address this problem by removing contrasted vasculature during DRR generation. Figure A.2(b) depicts a DRR created from the same CTA data and for the same parameters with the contrasted aorta removed by our method.

During DRR rendering, we determine if the sample at the current sample position along the ray belongs to the contrasted aorta by a lookup in an additional 3D volume. If the sample position is within the aorta the sample value is set to the Hounsfield value of non-contrasted blood before the current DRR value is updated. Thereby, a DRR is generated from the CTA data as it would be from non-contrasted CT. Parts of this work have been published in SPIE 2008 [31] and have resulted in a patent (EP2088556A1, see section C) filed by Siemens on behalf of the inventors.

APPENDIX B

Further Ultrasound Simulation Results

The following pages depict in detail the resulting images generated with our US simulation and visualization framework described in chapter 6. All the images contained in this section were generated on standard hardware and simulated in real-time.

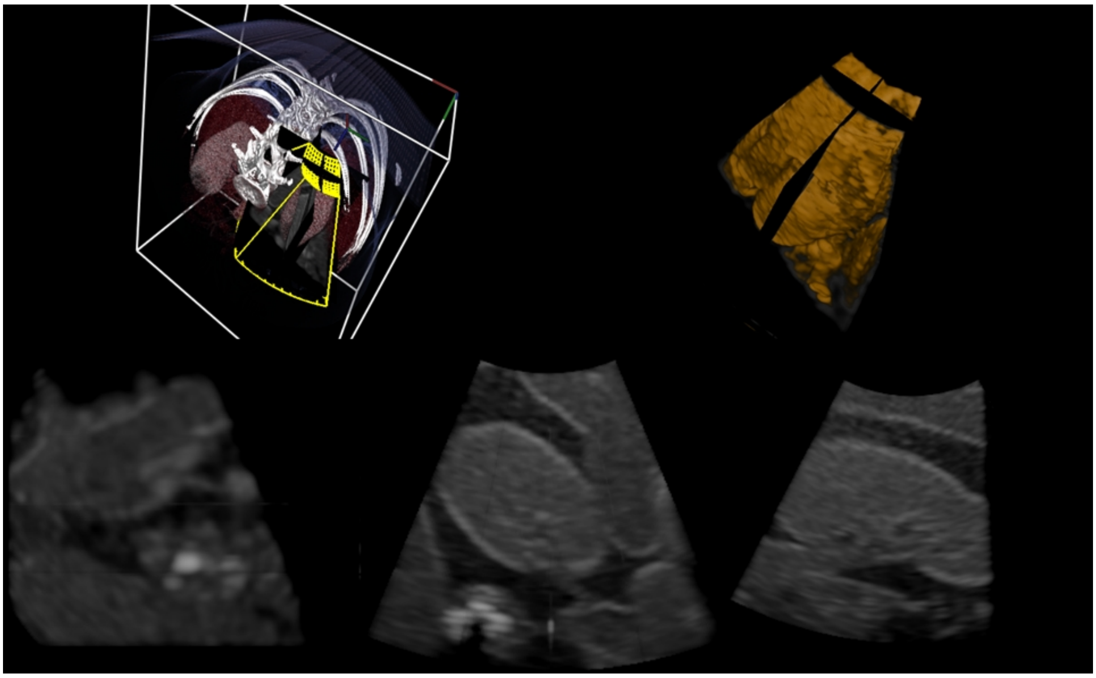
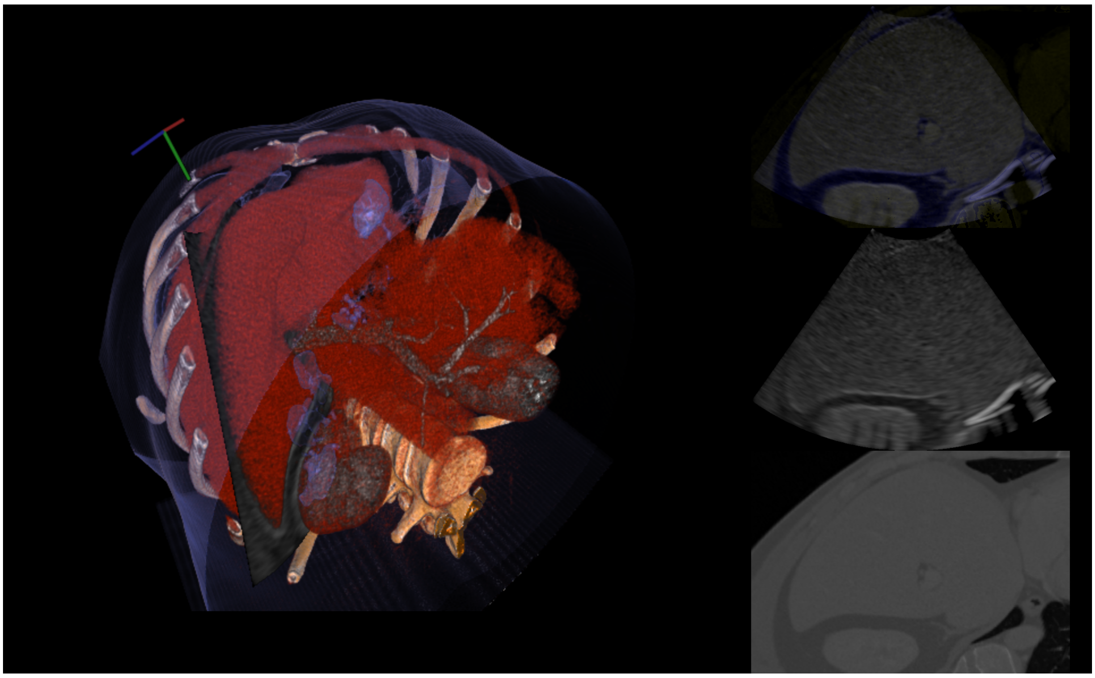
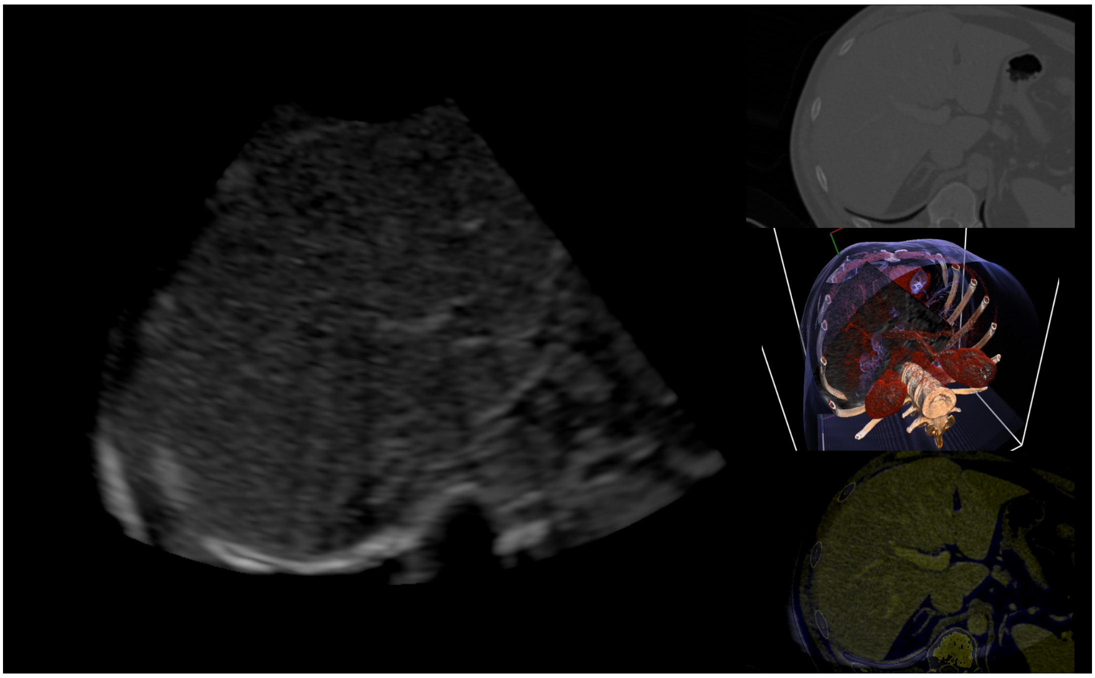


Figure B.1: Screen shot from visualization of simulated 3D ultrasound volume. Top row, from left to right: Volume rendering of CT data with wire frame rendering of ultrasound field of view and ultrasound MPRs inside the CT volume. Volume Rendering of simulated ultrasound volume with texture mapped MPR planes. Bottom row, from left to right: Axial, sagittal and coronal MPR planes extracted from simulated ultrasound volume.

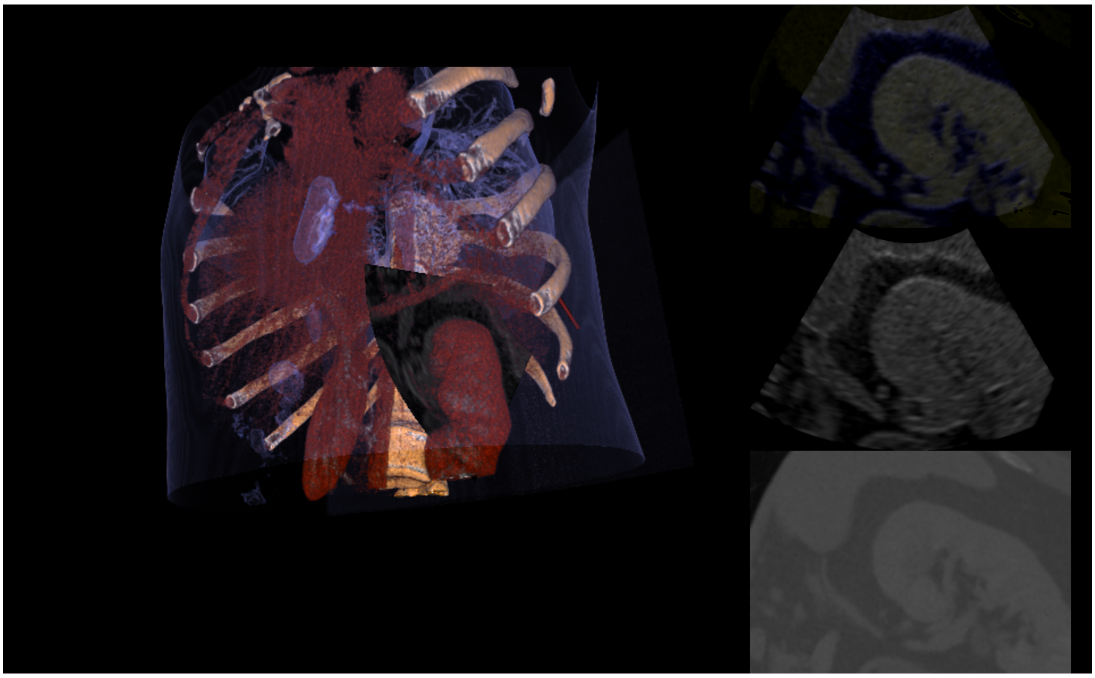


(a)

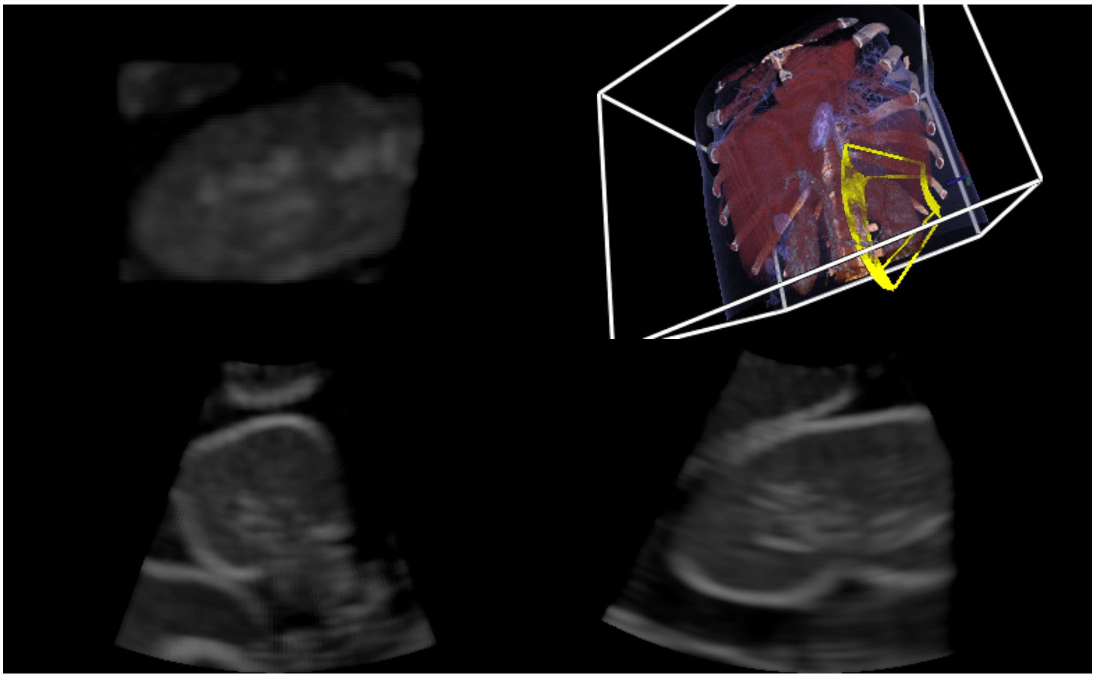


(b)

Figure B.2: (a) Simulation of a 2D B-Mode scan of the liver region. Left column: Ultrasound scan plane is embedded in volume rendering of CT data set. Right column from bottom to top: CT MPR at location and orientation of US scan plane; Simulated 2D US image; Overlay of CT MPR and simulated US Image. (b) Left column: Simulated 2D B-Mode scan of the liver region. Right column from top to bottom: CT MPR at location and orientation of US scan plane; Ultrasound scan plane is embedded in volume rendering of CT data set; Overlay of CT MPR and simulated US Image.

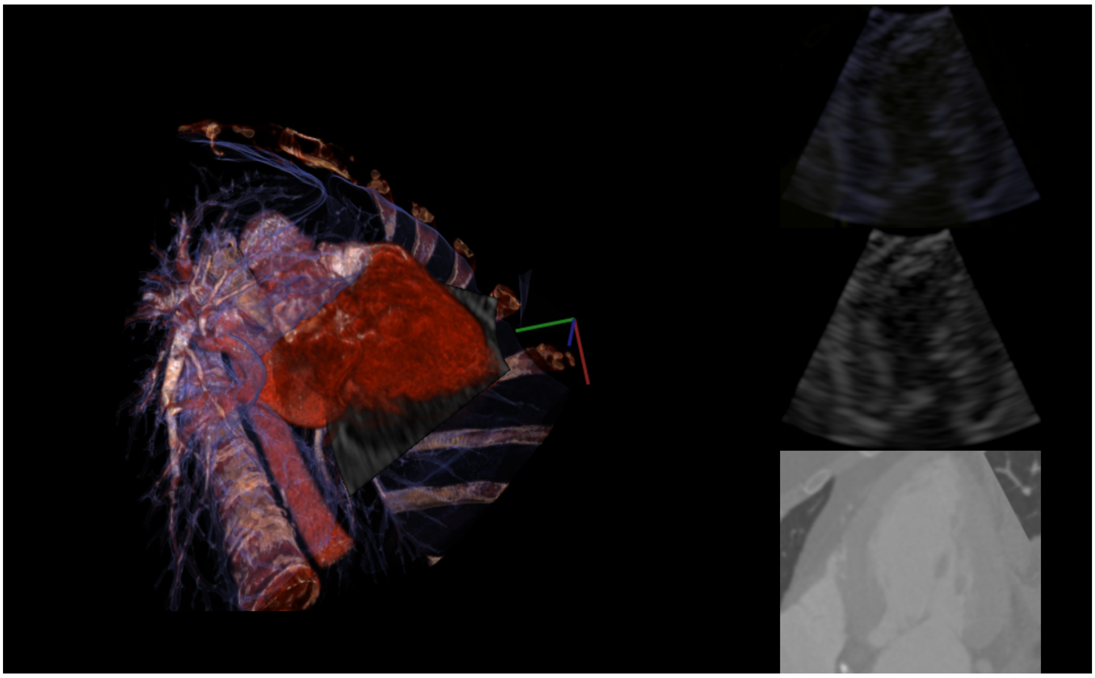


(a)

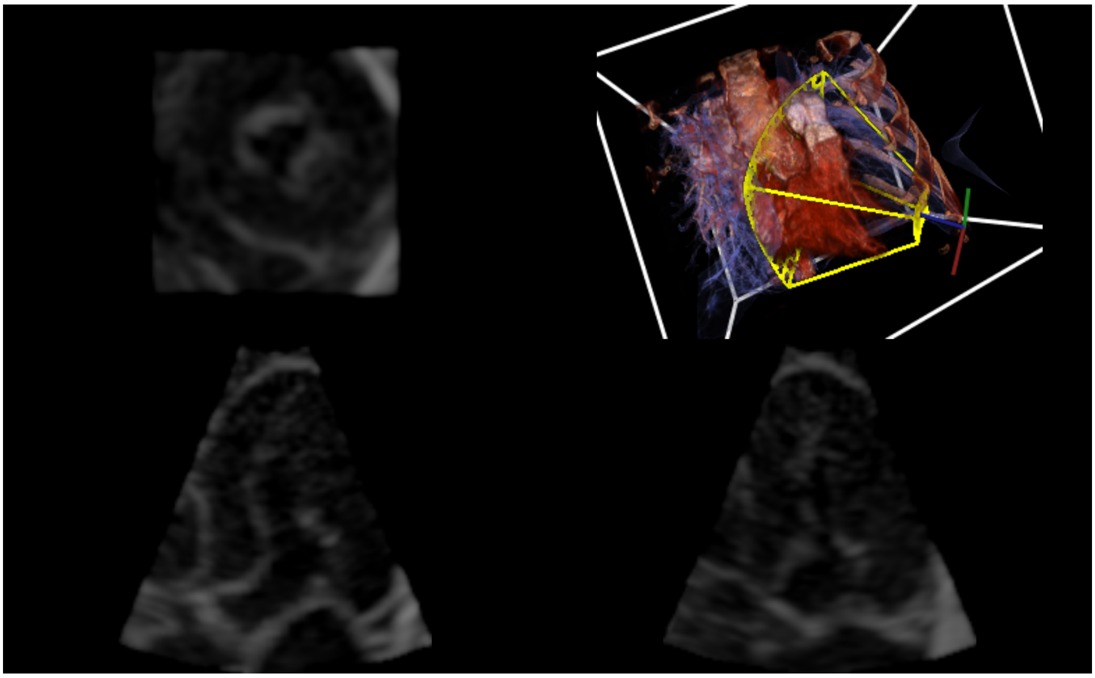


(b)

Figure B.3: (a) Left column: Simulation of a 2D B-Mode scan of the kidney region. Ultrasound scan plane is embedded in volume rendering of CT data set. Right column from bottom to top: CT MPR at location and orientation of US scan plane. Mid: Simulated 2D US image. Top: Overlay of CT MPR and simulated US Image. (b) Simulation of a 3D B-Mode scan of the kidney region. Top right quadrant: Ultrasound scan geometry is depicted as yellow wire frame rendering in 3D CT visualization. Other quadrants in image depict MPRs axially extracted from simulated volume.

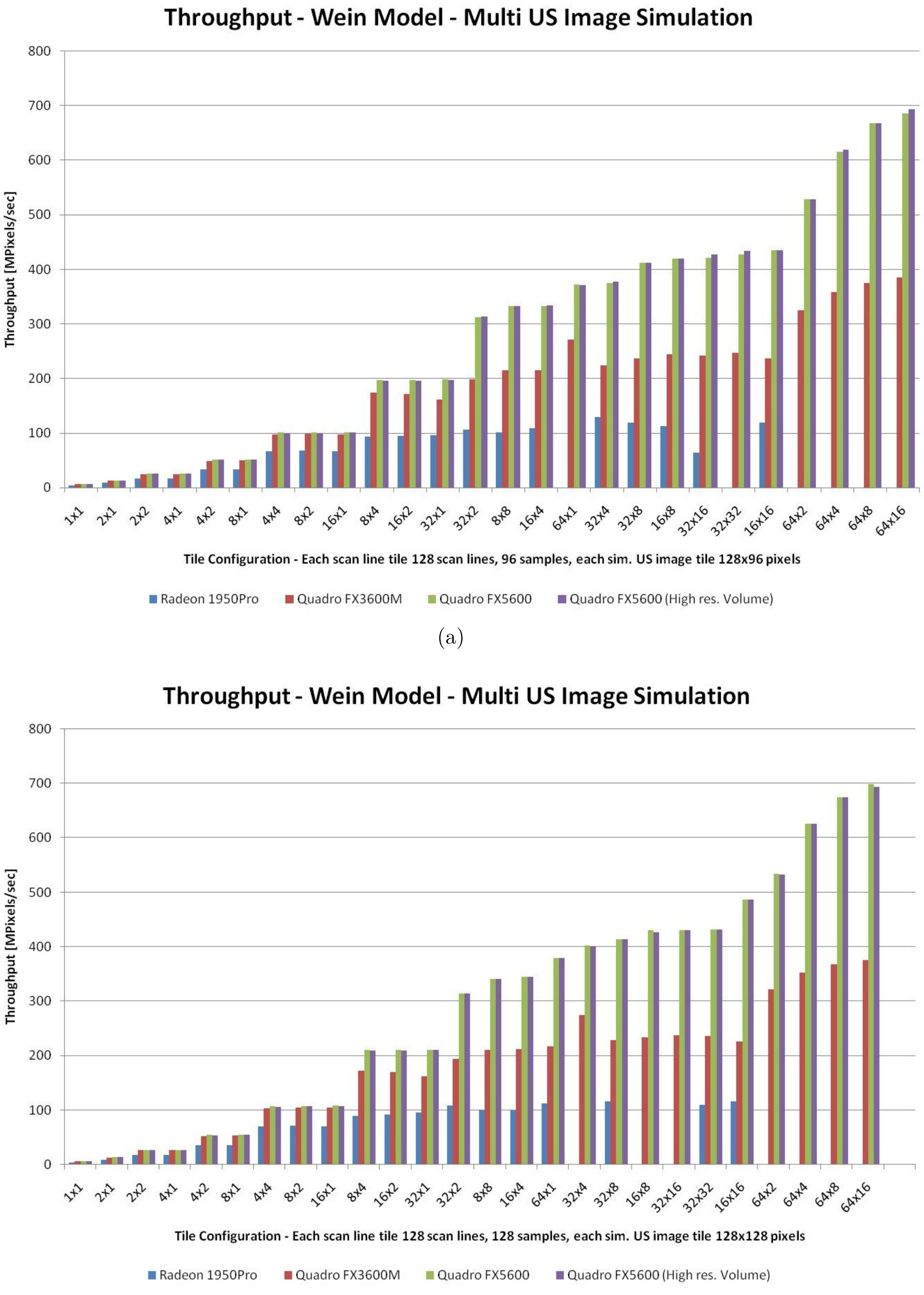


(a)



(b)

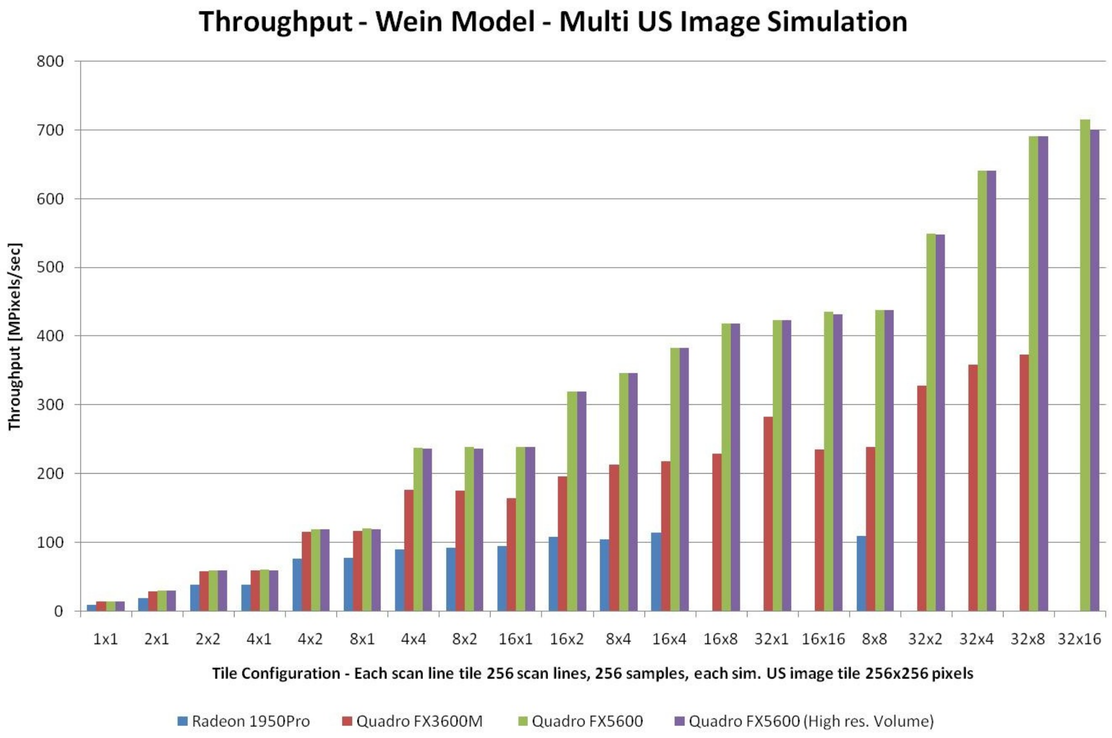
Figure B.4: (a) Simulation of a 2D B-Mode scan of the heart. Left column: Ultrasound scan plane is embedded in volume rendering of CT data set. Right column from bottom to top: CT MPR at location and orientation of US scan plane. Mid: Simulated 2D US image. Top: Overlay of CT MPR and simulated US Image. (b) Simulation of a 3D B-Mode scan of the heart. Top right corner : Ultrasound scan geometry is depicted as yellow wire frame rendering in 3D CT visualization. Other views depict axial MPRs extracted from simulated volume.



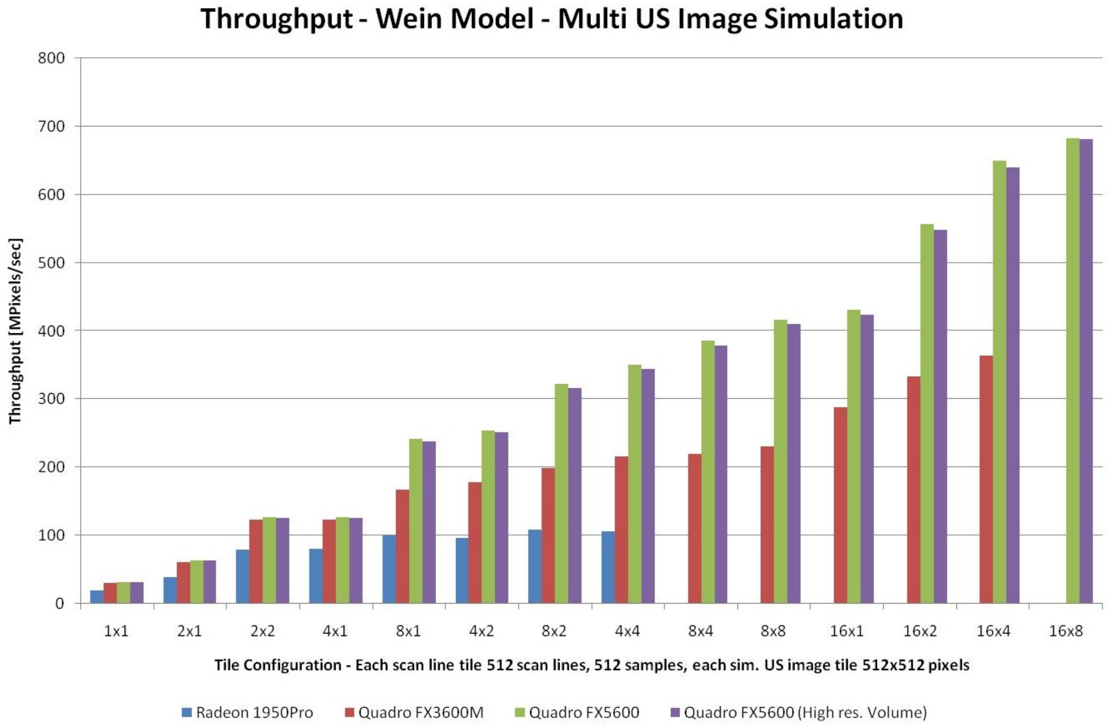
(a)

(b)

Figure B.5: Throughput [MPixels/sec] for different image tile configurations: the throughput increases with the number of tiles and is typically optimal for square configurations.



(a)



(b)

Figure B.6: Throughput [MPixels/sec] for different image tile configurations: the throughput increases with the number of tiles and is typically optimal for square configurations.

APPENDIX C

Authored and Co-Authored Publications

i. AUTOMATIC NON-LINEAR MAPPING OF PRE-PROCEDURE CT VOLUMES TO 3D ULTRASOUND, *W. Wein, O. Kutter, A. Aichert, D. Zikic, A. Kamen and N. Navab*, 2010 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, Rotterdam, the Netherlands, 14-17 April 2010

ii. LINEAR INTENSITY-BASED IMAGE REGISTRATION BY MARKOV RANDOM FIELDS AND DISCRETE OPTIMIZATION, *D. Zikic, B. Glocker, O. Kutter, M. Groher, N. Komodakis, A. Kamen, N. Paragios and N. Navab*, Medical Image Analysis, 2010

iii. FIRST DEMONSTRATION OF 3D LYMPHATIC MAPPING IN BREAST CANCER USING FREEHAND SPECT, *T. Wendler, K. Herrmann, A. Schnelzer, T. Lasser, J. Traub, O. Kutter, A. Ehlerding, K. Scheidhauer, T. Schuster, M. Kiechle, M. Schwaiger, N. Navab, S. Ziegler, and A. K. Buck*, European Journal of Nuclear Medicine and Molecular Imaging, 2010

iv. FIRST EXPERIENCES WITH FREEHAND SPECT FOR INTRAOPERATIVE 3D LYMPHATIC MAPPING , *A. Buck, A. Schnelzer, T. Wendler, K. Herrmann, T. Lasser, J. Traub, O. Kutter, T. Schuster, A. Ehlerding, K. Scheidhauer, M. Kiechle, N. Navab, S. Ziegler, M. Schwaiger*, Proceedings of DGN Nuklearmedizin 2010, Leipzig, Germany, April 2010

v. GPU-ACCELERATED RENDERING FOR MEDICAL AUGMENTED REALITY IN MINIMALLY-INVASIVE PROCEDURES, *M. Wieczorek, A. Aichert, O. Kutter, C. Bichlmeier, J. Landes, S.M.Heining, E. Euler and N.Navab*, Bildverarbeitung für die Medizin, March 2010

vi. VIDEO-BASIERTES TRACKING EINES BRONCHOSKOPS - DESIGN UND QUANTITATIVE EVALUIERUNG, *Tobias Reichl, Benedikt Schultis, Oliver Kutter, Manuela*

Menzel, Hubert Hautmann, Nassir Nav, Bildverarbeitung für die Medizin, March 2010

vii. Markov Random Field Optimization for Intensity-based 2D-3D Registration, *D. Zikic, B. Glocker, O. Kutter, M. Groher, N. Komodakis, A. Kamen, N. Paragios, N. Navab*, SPIE Medical Imaging, San Diego, California, USA, February 2010

viii. Intraoperative Positioning of Mobile C-arms using Artificial Fluoroscopy, *P. Dressel, L. Wang, O. Kutter, J. Traub, S.M. Heining, N. Navab*, SPIE Medical Imaging, San Diego, California, USA, February 2010

ix. Advanced Training Methods using an Augmented Reality Ultrasound Simulator, *T. Blum, S.M. Heining, O. Kutter, N. Navab*, The 8th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR), Orlando, Florida, USA, 19-23 October 2009

x. Contextual In-Situ Visualization for Port Placement in Keyhole Surgery: Evaluation of Three Target Applications by Two Surgeons and Eighteen Medical Trainees, *C. Bichlmeier, S. Holdstock, S.M. Heining, S. Weidert, E. Euler, O. Kutter, N. Navab*, The 8th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR), Orlando, Florida, USA, 19-23 October 2009

xi. Real-Time Learning of Accurate Patch Rectification, *S. Hinterstoisser, O. Kutter, N. Navab, P. Fua, V. Lepetit*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Miami, Florida (USA), June 2009

xii. Clinical introduction of freehand SPECT for image-guided sentinel lymph node biopsy, *T. Wendler, K. Herrmann, A. Schnelzer, T. Lasser, J. Traub, O. Kutter, T. Schuster, M. Kiechle, M. Schwaiger, N. Navab, S. Ziegler, A. Buck*, Proceedings of Annual Congress of the European Association of of Nuclear Medicine (EANM), Barcelona, Spain, October 2009

xiii. Multi-modal registration based Ultrasound Mosaicking, *O. Kutter, W. Wein, N. Navab*, Medical Image Computing and Computer-Assisted Intervention (MICCAI), London, UK, 20-24 September 2009

xiv. Development of an Augmented Reality (AR) training environment for orthopedic surgery procedures, *B. Ockert, C. Bichlmeier, S.M. Heining, O. Kutter, N. Navab, E. Euler*, Proceedings of The 9th Computer Assisted Orthopaedic Surgery (CAOS), Boston, USA, June 2009

xv. A GPU-Based Framework for Simulation of Medical Ultrasound, *O. Kutter, A. Karamalis, W. Wein, N. Navab*, SPIE Medical Imaging, Orlando, Florida, USA, February 2009

xvi. FAST HYBRID FREEHAND ULTRASOUND VOLUME RECONSTRUCTION, *A. Karamalis, W. Wein, O. Kutter, N. Navab*, SPIE Medical Imaging, Orlando, Florida, USA, February 2009

xvii. VISUALIZATION AND GPU-ACCELERATED SIMULATION OF MEDICAL ULTRASOUND FROM CT IMAGES, *O. Kutter, R. Shams, N. Navab*, Computer Methods and Programs in Biomedicine, Volume 94, Issue 3, June 2009

xviii. REAL-TIME VOLUME RENDERING FOR HIGH QUALITY VISUALIZATION IN AUGMENTED REALITY, *O. Kutter, A. Aichert, C. Bichlmeier, J. Traub, S.M. Heining, B. Ockert, E. Euler, N. Navab*, International Workshop on Augmented environments for Medical Imaging including Augmented Reality in Computer-aided Surgery (AMI-ARCS), New York, USA, September 2008

xix. THE VISIBLE KOREAN HUMAN PHANTOM: REALISTIC TEST & DEVELOPMENT ENVIRONMENTS FOR MEDICAL AUGMENTED REALITY, *C. Bichlmeier, B. Ockert, O. Kutter, M. Rustaee, S.M. Heining, N. Navab*, International Workshop on Augmented environments for Medical Imaging including Augmented Reality in Computer-aided Surgery (AMI-ARCS), New York, USA, September 2008

xx. DYNAMIC MODEL-DRIVEN QUANTIFICATION AND VISUAL EVALUATION OF THE AORTIC VALVE FROM 4D CT, *R. I. Ionasec, B. Georgescu, E. Gassner, S. Vogt, O. Kutter, M. Scheuering, N. Navab, D. Comaniciu*, Medical Image Computing and Computer-Assisted Intervention (MICCAI), New York, USA, 6-10 September 2008

xxi. SPATIO-TEMPORAL REGISTRATION IN MULTIPLANE MRI ACQUISITIONS FOR 3D COLON MOTILIY ANALYSIS, *O. Kutter, S. Kirchhoff, M. Berkovic, M. Reiser, N. Navab*, SPIE Medical Imaging, San Diego, California, USA, 16-21 February 2008

xxii. ADVANCED 2D-3D REGISTRATION FOR ENDOVASCULAR AORTIC INTERVENTIONS: ADDRESSING DISSIMILARITY IN IMAGES, *S. Demirci, O. Kutter, F. Manstad-Hulaas, R. Bauernschmitt, N. Navab*, SPIE Medical Imaging, San Diego, California, USA, 16-21 February 2008

xxiii. SIMULATION AND FULLY AUTOMATIC MULTIMODAL REGISTRATION OF MEDICAL ULTRASOUND, *W. Wein, A. Khamene, D.-A. Clevert, O. Kutter, N. Navab*, Medical Image Computing and Computer-Assisted Intervention (MICCAI), Brisbane, Australia, October 2007

xxiv. TOWARDS AN INTEGRATED PLANNING AND NAVIGATION SYSTEM FOR AORTIC STENT-GRAFT PLACEMENT, *O. Kutter, S. Kettner (=Demirci), E.U. Braun, N. Navab, R. Lange, R. Bauernschmitt*, International Journal of Computer Assisted Radiology and Surgery, 2007; Vol. 2 Suppl. 1, page 239 - 241.

xxv. A NOVEL SEGMENTATION AND NAVIGATION TOOL FOR ENDOVASCULAR STENTING OF AORTIC ANEURYSMS, *M. Feuerstein, K. Filippatos, O. Kutter, E.U. Schirmbeck, R. Bauernschmitt, N. Navab*, International Journal of Computer Assisted Radiology and Surgery, 2006; Vol. 1 Suppl. 1, page 280 - 282.

xxvi. CAMPAR: A software framework guaranteeing quality for medical augmented reality, *T. Sielhorst, M. Feuerstein, J. Traub, O. Kutter, N. Navab*, International Journal of Computer Assisted Radiology and Surgery, 2006; Vol. 1 Suppl. 1, page 29 - 30.

Patents and Invention Disclosures

i. Line-based calibration of ultrasound transducer integrated with a pose sensor, *F. Sauer, A. Khamene, O. Kutter and S. Kassner*, US20080287787

ii. Method and system for registration to align at least one preoperative image to at least one intraoperatively acquired image, *S. Demirci, O. Kutter and N. Navab*, EP2088556A1

iii. Fast Non-Linear Motion and Structure Tracking in Real-time 3D Imaging, *W. Wein, O. Kutter, A. Kamen, D. Zikic and L. Grady*, **Invention Disclosure** filed September 2009

APPENDIX D

Abstracts of Major Publications not Discussed in the Dissertation

Linear Intensity-based Image Registration by Markov Random Fields and Discrete Optimization

D. Zikic, B. Glocker, O. Kutter, M. Groher, N. Komodakis, A. Kamen, N. Paragios and N. Navab

We propose a framework for the intensity-based registration of images by linear transformations, based on a discrete Markov Random Field (MRF) formulation. Here, the challenge arises from the fact that optimizing the energy associated with this problem requires a high-order MRF model. Currently, methods for optimizing such high-order models are less general, easy to use, and efficient, than methods for the popular second-order models. Therefore, we propose an approximation to the original energy by an MRF with tractable second-order terms. The approximation at a certain point p in the parameter space is the normalized sum of evaluations of the original energy at projections of p to two-dimensional subspaces. These subspaces are formed by variation of all possible pairs of parameters. We demonstrate the quality of the proposed approximation by computing the correlation with the original energy, and show that registration can be performed by discrete optimization of the approximated energy in an iteration loop. A search space refinement strategy is employed over iterations to achieve sub-pixel accuracy, while keeping the number of labels small for efficiency. The proposed framework can encode any similarity measure, does not require a derivative of the similarity measure, and implements any specific class of linear transformations by simple changes of the MRF topology. It is robust to the settings of the internal parameters, and allows an intuitive control of the parameter ranges. We demonstrate the applicability of the framework by two applications: Mono- and multi-modal intensity-based registration, and 2D-3D registration of medical images. The evaluation is performed by random studies and real registration tasks. The tests indicate increased robustness and precision compared to corresponding standard optimization of the original energy, and demonstrate robustness to noise. A strong further potential of the proposed framework is to allow the transfer of upcoming advances in MRF optimization to linear registration problems.

Accepted for publication in Medical Image Analysis

First demonstration of 3D lymphatic mapping in breast cancer using freehand SPECT

T. Wendler, K. Herrmann, A. Schnelzer, T. Lasser, J. Traub, O. Kutter, A. Ehlerding, K. Scheidhauer, T. Schuster, M. Kiechle, M. Schwaiger, N. Navab, S.I. Ziegler, and A. K. Buck

Aims: Freehand SPECT is a 3D tomographic imaging modality based on data acquisition with a hand-held detector that is moved freely, in contrast to conventional, fixed gamma camera systems. In this pilot study, the feasibility of freehand SPECT for 3D lymphatic mapping in breast cancer was evaluated.

Methods: A total of 85 patients (pts) (age, 29-88 years) with an initial diagnosis of invasive breast cancer and no clinical evidence of nodal involvement prospectively underwent sentinel lymph node (SLN) biopsy. Preoperative lymphatic mapping (35 - 87 MBq^{99m}Tc-Nanocoll) included tomographic imaging with a SPECT/CT device (Siemens Symbia T6) serving as reference. Initially, the freehand SPECT approach was assessed in a pilot study consisting of 50 pts. The quality of each freehand SPECT acquisition was assessed and ranked as good, intermediate, or poor. In another series comprising a further 35 pts (validation study), a guidance system for the acquisition was implemented based on the results of the pilot study, ensuring acquisitions with good quality. For 3D tomographic image reconstruction, ad hoc models and iterative reconstruction algorithms were used in all 85 pts. To allow for adequate comparison, SPECT/CT data and freehand SPECT data were registered within the same coordinate system.

Results: In the pilot study, freehand SPECT enabled mapping of 24/83 SLNs in 20/44 pts (3 drop-outs, 3 pts without SLN neither in SPECT/CT nor in freehand SPECT). Using SPECT/CT as reference, the accuracy of freehand SPECT was 77.8% (7/9 nodes) in scans with good quality, while for intermediate and poor quality scans, the accuracy was reduced to 34.3% and 12.8%, respectively. In the validation study, quality feedback improved the results significantly and freehand SPECT enabled the mapping of at least one SLN in 87.5% of the pts (28/32 - 3 drop-outs). Compared to the reference method, freehand SPECT showed a sensitivity of 83.3% (35/42 nodes). False negative findings were related to insufficient scanning time, insufficient coverage of the axillary region, close proximity of the SLN to the injection site, and low tracer uptake in the SLNs

Conclusions: In this preliminary study, we could demonstrate that 3D localization of SLNs is feasible using freehand SPECT technology. Prerequisites for acquisition of a good scan quality, most likely allowing precise SLN mapping, have been defined. This approach has high potential to allow image-guided biopsy and further standardization of SLN dissection, thus bringing 3D nuclear imaging into the operating room.

Accepted for publication in European Journal of Nuclear Medicine and Molecular Imaging

Real-Time Learning of Accurate Patch Rectification

S. Hinterstoisser, O. Kutter, N. Navab, P. Fua, V. Lepetit

Recent work showed that learning-based patch rectification methods are both faster and more reliable than affine region methods. Unfortunately, their performance improvements are founded in a computationally expensive offline learning stage, which is not possible for applications such as SLAM. In this paper we propose an approach whose training stage is fast enough to be performed at run-time without the loss of accuracy or robustness. To this end, we developed a very fast method to compute the mean appearances of the feature points over sets of small variations that span the range of possible camera viewpoints. Then, by simply matching incoming feature points against these mean appearances, we get a coarse estimate of the viewpoint that is refined afterwards. Because there is no need to compute descriptors for the input image, the method is very fast at run-time. We demonstrate our approach on tracking by- detection for SLAM, real-time object detection and pose estimation applications.

Spatio-temporal registration in multiplane MRI acquisitions for 3D Colon Motiliy analysis

O. Kutter, S. Kirchhoff, M. Berkovic, M. Reiser, N. Navab

In this paper we present a novel method for analyzing and visualizing dynamic peristaltic motion of the colon in 3D from two series of differently oriented 2D MRI images. To this end, we have defined a MRI examination protocol, and introduced methods for spatio-temporal alignment of the two MRI image series into a common reference. This represents the main contribution of this paper, which enables the 3D analysis of peristaltic motion. The objective is to provide a detailed insight into this complex motion, aiding in the diagnosis and characterization of colon motion disorders. We have applied the proposed spatio-temporal method on Cine MRI data sets of healthy volunteers. The results have been inspected and validated by an expert radiologist. Segmentation and cylindrical approximation of the colon results in a 4D visualization of the peristaltic motion.

Advanced 2D-3D Registration for Endovascular Aortic Interventions: Addressing Dissimilarity in Images

S. Demirci, O. Kutter, F. Manstad-Hulaas, R. Bauernschmitt, N. Navab

In the current clinical workflow of minimally invasive aortic procedures navigation tasks are performed under 2D or 3D angiographic imaging. Many solutions for navigation enhancement suggest an integration of the preoperatively acquired computed tomography angiography (CTA) in order to provide the physician with more image information and reduce contrast injection and radiation exposure. This requires exact registration algorithms that align the CTA volume to the intraoperative 2D or 3D images. Additional to the real-time constraint, the registration accuracy should be independent of image dissimilarities due to varying presence of medical instruments and contrast agent. In this paper, we propose efficient solutions for image-based 2D-3D and 3D-3D registration that reduce the dissimilarities by image preprocessing, e.g. implicit detection and segmentation, and adaptive weights introduced into the registration procedure. Experiments and evaluations are conducted on real patient data.

Towards an Integrated Planning and Navigation System for Aortic Stent-Graft Placement

O. Kutter, S. Kettner, E.U. Braun, N. Navab, R. Lange, R. Bauernschmitt

The implantation of an endovascular stent-graft, inside the aorta is a minimally-invasive procedure for the treatment of aortic aneurysms and aortic dissections. In the current clinical workflow there is no technical guidance during the intervention except for X-Ray and ultrasound images from a TEE (transesophageal echocardiography) probe. However, exact placement of the stent-graft is crucial for the success of the procedure, thus there is a need for methods and tools aiding in the implantation of the stent-graft. Our system aims at supporting the surgeons during the intervention by visualizing registered pre-operative CTA images and intra-operative X-Ray images. Thereby a roadmap for the catheter navigation can be displayed and the physician has access to all spatial 3D information necessary for the exact graft placement. By this method, we hope to enhance the accuracy of the surgeon's actions and reduce the amount of contrast agent.

A Novel Segmentation and Navigation Tool for Endovascular Stenting of Aortic Aneurysms

M. Feuerstein, K. Filippatos, O. Kutter, E.U. Schirmbeck, R. Bauernschmitt, and N. Navab

Endovascular stenting is a minimally invasive technique to exclude an aortic aneurysm or dissection from the circulatory system. Currently, there is no technical aid to guide the surgical staff during the intervention, except the default visualization interface provided with the CT scanner and mobile C-arm. The purpose of our proposed system is two-fold: (1) In the planning phase, a modified graph cuts algorithm automatically segments the aorta and aneurysm, so the surgical staff can choose an appropriate type of stent to match the segmented location, length, and diameter of the aneurysm and aorta. (2) During implantation of the stent graft, after a landmark based registration of CT and angiography data, the current position of the stent can be visualized in the 3D CT data set at any time. This will enhance the accuracy of the actions of the surgeon, along with a minimum use of angiography, leading to less radiation exposure and less contrast agent injection.

CAMPAR: A Software Framework guaranteeing Quality for Medical Augmented Reality

T. Sielhorst, M. Feuerstein, J. Traub, O. Kutter, N.Navab

In situ visualization is the projection of imaging data into the doctor's field of view in real time for medical procedures. This kind of augmented reality (AR) needs other requirements in terms of quality of service, reliability, and accuracy than its counterparts in industry and entertainment. We believe in situ visualization during medical procedures such as minimally invasive surgery. For any system based on augmented reality visualization we need the three essential requirements: Reliability, usability, and interoperability. Since the persuasive power of this kind of visualization is very high, visualization software must guarantee certain accuracy constraints by monitoring it during run time. This paper describes a general approach for a software architecture needed for in situ visualization.

Glossary

Fluorodeoxyglucose Fluorodeoxyglucose is a transport molecule commonly used for radioactive tracers in functional imaging modalities, e.g. SPECT, PET.

Acronyms

LC^2	Linear Correlation of Linear Combination
1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
4D	four-dimensional, or three-dimensional over time
API	Application Programmer Interface
AR	Augmented Reality
CAS	Computer Aided Surgery
CC	Correlation Coefficient
CFD	Computational Fluid Dynamic
Cg	C for Graphics
CPU	Central Processing Unit
CR	Correlation Ratio
CT	Computed Tomography
Ct	C for Throughput Computing
CTA	Computed Tomography Angiography
CUDA	Compute Unified Device Architecture
DLL	dynamic-link library
DOF	Degrees of Freedom
DRR	Digitally Reconstructed Radiograph
DVR	Direct Volume Rendering

fps frames per second

GLSL OpenGL Shading Language

GPGPU General Purpose GPU

GPU Graphics Processing Unit

HLSL High Level Shading Language

HMD Head Mounted Display

HPC High Performance Computing

HU Hounsfield Unit

LED Light emitting Diode

LNCC Local Normalized Cross Correlation

MI Mutual Information

MIP Maximum Intensity Projection

MPR Multi Planar Reconstruction

MRA Magnetic Resonance Imaging Angiography

MRF Markov Random Field

MRI Magnetic Resonance Imaging

NCC Normalized Cross Correlation

NGF Normalized Gradient Fields

OpenCL Open Computing Language

OpenGL Open Graphics Library

PET Positron Emission Tomography

RAMP Reality Augmentation for Medical Procedures

ROI Region of Interest

SAD Sum of absolute differences

SPECT Single Photon Emission Computed Tomography

SSD	Sum of squared differences
TEE	Transesophegeal Echocardioagraphy
US	Ultrasound
VKH	Visible Korean Human
VKHP	Visible Korean Human Phantom
VOI	Volume of Interest
VR	Virtual Reality
X-ray	X-ray Imaging

References

[1] *UltraSim: Ultrasound training simulator*, MedSim Advanced Medical Simulations, Ltd., http://www.medsim.com/, 2008.

[2] K. Abd-Elmoniem, A. Youssef, and Y. Kadah, *Real-time speckle reduction and coherence enhancement in ultrasoundimaging via nonlinear anisotropic diffusion*, Biomedical Engineering, IEEE Transactions on, 49 (2002), pp. 997–1014.

[3] A. Achim, A. Bezerianos, and P. Tsakalides, *Novel Bayesian multiscale method for speckle removal in medicalultrasound images*, Medical Imaging, IEEE Transactions on, 20 (2001), pp. 772–783.

[4] D. Aiger and D. Cohen-Or, *Real-time ultrasound imaging simulation*, Real-Time Imag., 4 (1998), pp. 263–274.

[5] D. Aiger and D. Cohen-Or, *Mosaicing ultrasonic volumes for visual simulation*, IEEE Computer Graphics and Applications, (2000), pp. 53–61.

[6] M. Bajura, H. Fuchs, and R. Ohbuchi, *Merging virtual objects with the real world: seeing ultrasound imagery within the patient*, in Proceedings of the 19th annual conference on Computer graphics and interactive techniques, ACM Press, 1992, pp. 203–210.

[7] M. A. Bauer, *Tracking Errors in Augmented Reality*, PhD thesis, Technische Universität München, 2007.

[8] J. Beyer, M. Hadwiger, S. Wolfsberger, and K. Bühler, *High-quality multimodal volume rendering for preoperative planning of neurosurgical interventions*, IEEE Transactions on Visualization and Computer Graphics, 13 (2007), pp. 1696–1703.

[9] C. Bichlmeier, *Immersive, Interactive and Contextual In-Situ Visualization for Medical Applications*, PhD thesis, Technische Universität München, 2010.

[10] C. Bichlmeier, S. Holdstock, S. M. Heining, S. Weidert, , E. Euler, O. Kutter, and N. Navab, *Contextual In-Situ Visualization for Port Placement in Keyhole Surgery: Evaluation of Three Target Applications by Two Surgeons and Eighteen Medical Trainees*, in Proceedings of the 8th International Symposium on Mixed and Augmented Reality (ISMAR), Oct. 2009.

[11] C. Bichlmeier and N. Navab, *Virtual window for improved depth perception in medical ar*, in International Workshop for Augmented Environments for Medical Imaging and Computer-aided Surgery (AMI-ARCS), Copenhagen, Denmark, Oct. 2006, MICCAI Society.

[12] C. Bichlmeier, B. Ockert, O. Kutter, M. Rustaee, S. M. Heining, and N. Navab, *The visible korean human phantom: Realistic test & development environments for medical augmented reality*, in International Workshop for Augmented Environments for Medical Imaging and Computer-aided Surgery (AMI-ARCS), New York, USA, Sept. 2008, MICCAI Society.

[13] C. Bichlmeier, M. Rustaee, S. Heining, and N. Navab, *Virtually extended surgical drilling device: Virtual mirror for navigated spine surgery*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), N. Ayache, S. Ourselin, and A. Maeder, eds., Lecture Notes in Computer Science, Springer, 2007.

[14] C. Bichlmeier, T. Sielhorst, and N. Navab, *The tangible virtual mirror: New visualization paradigm for navigated surgery*, in International Workshop for Augmented Environments for Medical Imaging and Computer-aided Surgery (AMI-ARCS), Copenhagen, Denmark, Oct. 2006, MICCAI Society.

[15] C. Bichlmeier, F. Wimmer, S. M. Heining, and N. Navab, *Contextual Anatomic Mimesis: Hybrid In-Situ Visualization Method for Improving Multi-Sensory Depth Perception in Medical Augmented Reality*, in Proceedings of the 6th International Symposium on Mixed and Augmented Reality (ISMAR), Nov. 2007, pp. 129–138.

[16] W. Birkfellner, K. Huber, F. Watzinger, M. Figl, F. Wanschitz, R. Hanel, D. Rafolt, R. Ewers, and H. Bergmann, *Development of the varioscope ar – a see-through hmd for computer-aided surgery*, in Proc. IEEE and ACM Int'l Symp. on Augmented Reality, IEEE Computer Society Press, 2000, pp. 54–59.

[17] M. Blackwell, C. Nikou, A. M. D. Gioia, and T. Kanade, *An image overlay system for medical data visualization*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), I. W. M. Wells, A. C. F. Colchester, and S. L. Delp, eds., vol. 1496 of Lecture Notes in Computer Science, Cambridge, MA, USA, October 1998, Springer, pp. 232–240.

[18] T. Blodgett, C. Meltzer, and D. Townsend, *PET/CT: form and function*, Radiology, 242 (2007), p. 360.

[19] T. Blum, S. M. Heining, O. Kutter, and N. Navab, *Advanced training methods using an augmented reality ultrasound*, in Proceedings of the 8th International Symposium on Mixed and Augmented Reality (ISMAR), Oct. 2009.

[20] Y. Boykov, O. Veksler, and R. Zabih, *Fast approximate energy minimization via graph cuts*, IEEE transactions on Pattern Analysis and Machine Intelligence (PAMI), 23 (2001), pp. 1222–1239.

[21] S. Britz-Cunningham and S. Adelstein, *Molecular targeting with radionuclides: state of the science*, Journal of Nuclear Medicine, 44 (2003), pp. 1945–1961.

[22] M. Brown and D. Lowe, *Automatic panoramic image stitching using invariant features*, International Journal of Computer Vision, 74 (2007), pp. 59–73.

[23] I. Buck, T. Foley, D. Horn, J. Sugerman, K. Fatahalian, M. Houston, and P. Hanrahan, *Brook for GPUs: stream computing on graphics hardware*, ACM Transactions on Graphics (TOG), 23 (2004), pp. 777–786.

[24] M. Burns, M. Haidacher, W. Wein, I. Viola, and E. Groeller, *Feature emphasis and contextual cutaways for multimodal medical visualization*, in EuroVis 2007 Proceedings, May 2007.

[25] C. Chefd'hotel, G. Hermosillo, and O. Faugeras, *Flows of diffeomorphisms for multimodal image registration*, 2002, pp. 753–756.

[26] J. Cormier, K. Thomas, R. Chari, and C. Pinson, *Management of hepatocellular carcinoma*, Journal of Gastrointestinal Surgery, 10 (2006), pp. 761–780.

[27] J. Cutting, *How the eye measures reality and virtual reality*, Behavior Research Methods Instruments and Computers, 29 (1997), pp. 27–36.

[28] J. Cutting and P. Vishton, *Perceiving layout and knowing distances: The integration, relative potency, and contextual use of different information about depth*, Perception of space and motion, (1995), pp. 69–117.

[29] L. Dagum and R. Menon, *OpenMP: An industry-standard API for shared-memory programming*, IEEE Computational Science & Engineering, 5 (1998), pp. 46–55.

[30] R. Daniel, *Dynamic Resolution in GPU-Accelerated Volume Rendering to Autostereoscopic Multiview Lenticular Displays*, EURASIP Journal on Advances in Signal Processing, 2009 (2008).

[31] S. Demirci, O. Kutter, F. Manstad-Hulaas, R. Bauernschmitt, and N. Navab, *Advanced 2d-3d registration for endovascular aortic interventions: Addressing dissimilarity in images*, in SPIE Medical Imaging, San Diego, California, USA, February 2008.

[32] P. Dressel, L. Wang, O. Kutter, J. Traub, S. M. Heining, and N. Navab, *Intraoperative positioning of mobile c-arms using artificial fluoroscopy*, in Proceedings of SPIE Medical Imaging, vol. 7625, California, USA, February 2010, p. 762506.

[33] V. Dutt and J. Greenleaf, *Adaptive speckle reduction filter for log-compressed B-scan images*, Medical Imaging, IEEE Transactions on, 15 (1996), pp. 802–813.

[34] P. J. Edwards, L. G. Johnson, D. J. Hawkes, M. R. Fenlon, A. J. Strong, and M. J. Gleeson, *Clinical experience and perception in stereo augmented reality surgical navigation*, in Proceesings of Medical Imaging and Augmented Reality: Second International Workshop, MIAR 2004, Beijing, China, August 19-20, 2004., 2004, pp. 369–376.

[35] W. Eisenmenger, *The mechanisms of stone fragmentation in ESWL.*, Ultrasound Med Biol, 27 (2001), pp. 683–93.

[36] K. Engel, M. Hadwiger, J. Kniss, and C. Rezk-Salama, *Real-Time Volume Graphics*, AK Peters, Ltd., 2006.

[37] K. Engel, M. Kraus, and T. Ertl, *High-quality pre-integrated volume rendering using hardware-accelerated pixel shading*, Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware, (2001), pp. 9–16.

[38] H.-H. Enricke, *SONOSim3D: a multimedia system for sonography simulation and education with an extensible case database*, European J. of Ultrasound, 7 (1998), pp. 225–300.

[39] E. Even-Sapir, Z. Keidar, and R. Bar-Shalom, *Hybrid imaging (spect/ct and pet/ct)-improving the diagnostic accuracy of functional/metabolic and anatomic imaging*, in Seminars in Nuclear Medicine, vol. 39, Elsevier, 2009, pp. 264–275.

[40] Z. Fan, C. Vetter, C. Guetter, D. Yu, R. Westermann, A. Kaufman, and C. Xu, *Optimized GPU implementation of learning-based non-rigid multimodal registration*, in Proceedings of SPIE, vol. 6914, 2008, p. 69142Y.

[41] O. Faugeras, *Three-Dimensional Computer Vision. A Geometric Viewpoint*, Artificial Intelligence, MIT Press, 2001.

[42] L. Feldkamp, L. Davies, and J. Kress, *Practical cone-beam proof algorithm*, J. Opt. Soc. Am., 6 (1984), pp. 612–619.

[43] A. Fenster, D. Downey, and H. Cardinal, *Three-dimensional ultrasound imaging*, Phys Med Biol, 46 (2001), pp. 67–99.

[44] R. Fernando, *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics*, Pearson Higher Education, 2004.

[45] M. Feuerstein, T. Mussack, S. M. Heining, and N. Navab, *Intraoperative laparoscope augmentation for port placement and resection planning in minimally invasive liver resection*, IEEE Trans. Med. Imag., 27 (2008), pp. 355–369.

[46] J. Fischer, D. Bartz, and W. Strasser, *Occlusion handling for medical augmented reality using a volumetric phantom model*, Proceedings of the ACM symposium on Virtual reality software and technology, (2004), pp. 174–177.

[47] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, *Multiscale vessel enhancement filtering*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), I. W. M. Wells, A. C. F. Colchester, and S. L. Delp, eds., vol. 1496 of Lecture Notes in Computer Science, Cambridge, MA, USA, October 1998, Springer-Verlag, pp. 130–137.

[48] A. François, E. Kang, and M. U., *A handheld virtual mirror*, in Proceedings of SIGGRAPH, 2002, p. 140.

[49] H. Fuchs, M. A. Livingston, R. Raskar, D. Colucci, K. Keller, A. State, J. R. Crawford, P. Rademacher, S. H. Drake, and A. A. Meyer, *Augmented reality visualization for laparoscopic surgery*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), I. W. M. Wells, A. C. F. Colchester, and S. L. Delp, eds., vol. 1496 of Lecture Notes in Computer Science, Cambridge, MA, USA, October 1998, Springer, pp. 934–943.

[50] A. Fuhrmann and R. Splechtna, *Comprehensive calibration and registration procedures for augmented reality*, (2001), p. 219.

[51] S. Ganguli, J. B. Kruskal, D. D. Brennan, and R. A. Kane, *Intraoperative laparoscopic ultrasound*, Radiologic Clinics of North America, 44 (2006), pp. 925–935.

[52] A. Gee, G. Treece, R. Prager, C. Cash, and L. Berman, *Rapid registration for wide field of view freehand three-dimensional ultrasound*, IEEE transactions on medical imaging, 22 (2003), pp. 1344–1357.

[53] S. Gill, P. Mousavi, G. Fichtinger, E. Chen, J. Boisvert, D. Pichora, and P. Abolmaesumi, *Biomechanically constrained groupwise us to ct registration of the lumbar spine*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), G.-Z. Yang, D. Hawkes, D. Rueckert, A. Noble, and C. Taylor, eds., vol. 5761 of Lecture Notes in Computer Science, Springer, 2009, pp. 803–810.

[54] L. Grady, *Random walks for image segmentation*, IEEE Trans. on Pattern Analysis and Machine Intelligence, 28 (2006), pp. 1768–1783.

[55] L. Grady, T. Schiwietz, S. Aharon, and R. Westermann, *Random walks for interactive organ segmentation in two and three dimensions: Implementation and validation*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted

Intervention (MICCAI), J. S. Duncan and G. Gerig, eds., vol. 3750 of Lecture Notes in Computer Science, Palm Springs, CA, Oct. 2005, MICCAI Society, Springer, pp. 773–780.

[56] V. Grau, H. Becher, and J. Noble, *Registration of Multiview Real-Time 3-D Echocardiographic Sequences*, Medical Imaging, IEEE Transactions on, 26 (2007), pp. 1154–1165.

[57] E. Grube, J. Laborde, U. Gerckens, T. Felderhoff, B. Sauren, L. Buellesfeld, R. Mueller, M. Menichelli, T. Schmidt, B. Zickmann, et al., *Percutaneous implantation of the CoreValve self-expanding valve prosthesis in high-risk patients with aortic valve disease: the Siegburg first-in-man study*, Circulation, 114 (2006), p. 1616.

[58] E. Haber and J. Modersitzki, *Intensity Gradient Based Registration and Fusion of Multi-modal Images*, in International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), vol. 4191, Springer, September 2006, pp. 726–733.

[59] M. Hadwiger, A. Kratz, C. Sigg, and K. Bühler, *GPU-Accelerated Deep Shadow Maps for Direct Volume Rendering*, Graphics Hardware 2006: Eurographics Symposium Proceedings, Vienna, Austria, September 3-4, 2006, (2006).

[60] M. Hadwiger, C. Langer, H. Scharsach, and K. Bühler, *State of the Art Report 2004 on GPU-Based Segmentation*, tech. report, Technical Report TR-VRVIS-2004-17, VRVis Research Center, 2004. Online available at http://medvis. vrvis. at/fileadmin/publications/TR_VRVIS_2004_17. pdf.

[61] M. Hadwiger, P. Ljung, C. Salama, and T. Ropinski, *Advanced illumination techniques for GPU volume raycasting*, in ACM SIGGRAPH ASIA 2008 courses, ACM, 2008, p. 1.

[62] M. Haidacher, *Importance-Driven Rendering in Interventional Imaging*, master's thesis, TU Wien, 2007.

[63] J. Hajnal, D. Hawkes, and D. Hill, *Medical Image Registration*, CRC Press, 2001.

[64] D. Harkin, M. Dillon, P. Blair, P. Ellis, and F. Kee, *Endovascular ruptured abdominal aortic aneurysm repair (EVRAR): a systematic review*, European Journal of Vascular & Endovascular Surgery, 34 (2007), pp. 673–681.

[65] M. Harris and D. Luebke, *GPGPU: General-purpose computation on graphics hardware*, in International Conference on Computer Graphics and Interactive Techniques: ACM SIGGRAPH 2005 Courses: Los Angeles, California, Association for Computing Machinery, Inc, One Astor Plaza, 1515 Broadway, New York, NY, 10036-5701, USA,, 2005.

[66] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2nd ed., 2003.

[67] D. Hawkes, D. Barratt, T. Carter, J. McClelland, and B. Crum, *Non-Rigid Registration*, Image-Guided Interventions - Technology and Applications, Springer, 2008, ch. 7.

[68] W. Hedrick, D. Hykes, and D. Starchman, *Ultrasound Physics and Instrumentation*, Mosby - Year Book, Inc., third ed., 1995.

[69] I. M. Heer, K. Middendorf, S. Müller-Egloff, M. Dugas, and A. Strauss, *Ultrasound training: the virtual patient*, Ultrasound Obstet. Gynecol., 24 (2004), pp. 440–444.

[70] D. Henry, J. Troccaz, J. L. Bosson, and O. Pichot, *Ultrasound imaging simulation: Application to the diagnosis of deep venous thromboses of lower limbs*, in MICCAI98, I. W. M. Wells, A. C. F. Colchester, and S. L. Delp, eds., Cambridge, MA, USA, October 1998, pp. 1032–1040.

[71] G. Hermosillo, *Variational Methods for Multimodal Image Matching*, PhD thesis, Universite de Nice - Sophia Antipolis, 2002.

[72] D. L. Hill, P. G. Batchelor, M. Holden, and D. J. Hawkes, *Medical image registration*, Phys. Med. Biol., 46 (2001), pp. R1–R45.

[73] H. Ho, G. Sands, H. Schmid, K. Mithraratne, G. Mallinson, and P. Hunter, *A hybrid 1d and 3d approach to hemodynamics modelling for a patient-specific cerebral vasculature and aneurysm*, in Medical Image Computing and Computer-Assisted Intervention – MICCAI 2009, G.-Z. Yang, D. Hawkes, D. Rueckert, A. Noble, and C. Taylor, eds., vol. 5762 of Lecture Notes in Computer Science, Springer, 2009, pp. 323–330.

[74] W. A. Hoff and T. L. Vincent, *Analysis of head pose accuracy in augmented reality*, IEEE Trans. Visualization and Computer Graphics, 6 (2000), pp. 319–334.

[75] A. Hostettler, C. Forest, A. Forgione, L. Soler, and J. Marescaux, *Real-time ultrasonography simulator based on 3D CT-scan images*, in Medicine Meets Virtual Reality (MMVR), vol. 111, Feb. 2005, pp. 191–193.

[76] J. Huang, J. Triedman, N. Vasilyev, Y. Suematsu, R. Cleveland, and P. Dupont, *Imaging Artifacts of Medical Instruments in Ultrasound-Guided Interventions*, Journal of Ultrasound in Medicine, 26 (2007), p. 1303.

[77] X. Huang, N. Hill, J. Ren, G. Guiraudon, D. Boughner, and T. Peters, *Dynamic 3d ultrasound and mr image registration of the beating heart*, in Medical Image Computing and Computer-Assisted Intervention - MICCAI 2005, 8th International Conference, J. S. Duncan and G. Gerig, eds., vol. 3750 of Lecture Notes in Computer Science, Palm Springs, CA, USA, September 2005, Springer Verlag, p. 171.

[78] X. Huang, N. Hill, J. Ren, and T. Peters, *Rapid registration of multimodal images using a reduced number of voxels*, in Proc. of SPIE Vol, vol. 6141, pp. 614116–1.

[79] P. Huber, J. Jenne, R. Rastert, I. Simiantonakis, H. Sinn, H. Strittmatter, D. von Fournier, M. Wannenmacher, and J. Debus, *A New Noninvasive Approach in Breast Cancer Therapy Using Magnetic Resonance Imaging-guided Focused Ultrasound Surgery*, Cancer Research, 61 (2001), pp. 8441–8447.

[80] R. I. Ionasec, B. Georgescu, E. Gassner, S. Vogt, O. Kutter, M. Scheuering, N. Navab, and D. Comaniciu, *Dynamic model-driven quantification and visual evaluation of the aortic valve from 4d ct*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), D. Metaxas, L. Axel, G. Fichtinger, and G. Szekely, eds., Lecture Notes in Computer Science, New York, USA, September 2008, Springer.

[81] G. Ionescu, S. Lavallee, J. Demongeot, et al., *Automated registration of ultrasound with CT images: application to computer assisted prostate radiotherapy and orthopedics*, Lecture Notes in Computer Science, (1999), pp. 768–777.

[82] A. J. Jensen, *Field: A program for simulating ultrasound systems*, in Medical & Biological Engineering & Computing, vol. 34, 1996, pp. 351–353.

[83] J. Jensen and I. Nikolov, *Fast simulation of ultrasound images*, Ultrasonics Symposium, 2000 IEEE, 2 (2000), pp. 1721–1724.

[84] J. A. Jensen, *Linear description of ultrasound imaging systems.* Technical Univ. of Denmark, Ørsted, Denmark, July 1999.

[85] J. A. Jensen, *Simulation of advanced ultrasound systems using Field II*, in IEEE Int. Symp. on Biomedical Imaging (ISBI), Apr. 2004, pp. 636–639.

[86] J. A. Jensen and N. B. Svendsen, *Calculation of pressure fields from arbitrarily shaped, apodized, and excited ultrasound transducers*, IEEE Trans. Ultrason., 39 (1992), pp. 262–267.

[87] B. Kainz, M. Grabner, A. Bornik, S. Hauswiesner, J. Muehl, and D. Schmalstieg, *Ray casting of multiple volumetric datasets with polyhedral boundaries on manycore gpus*, ACM Trans. Graph., 28 (2009), pp. 1–9.

[88] D. Kalkofen, E. Mendez, and D. Schmalstieg, *Interactive Focus and Context Visualization for Augmented Reality*, in Proceedings of the 6th International Symposium on Mixed and Augmented Reality (ISMAR), Nov. 2007, pp. 191–200.

[89] A. Karamalis, *Gpu ultrasound simulation and volume reconstruction*, master's thesis, Technische Universität München, München, Germany, February 2009.

[90] A. Karamalis, W. Wein, O. Kutter, and N. Navab, *Fast hybrid freehand ultrasound volume reconstruction*, in SPIE Medical Imaging, Orlando, Florida, USA, February 2009.

[91] F. Keus, J. De Jong, H. Gooszen, and C. Van Laarhoven, *Laparoscopic versus open cholecystectomy for patients with symptomatic cholecystolithiasis.*, Cochrane database of systematic reviews (Online), (2006).

[92] A. Khamene, P. Bloch, W. Wein, M. Svatos, and F. Sauer, *Automatic registration of portal images and volumetric CT for patient positioning in radiation therapy*, Medical Image Analysis, 10 (2006), pp. 96–112.

[93] A. Khamene, R. Chisu, W. Wein, N. Navab, and F. Sauer, *A novel projection based approach for medical image registration*, in Third International Workshop on Biomedical Image Registration (WBIR), 2006.

[94] A. Khamene and F. Sauer, *A novel phantom-less spatial and temporal ultrasound calibration method*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), Lecture Notes in Computer Science, Springer, 2005, pp. 65–72.

[95] M. J. Kilgard, *Improving shadows and reflections via the stencil buffer*, tech. report, NVIDIA, 2000. Online available as `http://developer.nvidia.com/attach/6641` [Online; accessed 31-Dec-2009].

[96] E. Kilgariff and R. Fernando, *The GeForce 6 series GPU architecture*, in SIGGRAPH '05: ACM SIGGRAPH 2005 Courses, New York, NY, USA, 2005, ACM, p. 29.

[97] G. Kindlmann, *Transfer functions in direct volume rendering: Design, interface, interaction*, Siggraph Course Notes, 5 (2002).

[98] A. P. King, P. J. Edwards, C. R. Maurer, Jr., D. A. de Cunha, D. J. Hawkes, D. L. G. Hill, R. P. Gaston, M. R. Fenlon, A. J. Strong, C. L. Chandler, A. Richards, and M. J. Gleeson, *Design and evaluation of a system for microscope-assisted guided interventions*, IEEE Trans. Medical Imaging, 19 (2000), pp. 1082–1093.

[99] D. B. Kirk and W. mei W. Hwu, *Programming Massively Parallel Processors*, MIT Press, 2010.

[100] J. Klein, D. Bartz, O. Friman, M. Hadwiger, B. Preim, F. Ritter, A. Vilanova, and G. Zachmann, *Advanced algorithms in medical computer graphics*, Proc. Eurographics, State-of-the-Art Report, Eurographics Association,(Crete, Greece), (2008).

[101] J. Kniss, G. Kindlmann, and C. Hansen, *Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets*, in Proceedings of the conference on Visualization'01, IEEE Computer Society Washington, DC, USA, 2001, pp. 255–262.

[102] J. Kniss, G. Kindlmann, and C. Hansen., *Multidimensional Transfer Functions for Volume Rendering*, Elsevier, 2005, ch. 9, pp. 189–210.

[103] A. Kratz, M. Hadwiger, A. Fuhrmann, R. Splechtna, and K. Bühler, *GPU-Based High-Quality Volume Rendering For Virtual Environments*, in International Workshop for Augmented Environments for Medical Imaging and Computer-aided Surgery (AMI-ARCS), Copenhagen, Denmark, Oct. 2006, MICCAI Society, Springer.

[104] A. Krieger, R. Susil, C. Menard, J. Coleman, G. Fichtinger, E. Atalar, and L. Whitcomb, *Design of a novel MRI compatible manipulator for image guided prostate interventions*, Biomedical Engineering, IEEE Transactions on, 52 (2005), pp. 306–313.

[105] J. Krüger, J. Schneider, and R. Westermann, *ClearView: An interactive context preserving hotspot visualization technique*, IEEE Trans. Visualization and Computer Graphics, 12 (2006), pp. 941–948.

[106] J. Kruger and R. Westermann, *Acceleration Techniques for GPU-based Volume Rendering*, Proceedings of the 14th IEEE Visualization 2003 (VIS'03), (2003).

[107] J. Krüger and R. Westermann, *Linear algebra operators for GPU implementation of numerical algorithms*, ACM Transactions on Graphics (TOG), 22 (2003), pp. 908–916.

[108] A. Kubias, F. Deinzer, T. Feldmann, D. Paulus, B. Schreiber, and T. Brunner, *2D/3D Image Registration on the GPU*, Pattern Recognition and Image Analysis, 18 (2008), pp. 381–389.

[109] O. Kutter, A. Aichert, C. Bichlmeier, S. M. Heining, B. Ockert, E. Euler, and N. Navab, *Real-time Volume Rendering for High Quality Visualization in Augmented Reality*, in International Workshop for Augmented Environments for Medical Imaging and Computer-aided Surgery (AMI-ARCS), New York, USA, Sept. 2008, MICCAI Society.

[110] O. Kutter, A. Karamalis, W. Wein, and N. Navab, *A gpu-based framework for simulation of medical ultrasound*, in SPIE Medical Imaging, Orlando, Florida, USA, February 2009.

[111] O. Kutter, R. Shams, and N. Navab, *Visualization and GPU-accelerated simulation of medical ultrasound from CT images*, Computer Methods and Programs in Biomedicine, 94 (2009), pp. 250–266.

[112] O. Kutter, W. Wein, and N. Navab, *Multi-modal registration based ultrasound mosaicing*, in MICCAI09, G.-Z. Yang, D. Hawkes, D. Rueckert, A. Noble, and C. Taylor, eds., vol. 5761 of Lecture Notes in Computer Science, Springer, 2009, pp. 763–770.

[113] T. Lange, N. Papenberg, S. Heldmann, J. Modersitzki, B. Fischer, H. Lamecker, and P. Schlag, *3D ultrasound-CT registration of the liver using combined landmark-intensity information*, International Journal of Computer Assisted Radiology and Surgery, 4 (2009), pp. 79–88.

[114] D. A. LaRose, *Iterative X-Ray/CT registration using accelerated volume rendering*, PhD thesis, Carnegie Mellon University, May 2001.

[115] A. Lefohn, S. Sengupta, J. Kniss, R. Strzodka, and J. Owens, *Glift: Generic, efficient, random-access GPU data structures*, ACM Transactions on Graphics (TOG), 25 (2006), pp. 60–99.

[116] M. Lerotic, A. J. Chung, G. Mylonas, and G.-Z. Yang:, *pq -space based non-photorealistic rendering for augmented reality*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), N. Ayache, S. Ourselin, and A. Maeder, eds., Lecture Notes in Computer Science, Springer, 2007, pp. 102–109.

[117] A. Leroy, P. Mozer, Y. Payan, and J. Troccaz, *Rigid registration of freehand 3D ultrasound and CT-Scan kidney images*, in Medical Image Computing and Computer-Assisted Intervention – MICCAI 2004, Lecture Notes in Computer Science, Springer, September 2004, p. 837ff.

[118] M. Levoy, *Display of surfaces from volume data*, IEEE Computer Graphics and Applications, 8 (1988), pp. 29–37.

[119] E. Lindholm, J. Nickolls, S. Oberman, and J. Montrym, *NVIDIA Tesla: A Unified Graphics and Computing Architecture*, IEEE Micro, (2008), pp. 39–55.

[120] S. Lindholm, P. Ljung, M. Hadwiger, and A. Ynnerman, *Fused Multi-Volume DVR using Binary Space Partitioning*, in Computer Graphics Forum, vol. 28, Blackwell Publishing Ltd, 2009, pp. 847–854.

[121] C. Linte, M. Wierzbicki, J. Moore, S. Little, G. Guiraudon, and T. Peters, *Towards subject-specific models of the dynamic heart for image-guided mitral valve surgery*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), N. Ayache, S. Ourselin, and A. Maeder, eds., vol. 4792 of Lecture Notes in Computer Science, Springer, 2007, pp. 94–101.

[122] C. A. Linte, A. D. Wiles, N. Hill, J. Moore, C. Wedlake, G. Guiraudon, D. Jones, D. Bainbridge, and T. M. Peters, *An augmented reality environment for image-guidance of off-pump mitral valve implantation*, in Medical Imaging 2007: Visualization and Image-Guided Procedures, K. R. Cleary

and M. I. Miga, eds., Proceedings of SPIE, San Diego, California, USA, February 2007.

[123] P. Ljung, F. Hernell, and A. Ynnerman, *Local ambient occlusion in direct volume rendering*, IEEE Trans. Visualization and Computer Graphics, 99 (2009).

[124] K. Low, A. Ilie, G. Welch, and A. Lastra, *Combining head-mounted and projector-based displays for surgical training*, Virtual Reality, 2003. Proceedings. IEEE, (2003), pp. 110–117.

[125] D. Luebke, M. Harris, J. Krüger, T. Purcell, N. Govindaraju, I. Buck, C. Woolley, and A. Lefohn, *GPGPU: general purpose computation on graphics hardware*, Proceedings of the conference on SIGGRAPH 2004 course notes, (2004).

[126] E. Lum, B. Wilson, and K. Ma, *High-quality lighting and efficient pre-integration for volume rendering*, Proceedings Joint Eurographics-IEEE TVCG Symposium on Visualization 2004 (VisSym04), (2004), pp. 25–34.

[127] G. Lyman, A. Giuliano, M. Somerfield, A. Benson III, D. Bodurka, H. Burstein, A. Cochran, H. Cody III, S. Edge, S. Galper, et al., *American Society of Clinical Oncology guideline recommendations for sentinel lymph node biopsy in early-stage breast cancer*, Journal of clinical oncology, 23 (2005), p. 7703.

[128] G. Lyman, K. Van Zee, and S. Edge, *Impact and Remaining Challenges in the Use of Sentinel Lymph Node Biopsy in Early-stage Breast Cancer*, American Society of Clinical Oncology Educational Book, 2009 (2009), p. 346.

[129] D. Magee, Y. Zhu, R. Ratnalingam, P. Gardner, and D. Kessel, *An augmented reality simulator for ultrasound guided needle placement training*, Medical and Biological Engineering and Computing, 45 (2007), pp. 957–967.

[130] J. B. A. Maintz and M. A. Viergever, *A survey of medical image registration*, Medical Image Analysis, 2 (1998), pp. 1–36.

[131] T. Mastracci, L. Garrido-Olivares, C. Cinà, and C. Clase, *Endovascular repair of ruptured abdominal aortic aneurysms: a systematic review and meta-analysis*, Journal of Vascular Surgery, 47 (2008), pp. 214–221.

[132] H. Maul, A. Scharf, P. Baier, M. Wüstemann, H. H. Günter, G. Gebauer, and C. Sohn, *Ultrasound simulators: experience with the SonoTrainer and comparative review of other training systems*, Ultrasound Obstetrcs and Gynecology, 24 (2004), pp. 581–585.

[133] M. McCool and S. Du Toit, *Metaprogramming GPUs with Sh*, AK Peters, Ltd., 2004.

[134] M. Mellor and M. Brady, *Phase mutual information as a similarity measure for registration*, Medical Image Analysis, 9 (2005), pp. 330–343.

[135] L. Mercier, T. Langø, F. Lindseth, and L. Collins, *A review of calibration techniques for freehand 3-D ultrasound systems*, Ultrasound in Medicine & Biology, 31 (2005), pp. 143–165.

[136] R. Miller and T. Hollingsworth, *Rigid endoscopes: Optical and design considerations*, British Medical Bulletin, 42 (1986), pp. 226–229.

[137] J. Modersitzki, *Numerical Methods for Image Registration*, Oxford University Press, New York, 2004.

[138] N. Navab, M. Feuerstein, and C. Bichlmeier, *Laparoscopic virtual mirror - new interaction paradigm for monitor based augmented reality*, in Virtual Reality, Charlotte, North Carolina, USA, March 2007, pp. 43–50.

[139] N. Navab, M. Mitschke, and O. Schütz, *Camera-augmented mobile c-arm (camc) application: 3d reconstruction using a low-cost mobile c-arm*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), C. Taylor and A. Colchester, eds., vol. 1679 of Lecture Notes in Computer Science, Springer, 1999, pp. 688–697.

[140] T. Nelson and D. Pretorius, *Three-dimensional ultrasound imaging*, Ultrasound Med Biol, 24 (1998), pp. 1243–1270.

[141] H. Ngyuyen, *GPU Gems 3: Programming Techniques for High Performance Graphics and General-Purpose Computation*, 2007.

[142] D. Ni, Y. Qu, X. Yang, Y. Chui, T. Wong, S. Ho, and P. Heng, *Volumetric Ultrasound Panorama Based on 3D SIFT*, in Proceedings of the 11th International Conference on Medical Image Computing and Computer-Assisted Intervention, Part II, Springer-Verlag Berlin, Heidelberg, 2008, pp. 52–60.

[143] P. Novotny, *Real-time processing of three dimensional ultrasound for intracardiac surgery*, PhD thesis, Citeseer, 2007.

[144] P. Novotny, S. Jacobsen, N. Vasilyev, D. Kettler, I. Salgo, P. Dupont, P. Del Nido, and R. Howe, *3D ultrasound in robotic surgery: performance evaluation with stereo displays.*, Int J Med Robot, 2 (2006), pp. 279–85.

[145] P. Novotny, J. Stoll, P. Dupont, and R. Howe, *Real-Time Visual Servoing of a Robot Using Three-Dimensional Ultrasound*, Robotics and Automation, 2007 IEEE International Conference on, (2007), pp. 2655–2660.

[146] P. Novotny, J. Stoll, N. Vasilyev, P. del Nido, P. Dupont, T. Zickler, and R. Howe, *GPU based real-time instrument tracking with three-dimensional ultrasound*, Medical Image Analysis, 11 (2007), pp. 458–464.

[147] NVIDIA, *NVIDIA CUDA, Programming Guide, Version 2.3*, July 2009. `http://developer.download.nvidia.com/compute/cuda/2_3/toolkit/docs/NVIDIA_CUDA_ProgrammingGuide_2.3.pdf`.

[148] J. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Kruger, A. Lefohn, and T. Purcell, *A Survey of General-Purpose Computation on Graphics Hardware*, Computer Graphics Forum, 26 (2007), pp. 80–113.

[149] J. Park, M. Chung, S. Hwang, Y. Lee, D. Har, and H. Park, *Visible korean human: Improved serially sectioned images of the entire body*, IEEE Trans. Medical Imaging, 24 (2005), pp. 352–360.

[150] P. Paul, O. Fleig, and P. Jannin, *Augmented virtuality based on stereoscopic reconstruction in multimodal image-guided neurosurgery: Methods and performance evaluation*, IEEE Trans. Medical Imaging, 24 (2005), pp. 1500–1511.

[151] X. Pennec, P. Cachier, and N. Ayache, *Tracking brain deformations in time sequences of 3D US images*, Pattern Recognition Letters, 24 (2003), pp. 801–813.

[152] G. Penney, D. Barratt, C. Chan, M. Slomczykowski, T. Carter, P. Edwards, and D. Hawkes, *Cadaver validation of intensity-based ultrasound to ct registration*, Medical Image Analysis, 10 (2006), pp. 385–395.

[153] G. Penney, J. Blackall, M. Hamady, T. Sabharwal, A. Adam, and D. Hawkes, *Registration of freehand 3D ultrasound and magnetic resonance liver images*, Medical Image Analysis, 8 (2004), pp. 81–91.

[154] T. Peters and K. Cleary, eds., *Image-Guided Interventions - Technology and Applications*, Springer, May 2008.

[155] M. Pharr and R. Fernando, *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation (Gpu Gems)*, Addison-Wesley Professional, 2005.

[156] P.J.Edwards, D. D. Hill, D. D. Hawkes, and D. A. Colchester, *Neurosurgical guidance using the stereo microscope*, in Proceedings of Computer Vision, Virtual Reality and Robotics in Medicine, First International Conference, CVRMed'95, 1995.

[157] T. Poon and R. Rohling, *Three-dimensional extended field-of-view ultrasound*, Ultrasound in medicine & biology, 32 (2006), pp. 357–369.

[158] F. Raab, E. Blood, T. Steioner, and H. Jones, *Magnetic position and orientation tracking system*, IEEE Transactions on Aerospace and Electronic Systems, 15 (1979), pp. 709–718.

[159] F. Randima and J. Kilgard Mark, *The Cg Tutorial: The Definitive Guide to Programmable Real-Time Graphics*, USA: Addison-Wesley, 2003.

[160] C. Rezk-Salama, *Production-Ready GPU-Based Monte-Carlo Volume Rendering*, (2007).

[161] A. Roche, X. Pennec, G. Malandain, and N. Ayache, *Rigid registration of 3d ultrasound with mr images: A new approach combining intensity and gradient information*, IEEE Transactions on Medical Imaging, 20 (2001), pp. 1038–1049.

[162] R. Rohling, A. Gee, and L. Berman, *A comparison of freehand three-dimensional ultrasound reconstruction techniques*, Medical Image Analysis, 3 (1999), pp. 339–359.

[163] R. Rost, *OpenGL Shading Language*, Addison-Wesley Professional, 2006.

[164] F. Rousseau, R. Fablet, and C. Barillot, *Robust statistical registration of 3d ultrasound images using texture information*, in IEEE Int. Conf. on Image Processing, ICIP, vol. 3, Citeseer.

[165] D. Rueckert, L. Sonoda, C. Hayes, D. Hill, M. Leach, and D. Hawkes, *Nonrigid registration using free-form deformations: application to breast mr images*, IEEE Trans. Medical Imaging, 18 (1999), pp. 712–721.

[166] D. Ruijters, *Dynamic resolution in gpu-accelerated volume rendering to autostereoscopic multiview lenticular displays*, EURASIP Journal on Advances in Signal Processing, 2009 (2009), p. 8.

[167] D. Ruijters, B. ter Haar-Romeny, and P. Suetens, *Efficient GPU-accelerated elastic image registration*, in Proceedings of the Sixth IASTED International Conference on Biomedical Engineering, International Association of Science and Technology for Development,# 80, 4500- 16 Ave NW, Calgary, Alberta, T 3 B 0 M 6, Canada„ 2008.

[168] C. Salama, M. Keller, and P. Kohlmann, *High-Level User Interfaces for Transfer Function Design with Semantics*, IEEE Trans. Visualization and Computer Graphics, 12 (2006), pp. 1021–1028.

[169] C. R. Salama, *GPU-Based Monte-Carlo Volume Raycasting*, in Proc. Pacific Graphics, 2007.

[170] C. Sandor, S. Uchiyama, and H. Yamamoto, *Visuo-Haptic Systems: Half-Mirrors Considered Harmful*, EuroHaptics Conference, 2007 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2007. Second Joint, (2007), pp. 292–297.

[171] F. Sauer, A. Khamene, B. Bascle, and G. J. Rubino, *A head-mounted display system for augmented reality image guidance: Towards clinical evaluation for imri-guided neurosurgery*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), Lecture Notes in Computer Science, London, UK, 2001, Springer, pp. 707–716.

[172] F. Sauer, A. Khamene, B. Bascle, S. Vogt, and G. J. Rubinob, *Augmented reality visualization in imri operating room: System description and pre-clinical testing*, in Proceedings of SPIE, Medical Imaging, vol. 4681, 2002, pp. 446–454.

[173] F. Sauer, A. Khamene, and S. Vogt, *An augmented reality navigation system with a single-camera tracker: System design and needle biopsy phantom trial*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), vol. 2489 of Lecture Notes in Computer Science, Springer, 2002, pp. 116–124.

[174] F. Sauer, F. Wenzel, S. Vogt, Y. Tao, Y. Genc, and A. Bani-Hashemi, *Augmented workspace: designing an ar testbed*, in Proc. IEEE and ACM Int'l Symp. on Augmented Reality, 2000, pp. 47–53.

[175] H. Scharsach, *Advanced raycasting for virtual endoscopy on consumer graphics hardware*, master's thesis, Technische Universität Wien, Wien, Austria, April 2005.

[176] H. Scharsach, M. Hadwiger, A. Neubauer, S. Wolfsberger, and K. Bühler, *Perspective Isosurface and Direct Volume Rendering for Virtual Endoscopy Applications*, Proceedings of Eurovis/IEEE-VGTC Symposium on Visualization 2006, (2006), pp. 315–322.

[177] T. Scheuermann and J. Hensley, *Efficient histogram generation using scattering on GPUs*, Proceedings of the 2007 symposium on Interactive 3D graphics and games, (2007), pp. 33–37.

[178] O. Schillaci, *Hybrid SPECT/CT: a new era for SPECT imaging?*, European Journal of Nuclear Medicine and Molecular Imaging, 32 (2005), pp. 521–524.

[179] T. Schiwietz, S. Bose, J. Maltz, and R. Westermann, *A fast and high-quality cone beam reconstruction pipeline using the GPU*, Medical Imaging 2007: Physics of Medical Imaging. Edited by Hsieh, Jiang; Flynn, Michael J.. Proceedings of the SPIE„ 6510 (2007), p. 65105H.

[180] L. Seiler, D. Carmean, E. Sprangle, T. Forsyth, M. Abrash, P. Dubey, S. Junkins, A. Lake, J. Sugerman, R. Cavin, et al., *Larrabee: a many-core x86 architecture for visual computing*, (2008).

[181] P. Serruys, M. Kutryk, and A. Ong, *Coronary-artery stents*, The New England journal of medicine, 354 (2006), p. 483.

[182] K. Shah and R. Weissleder, *Molecular optical imaging: applications leading to the development of present day therapeutics.*, NeuroRx, 2 (2005), pp. 215–225.

[183] R. Shams, R. Hartley, and N. Navab, *Real-time simulation of medical ultrasound from CT images*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), D. Metaxas, L. Axel, G. Fichtinger, and G. Szekely, eds., Lecture Notes in Computer Science, New York, USA, September 2008, Springer, pp. 734–741.

[184] R. Shams and R. A. Kennedy, *Efficient histogram algorithms for NVIDIA CUDA compatible devices*, in Proc. Int. Conf. on Signal Processing and Communications Systems (ICSPCS), Gold Coast, Australia, Dec. 2007, pp. 418–422.

[185] R. Shams, P. Sadeghi, R. A. Kennedy, and R. Hartley, *Parallel computation of mutual information on the GPU with application to real-time registration of 3D medical images*, Computer Methods and Programs in Biomedicine, (2009).

[186] R. Shams, P. Sadeghi, R. A. Kennedy, and R. I. Hartley, *A survey of high performance medical image registration on multi-core, GPU and distributed architectures*, IEEE Signal Processing Mag. (under review), (2010).

[187] R. Shekhar and V. Zagrodsky, *Mutual information-based rigid and nonrigid registration of ultrasound volumes*, IEEE Transactions on Medical Imaging, 21 (2002), pp. 9–22.

[188] R. Shekhar, V. Zagrodsky, M. Garcia, and J. Thomas, *Registration of real-time 3-D ultrasound images of the heart for novel 3-D stress echocardiography*, IEEE transactions on medical imaging, 23 (2004), pp. 1141–1149.

[189] T. Sielhorst, M. A. Bauer, O. Wenisch, G. Klinker, and N. Navab, *Online estimation of the target registration error for n-ocular optical tracking systems*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), N. Ayache, S. Ourselin, and A. Maeder, eds., vol. 4792 of Lecture Notes in Computer Science, Brisbane, Australia, Oct. 2007, MICCAI Society, Springer, pp. 652–659.

[190] T. Sielhorst, M. Feuerstein, J. Traub, O. Kutter, and N. Navab, *Campar: A software framework guaranteeing quality for medical augmented reality*, International Journal of Computer Assisted Radiology and Surgery, 1 (2006), pp. 29–30.

[191] C. Sigg, M. Hadwiger, M. Gross, and K. Bühler, *Real-time high-quality rendering of isosurfaces*, tech. report, TR-VRVis-2004-015, VRVis Research Center, 2004.

[192] L. Soler and J. Marescaux, *Patient-specific Surgical Simulation*, World Journal of Surgery, 32 (2008), pp. 208–212.

[193] J. Stallkamp and M. Wapler, *UltraTrainer: a training system for medical ultrasound examination*, in Medicine Meets Virtual Reality (MMVR), Jan. 1998.

[194] P. Stefan, *Intra and inter-modal deformable image registration through efficient linear programming and mrfs on the gpu*, master's thesis, Technische Universität München, München, Germany, October 2009.

[195] S. Stegmaier, M. Strengert, T. Klein, and T. Ertl, *A Simple and Flexible Volume Rendering Framework for Graphics-Hardware–based Raycasting*, Proceedings of the International Workshop on Volume Graphics, 5 (2005), pp. 187–195.

[196] R. Steinmeier, R. Fahlbusch, O. Ganslandt, C. Nimsky, M. Buchfelder, M. Kaus, T. Heigl, G. Lenz, R. Kuth, and W. Huk, *Intraoperative magnetic resonance imaging with the magnetom open scanner: concepts, neurosurgical indications, and procedures: a preliminary report*, Neurosurgery, 43 (1998), pp. 739–748.

[197] P. R. Stepanishen, *Transient radiation from pistons in an infinite planar baffle*, Journal of Acoust. Soc. Am., 49 (1971), pp. 1629–1638.

[198] R. Strzodka, M. Droske, and M. Rumpf, *Image Registration by a Regularized Gradient Flow. A Streaming Implementation in DX9 Graphics Hardware*, Computing, 73 (2004), pp. 373–389.

[199] Y. Suematsu, J. Martinez, B. Wolf, G. Marx, J. Stoll, P. DuPont, R. Howe, J. Triedman, and P. del Nido, *Three-dimensional echo-guided beating heart surgery without cardiopulmonary bypass: atrial septal defect closure in a swine model*, 2005.

[200] Y. Suematsu, G. Marx, J. Stoll, P. DuPont, R. Cleveland, R. Howe, J. Triedman, T. Mihaljevic, B. Mora, B. Savord, et al., *Three-dimensional echocardiography–guided beating-heart surgery without cardiopulmonary bypass: A feasibility study*, The Journal of Thoracic and Cardiovascular Surgery, 128 (2004), pp. 579–587.

[201] R. Szeliski, *Image alignment and stitching: A tutorial*, tech. report, Microsoft Research, 2006.

[202] A. Tahmasebi, K. Hashtrudi-Zaad, D. Thompson, and P. Abolmaesumi, *A framework for the design of a novel haptic-based medical training simulator*, IEEE Transactions on Information Technology in Biomedicine, 12 (2008), pp. 658–666.

[203] J. Tardif, S. Roy, and J. Meunier, *Projector-based augmented reality in surgery without calibration*, in Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, vol. 1, 2003.

[204] C. Terkamp, G. Kirchner, J. Wedemeyer, A. Dettmer, J. Kielstein, H. Reindell, J. Bleck, M. Manns, and M. Gebel, *Simulation of abdomen sonography. evaluation of a new ultrasound simulator*, Ultraschall in Med, 24 (2003), pp. 239–244.

[205] S. Towfigh, F. Chen, R. Mason, N. Katkhouda, L. Chan, and T. Berne, *Laparoscopic appendectomy significantly reduces length of stay for perforated appendicitis*, Surgical endoscopy, 20 (2006), pp. 495–499.

[206] J. Traub, H. Heibel, P. Dressel, S. Heining, R. Graumann, and N. Navab, *A multi-view opto-xray imaging system: Development and first application in trauma surgery*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), N. Ayache, S. Ourselin, and A. Maeder, eds., Lecture Notes in Computer Science, 2007.

[207] J. Traub, P. Stefan, S.-M. Heining, T. Sielhorst, C. Riquarts, E. Euler, and N. Navab, *Hybrid navigation interface for orthopedic and trauma surgery*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), R. Larsen, M. Nielsen, and J. Sporring, eds., vol. 4190 of Lecture Notes in Computer Science, Copenhagen, Denmark, Oct. 2006, MICCAI Society, Springer, pp. 373–380.

[208] ——, *Towards a hybrid navigation interface: Comparison of a slice based navigation system with in-situ visualization*, in Medical Imaging and Augmented Reality, G.-Z. Yang, T. Jiang, D. Shen, L. Gu, and J. Yang, eds., vol. 4091 of Lecture Notes in Computer Science, Shanghai, China, Aug. 2006, Springer, pp. 179–186.

[209] G. E. Tupholme, *Generation of acoustic pulses by baffled plane pistons*, Mathematika, 16 (1969), pp. 209–224.

[210] A. Vahanian, O. Alfieri, N. Al-Attar, M. Antunes, J. Bax, B. Cormier, A. Cribier, P. De Jaegere, G. Fournial, A. Kappetein, et al., *Transcatheter valve implantation for patients with aortic stenosis: a position statement from the European Association of Cardio-Thoracic Surgery and the European Society of Cardiology, in collaboration with the European Association of Percutaneous Cardiovascular Interventions*, European Journal of Cardio-Thoracic Surgery, (2008).

[211] R. Vecchio, B. MacFayden, and F. Palazzo, *History of laparoscopic surgery*, Panminerva Medica, 42 (2000), pp. 87–90.

[212] C. Vetter, C. Guetter, C. Xu, and R. Westermann, *Non-rigid multi-modal registration on the GPU*, in Proceedings of SPIE, vol. 6512, 2007, p. 651228.

[213] F. P. Vidal, N. W. John, A. E. Healey, and D. A. Gould, *Simulation of ultrasound guided needle puncture using patient specific data with 3d textures and volume haptics*, Computer Animation and Virtual Worlds, 19 (2008), pp. 111–127.

[214] I. Viola, A. Kanitsar, and M. Groller, *Importance-driven volume rendering*, Proceedings of IEEE Visualization, 4 (2004), pp. 139–145.

[215] S. Vogt, A. Khamene, and F. Sauer, *Reality augmentation for medical procedures: System architecture, single camera marker tracking, and system evaluation*, International Journal of Computer Vision, 70 (2006), pp. 179–190.

[216] C. Wachinger, *Three-dimensional ultrasound mosaicing*, master's thesis, Technische Universität München, München, Germany, March 2007.

[217] C. Wachinger and N. Navab, *Similarity Metrics and Efficient Optimization for Simultaneous Registration*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

[218] C. Wachinger, W. Wein, and N. Navab, *Three-dimensional ultrasound mosaicing*, in MICCAI07, N. Ayache, S. Ourselin, and A. Maeder, eds., vol. 4792 of LNCS, Brisbane, Australia, Oct. 2007, MICCAI Society, Springer, pp. 327–335.

[219] ——, *Registration strategies and similarity measures for three-dimensional ultrasound mosaicing*, Academic Radiology, 15 (2008), pp. 1404–1415.

[220] L. Wang, J. Traub, S. M. Heining, S. Benhimane, E. Euler, R. Graumann, and N. Navab, *Parallax-free long bone x-ray image stitching*, in International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), London, UK, September 2009.

[221] M. Weidenbach, C. Wick, S. Pieper, K. Quast, T. Fox, G. Grunst, and R. DA, *Augmented reality simulator for training in two-dimensional echocardiography*, Computers and Biomedical Research, 33 (2000), pp. 11–022.

[222] W. Wein, S. Brunke, A. Khamene, M. Callstrom, and N. Navab, *Automatic ct-ultrasound registration for diagnostic imaging and image-guided intervention*, Medical Image Analysis, 12 (2008), pp. 577–585.

[223] W. Wein, E. Camus, M. John, M. Diallo, C. Duong, A. Al-Ahmad, R. Fahrig, A. Khamene, and C. Xu, *Towards guidance of electrophysiological procedures with real-time 3d intracardiac echocardiography fusion to c-arm ct*, in Proc. Int'l Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), G.-Z. Yang, D. Hawkes, D. Rueckert, A. Noble, and C. Taylor, eds., vol. 5761 of Lecture Notes in Computer Science, Springer, 2009, pp. 9–16.

[224] W. Wein, A. Khamene, D.-A. Clevert, O. Kutter, and N. Navab, *Simulation and fully automatic multimodal registration of medical ultrasound*, in MICCAI07, N. Ayache, S. Ourselin, and A. Maeder, eds., vol. 4791 of LNCS, Brisbane, Australia, Oct. 2007, MICCAI Society, Springer, pp. 136–143.

[225] W. Wein, O. Kutter, A. Aichert, D. Zikic, A. Kamen, and N. Navab, *Automatic non-linear mapping of pre-procedure ct volumes to 3d ultrasound*, in IEEE International Symposium on Biomedical Imaging, Rotterdam, The Netherlands, April 2010.

[226] W. Wein, B. Röper, and N. Navab, *Integrating diagnostic B-mode ultrasonography into CT-based radiation treatment planning*, IEEE Trans. Medical Imaging, 26 (2007), pp. 866–879.

[227] T. Wendler, K. Herrmann, A. Schnelzer, T. Lasser, J. Traub, O. Kutter, K. Scheidhauer, T. Schuster, M. Kiechle, M. Schwaiger, N. Navab, S. I. Ziegler, and A. K. Buck, *First demonstration of 3d lymphatic mapping in breast cancer using freehand spect*, European Journal of Nuclear Medicine and Molecular Imaging, (2010), pp. 1–10.

[228] T. Wendler, K. Herrmann, A. Schnelzer, T. Lasser, J. Traub, O. Kutter, T. Schuster, M. Kiechle, M. Schwaiger, N. Navab, S. Ziegler, and A. Buck, *Clinical introduction of freehand spect for image-guided sentinel lymph node biopsy*, in Proceedings of EANM 2009, Barcelona, Spain, Oct. 2009, European Association of Nuclear Medicine.

[229] M. N. Wernick, *Emission Tomography. The Fundamentals of PET and SPECT*, Academic Press, 2004.

[230] J. Woo, B. Hong, C. Hu, K. Shung, C. Kuo, and P. Slomka, *Non-Rigid Ultrasound Image Registration Based on Intensity and Local Phase Information*, Journal of Signal Processing Systems, 54 (2009), pp. 33–43.

[231] D. L. Wright, J. P. Rolland, and A. R. Kancherla, *Using virtual reality to teach radiographic positioning.*, Radiologic Technology, 66 (1995), pp. 233–8.

[232] Z. Yaniv, *Rigid Registration*, Image-Guided Interventions - Technology and Applications, Springer, 2008, ch. 6.

[233] J. A. Zagzebski, *Essentials of ultrasound physics*, Elsevier, 1996.

[234] Q. Zhang, R. Eagleson, and T. Peters, *Rapid Voxel Classification Methodology for Interactive 3D Medical Image Visualization*, in MICCAI07, N. Ayache, S. Ourselin, and A. Maeder, eds., vol. 4792 of LNCS, Brisbane, Australia, Oct. 2007, MICCAI Society, Springer.

[235] W. Zhang, J. Noble, and J. Brady, *Real time 3-D ultrasound to MR cardiovascular image registration using a phase-based approach*, in 3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2006, 2006, pp. 666–669.

[236] Z. Zhang, *A flexible new technique for camera calibration*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (2000), pp. 1330–1334.

[237] Y. Zhu, D. Magee, R. Ratnalingam, and D. Kessel, *A virtual ultrasound imaging system for the simulation of ultrasound-guided needle insertion procedures*, Proceedings of Medical Image Understanding and Analysis (MIUA), 1 (2006), pp. 61–65.

[238] ——, *A training system for ultrasound-guided needle insertion procedures.*, in MICCAI07, N. Ayache, S. Ourselin, and A. Maeder, eds., vol. 4791 of LNCS, Brisbane, Australia, Oct. 2007, MICCAI Society, Springer, p. 566.

[239] D. Zikic, B. Glocker, O. Kutter, M. Groher, N. Komodakis, A. Khamen, N. Paragios, and N. Navab, *Linear intensity-based image registration by markov random fields and discrete optimization*, Medical Image Analysis, In Press, Accepted Manuscript (2010), pp. –.

[240] D. Zikic, B. Glocker, O. Kutter, M. Groher, N. Komodakis, A. Khamene, N. Paragios, and N. Navab, *Markov random field optimization for intensity-based 2d-3d registration*, in Proceedings of SPIE Medical Imaging, vol. 7623, California, USA, February 2010, p. 762334.

[241] D. Zikic, W. Wein, A. Khamene, D.-A. Clevert, and N. Navab, *Fast deformable registration of 3d-ultrasound using a variational approach*, in Proc. Int'l

Conf. Medical Image Computing and Computer Assisted Intervention (MICCAI), R. Larsen, M. Nielsen, and J. Sporring, eds., vol. 4190, MICCAI Society, Springer, Oct. 2006, pp. 915–923.

[242] B. Zitová and J. Flusser, *Image registration methods: a survey*, Image and Vision Computing, 21 (2003), pp. 977–1000.